COMPUTER PROGRAMMING WITH

Java

PROGRAMMING LANGUAGE

M. Turhan Çoban

COMPUTER PROGRAMMING WITH JAVA PROGRAMMING LANGUAGE

JAVA 8 version

M. Turhan ÇOBAN

EGE University, School of Engineering, Dept of Mechanical Engineering

turhan.coban@turhancoban.com

COMPUTER PROGRAMMING WITH JAVA PROGRAMMING LANGUAGE

INDEX

1.0 INTRODUCTION TO JAVA PROGRAMMING LANGUAGE

1.1 PREAPARATION TO ESTABLISH JAVA PROGRAMMING ENVIRONMENT IN YOUR COMPUTER
1.2 INTRODUCTION TO JAVA PROGRAMING LANGUAGE
1.3 INTEGER AND REAL VARIABLES AND ARITHMETIC CALCULATIONS
1.4 WEEK 1 CLASS EXERSIZES
1.5 WEEK 1 HOMEWORKS

2.0 INTRODUCTION TO JAVA PROGRAMING LANGUAGE

2.1 STRING VARIABLES
2.2 CHARACTER VARIABLES
2.3 THE TERM FINAL AND CONSTANTS IN JAVA
2.4 WEEK 2 CLASS EXERSIZES
2.5 WEEK 2 HOMEWORKS

3.0 INTRODUCTION TO JAVA PROGRAMING LANGUAGE

3.1 BOOLEAN VARIABLES
3.2 IF CONTROL STRUCTURE
3.3 BIT PROCESSING IN JAVA
3.4 WEEK 3 CLASS EXERSIZES
3.5 WEEK 3 HOMEWORKS

4.0 INTRODUCTION TO JAVA PROGRAMING LANGUAGE

4.1 WHILE LOOP STRUCTURE
4.2 DO-WHILE LOOP STRUCTURE
4.3 FOR LOOP STRUCTURE
4.4 SWITCH-CASE STRUCTURE
4.5 WEEK 4 CLASS EXERCIZES
4.6 WEEK 4 HOMEWORKS

5.0 METHODS AND INTRODUCTION TO CLASSES

5.1 METHODS
5.2 INTRODUCTION TO CLASS CONCEPT AND DYNAMIC METHOD
5.3 RECURSİVE METHODS
5.4 USING METHODS WITH THE SAME NAME IN THE SAME CLASS (OVERLOADING)
5.5 WEEK 5 CLASS EXERCIZES
5.6 WEEK 5 HOMEWORKS

6.0 CLASS STRUCTURE

6.1 GENERAL DEFINITION OF CLASS
6.2 USAGE OF TERM THIS
6.3 HIDING OF CLASS VARIABLES FROM OUTSIDE WORLD
6.4 WEEK 6 CLASS EXERSIZES
6.5 WEEK 6 HOMEWORKS

7.0 CLASS STRUCTURE: COMPOSITION AND INHERITANCE

7.1 COMPOSITION AND PROTECTED, PUBLIC AND PRIVATE
7.2 INHERITANCE
7.3 SOME REAL CLASS EXAMPLES
7.4 WEEK 7 CLASS EXERSIZES

7.5 WEEK 7 HOMEWORKS

8.0 ARRAYS

8.1 GENERAL DEFINITION AN ONE DIMENSIONAL ARRAY VARIABLES
8.2 ONE DIMENSIONAL ARRAY VARIABLES OF OBJECTS
8.3 ENHANCED FOR STRUCTURE
8.4 MULTI-DIMENSIONAL ARRAY VARIABLES OF OBJECTS
8.5 WEEK 8 CLASS EXERSIZES
8.6 WEEK 8 HOMEWORKS

9.0 POLYMORPHISM, ABSTRACT CLASSES, INTERFACE, INNER CLASSES AND LAMBDA EXPRESSIONS

9.1 ABSTRACT CLASS
9.2 INTERFACE
9.3 INNER CLASSES AND LAMBDA EXPRESSIONS
9.4 WEEK 8 CLASS EXERSIZES
9.5 WEEK 8 HOMEWORKS

10.0 GRAPHICS

10.1 JAVA SWING GRAPHICS ENVIRONMENT AND JFRAME AND JPANEL CLASSES AND GRAPHIC DRAWING METHODS
10.2 JAVA SWING GRAPHICS ENVIRONMENT AND JAPPLET AND JPANEL CLASSES
10.3WEEK10 CLASS EXERSIZES
10.4WEEK 10 HOMEWORKS

11.0 GRAPHIC CONTROL

11.1 JBUTTON CONTROL
11.2 JLABEL, JTEXTFIELD AND JTEXTAREA GRAPHIC CONTROL CLASSES
11.3 WEEK 11 CLASS EXERSIZES
11.4 WEEK 11 HOMEWORK EXERSIZES

12.0 MORE GRAPHIC CONTROLS

12.1 MOUSE CONTROL
12.2 CONTROL BOX
12.3 RADIO CONTROL
12.4 LIST CONTROL JSCROLLPANE FORMAT STATEMENT
12.5 MENU CONTROL
12.6 JSLIDER GRAPHIC CONTROL
12.7 FORMATTING GRAPHIC PAGE
12.8 WEEK 12 CLASS EXERCIZES
12.9 WEEK 12 HOMEWORKS

13.0 EXCEPTION HANDLING (ERROR CONTROL)

13.1 GENERAL DEFINITIONS
13.2 WEEK 1(13) CLASS EXERSIZES
13.3 HOMEWORK 1(13)

14.0 ADVANCED DATA STRUCTURES

14.1 GENERIC METHODS
14.2 GENERIC CLASSES
14.3 USING LAMBDA EXPRESSIONS WITH GENERIC CLASSES
14.4 STRINGTOKENIZER CLASS
14.5 STRINGBUFFER CLASS
14.6 DYNAMIC MEMORY ALLOCATTED CLASSES : LIST, ARRAYLIST, VECTOR, LINKEDLIST,QUE CLASSES
14.7 DYNAMIC ARRAY STRUCTURES : TREES
14.8 DYNAMIC ARRAY STRUCTURES : HASHTABLES

14.9 DYNAMIC ARRAY STRUCTURES : NAVIGABLESET
14.10 A DYNAMIC ARRAY STRUCTURES : AN APPLICATION PROGRAM USING ARRAYLIST TO STORE DATA
14.11 WEEK 2(chapter 14) CLASS EXERSIZES
14.12 WEEK 2HOMEWORK 2(chapter 14)
14.13 WEEK 3(chapter 14) CLASS EXERSIZES
14.14 WEEK 3 HOMEWORK ASSIGNMENTS

15.0 FILES, READING AND WRITING DATA FROM FILES
15.1 FILE CLASS
15.2 READING SEQUENTIAL FILES THROUGH CHANNELS
15.2.1 BUFFERED READER CLASS
15.2.2 SCANNER CLASS
15.3 WRITING INTO SEQUENTIAL FILES THROUGH CHANNELS
15.4 ZIP,UNZIP CLASSES
15.5 WEEK 5(chapter 15) CLASS EXERSIZES
15.6 WEEK 5(chapter 15) HOMEWORKS

16.0 DATA BASE PROGRAMMING, ACCESSING DATABASES THROUGH JAVA PROGRAMMING
16.1 MYSQL DATABASE
16.2 USING MYSQL DATABASE WITH SQL COMMANDS IN CONSOLE ENVIRONMENT
16.3 USING MYSQL DATABASE WITH SQL COMMANDS IN JAVA PROGRAMS
16.4 WEEK 6-7(chapter 16) CLASS EXERSIZES
16.5 HOMEWORKS WEEK 6-7(chapter 16)

17. 0 READING/WRITING DATA FROM THE SPREADSHEET PROGRAMS
17.1 GENERAL DEFINITIONS
17.2 READING DATA FROM SPREADHEETS
17.3 ADDING FORMAT INFORMATION INTO SPREADHEETS
17.4 CLASS EXERCIZES 17 (WEEK 8)
17.5 HOMEWORK 17 (WEEK 8)

18. REAL TIME PROGRAMMING AND MULTITHREADING
18.1 PROGRAMMING WITH THREADS
18.2 SYNCHRONISATION
18.4 CLASS EXERCIZES 18 (WEEK 9)
18.5 HOMEWORK 19 (WEEK 9)

19.0 COMPILING JAVA PROGRAMS FROM INSIDE ANOTHER PROGRAM (REAL-TIME PROGRAM COMPILING)
19.1 COMPILING JAVA PROGRAMS IN REAL TIME
19.2 CREATING JAVA PROGRAMS AND COMPILING THEM IN REAL TIME
19.3 CLASS EXERCIZES 19 (WEEK 10)
19.4 HOMEWORK 19 (WEEK 10)

20.0 NETWORKING
20.1 UNIFORM RESOURCE LOCATERS (URL)
20.2 SOCKET PROGRAMMING (TCP/IP)
20.3 DATAGRAMS
20.4 CLASS EXERCIZES 20 (WEEK 11)
20.5 HOMEWORK 20 (WEEK 11)

Summary of Java Statements

Console Input Scanner input = **new** Scanner(System.in); **int** intValue = input.nextInt(); **long** longValue = input.nextLong(); **double** doubleValue = input.nextDouble(); **float** floatValue = input.nextFloat(); String string = input.next(); **Console Output** System.out.println(anyValue);	**GUI Input Dialog** String string = JOptionPane.showInputDialog("Enter input"); **int** intValue = Integer.parseInt(string); **double** doubleValue = Double.parseDouble(string); **Message Dialog** JOptionPane.showMessageDialog(**null**, "Enter input");

Primitive Data Types **byte** 8 bits (from -128 to 127) **short** 16 bits (From -32768 to 32767) **int** 32 bits (From -2157483648 to 2147483647) **long** 64 bits (From -9223372036854775808 to 9223372036854775808) **float** 32 bits (From -3.40292347e+38 to 3.40292347e+38) **double** 64 bits (From 1.7976931348623157e+308 to 1.7976931348623157e+308) **char** 16 bits (Unicode) **boolean** 1 bit (true/false)	**Arithmetic Operators** **+** addition **-** subtraction ***** multiplication **/** division **%** remainder **++var** preincrement **--var** predecrement **var++** postincrement **var--** postdecrement	**Assignment Operators** **=** assignment **+=** addition assignment **-=** subtraction assignment ***=** multiplication assignment **/=** division assignment **%=** remainder assignment

Relational Operators < less than <= less than or equal to > greater than >= greater than or equal to == equal to **!=** not equal	**Logical Operators** **&&** short circuit AND || short circuit OR **!** NOT ^ exclusive OR	**if** (condition1) {statements;} **else if** (condition2) {statements;} **else if** (condition3) {statements;} …… **else** {statements;}

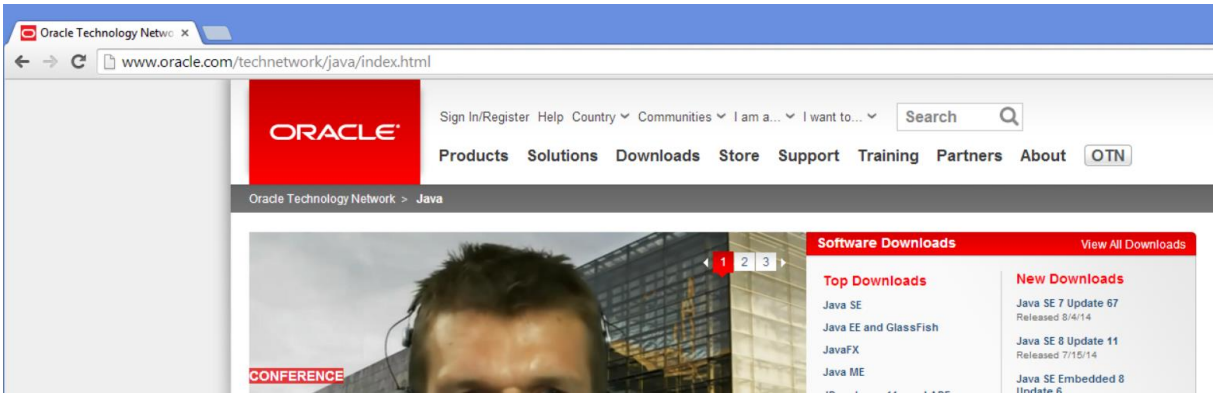
switch Statements
switch (intExpression) {
case value1:
statements;
break;
...
case valuen:
statements;
break;
default:
statements;
}
	While and do-while loop Statements
	
	while (condition) {
	statements;
	}
	
	do {
	statements;
	} **while** (condition);
		For loop statements
		
		for (init; condition;adjustment)
		{
		statements;
		}

Frequently Used Static Constants/Methods Math.PI Math.exp() Math.random() Math.pow(a, b) System.currentTimeMillis() System.out.println(anyValue) JOptionPane.showMessageDialog(**null**, message) JOptionPane.showInputDialog(prompt-message) Integer.parseInt(string) Double.parseDouble(string) Arrays.sort(type[] list) Arrays.binarySearch(type[] list, type key)	**Array/Length/Initializer** **int**[] list = **new int**[10]; list.length; **int**[] list = {1, 2, 3, 4}; **Multidimensional Array/Length/Initializer** **int**[][] list = new **int**[10][10]; list.length; list[0].length; **int**[][] list = {{1, 2}, {3, 4}};

1. INTRODUCTION TO JAVA LANGUAGE

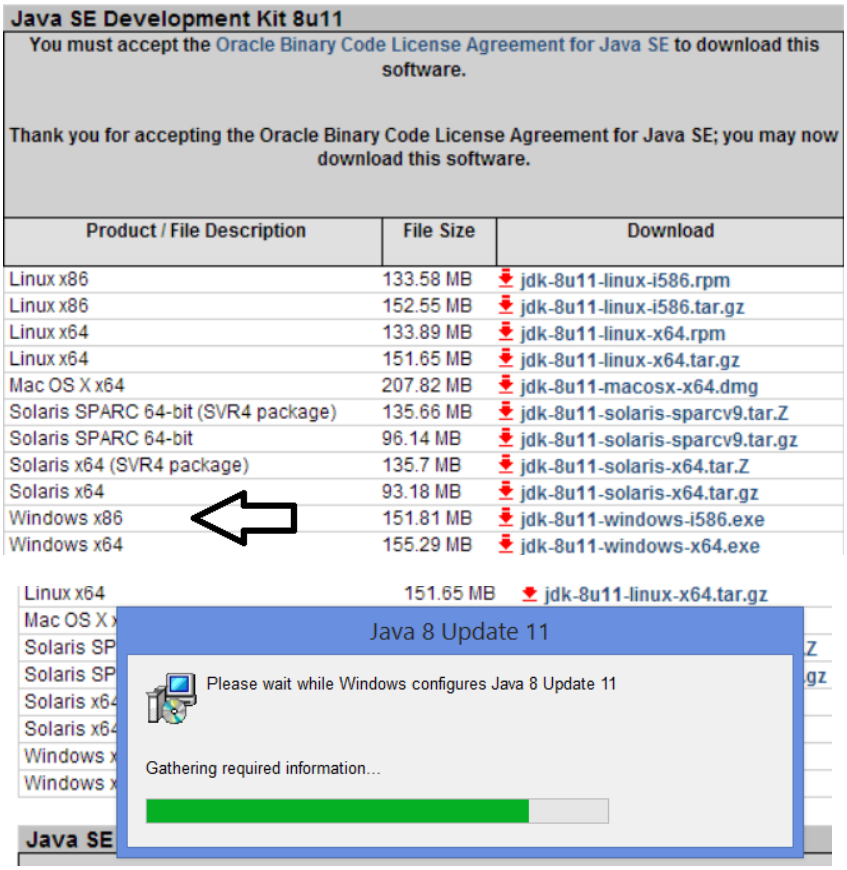
1.1 PREAPARATION TO ESTABLISH JAVA PROGRAMMING ENVIRONMENT IN YOUR COMPUTER

In order to start computer programming with java language you should have java development kid (JDK) in your computer. JDK package is available from sun computer company and you can get it from internet adres http://www.oracle.com/technetwork/java/index.html

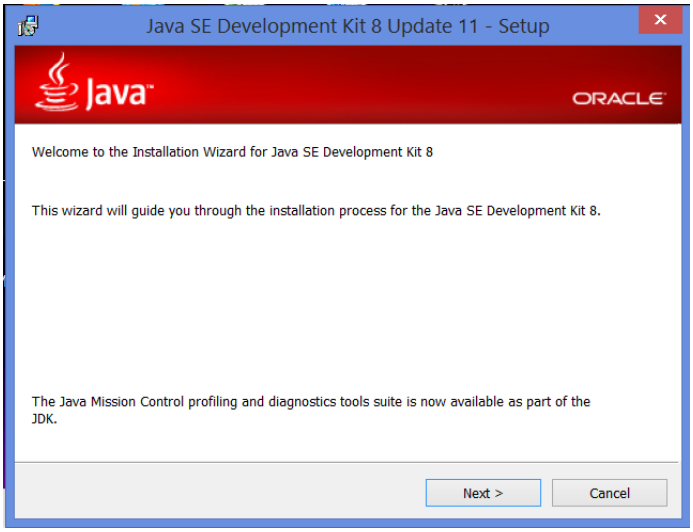


Select Java SE, then select JDK8 Update 11 or most recent JDK package available to load, push Download button

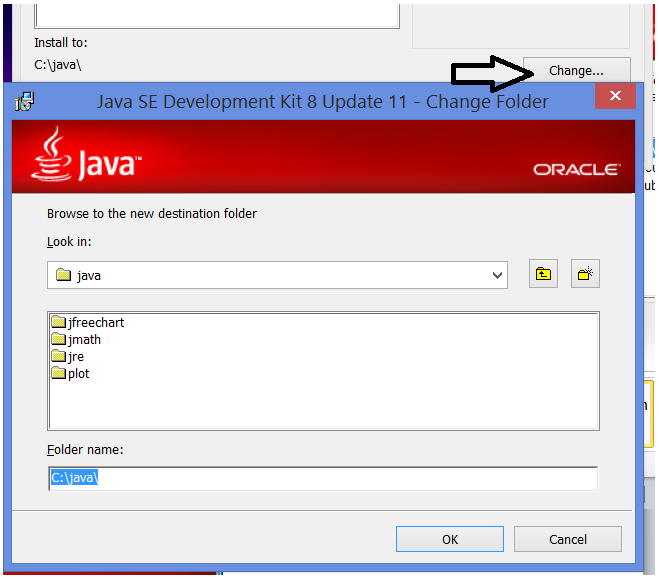
Select Licence agreement and then Select your platform (Windows, Linux etc.) . A note for 64 bit users. Java has huge library and 32 bit version has better completed libraries specially for the ones that would like to connect to external devices, so I am using 32 bit version of JDK



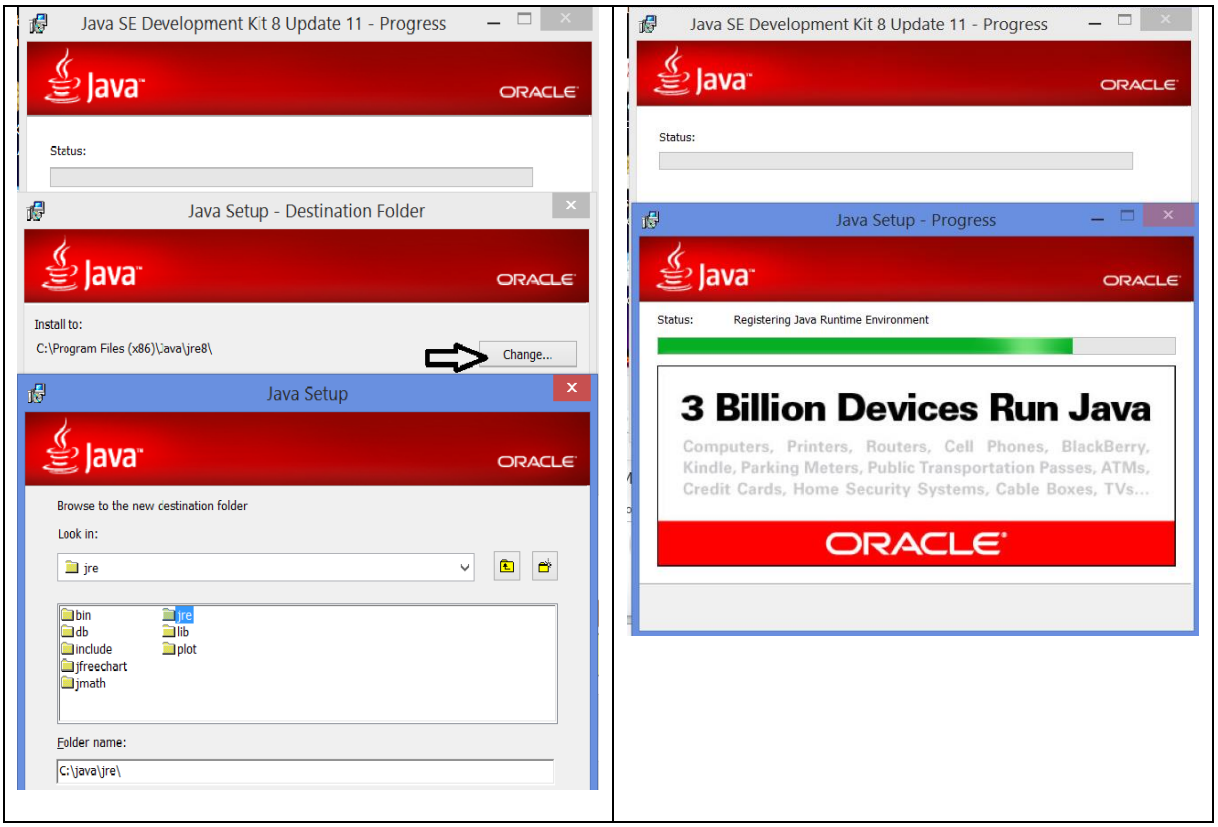
After saving the program we can start it by double click



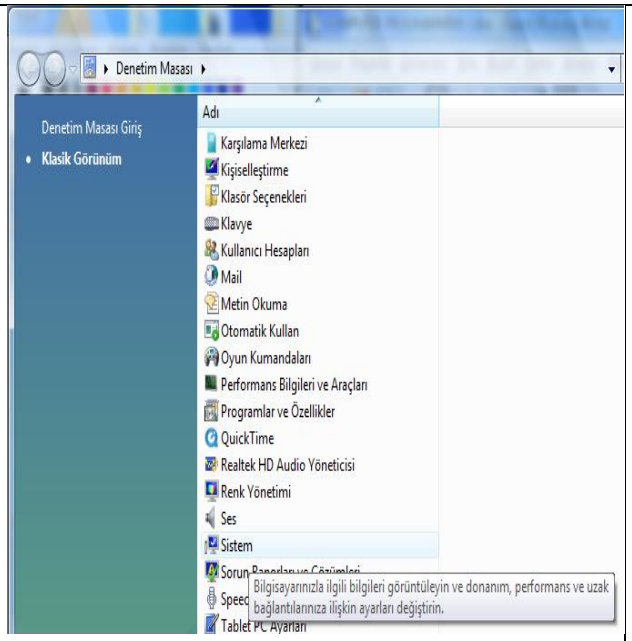
Specify the directory you would like to install java Develpment Kit (JDK), I do recommend a simple directory to easy Access , such as d:\java, Then install it

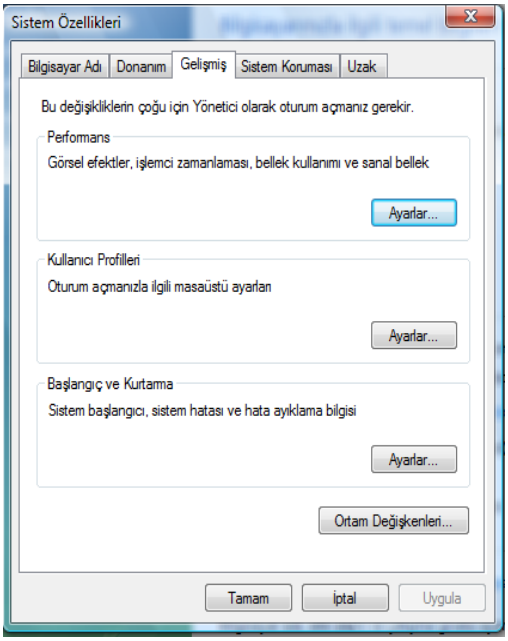


Select the folder you would like to put JDK set. I do recommend d:\java as a simple option. Then program will ask the folder for jre (Java Runtime environment) we can select jre folder under the java folder we already selected (d:\java\jre)

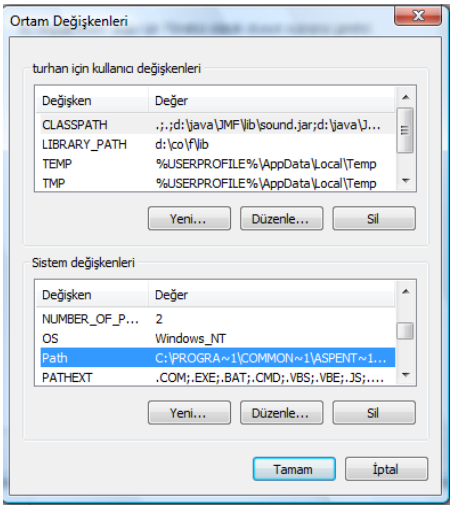


Now java is installed. The next is to add java directories to path statement. When the computer could not able to find a command it will check the directories given in path statement and if it finds it there it will use it. In order to to do this, from control panel select the system folder

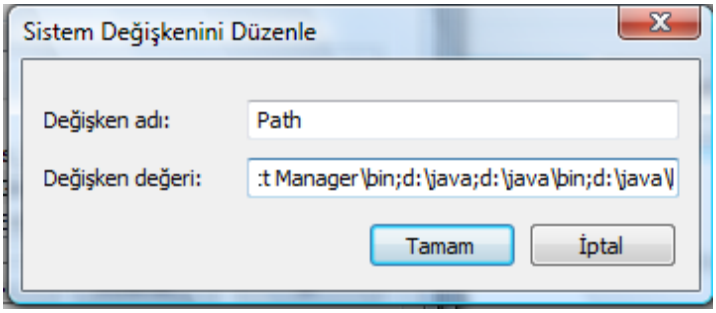




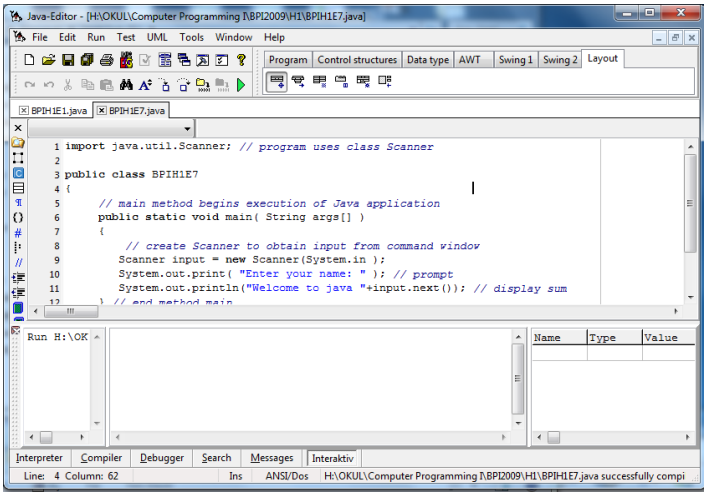
Select environmental variables and find path



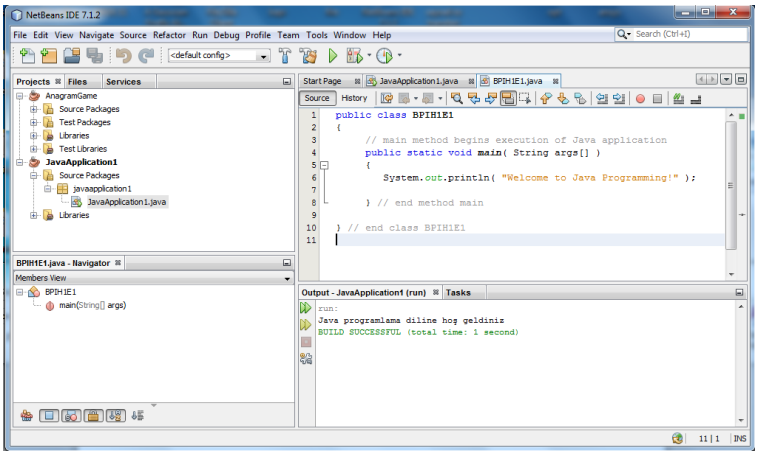
select Path statement and add bin and lib directories path in out case d:\java\bin and d:\java\lib.



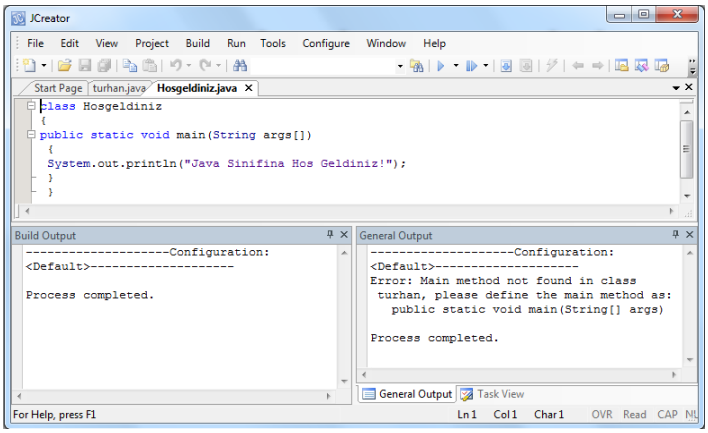
Next, I recommend you to use an editor to edit and compile java programs. There are several freeware good editors are available to use togetger wiyh java. Let us look at some of these editors. Everyone java Editor(http://sourceforge.net/projects/eje/) is a simple java editör. Java-editor (http://javaeditor.org/index.php/Download) is another simple java editör created to use directly with java.



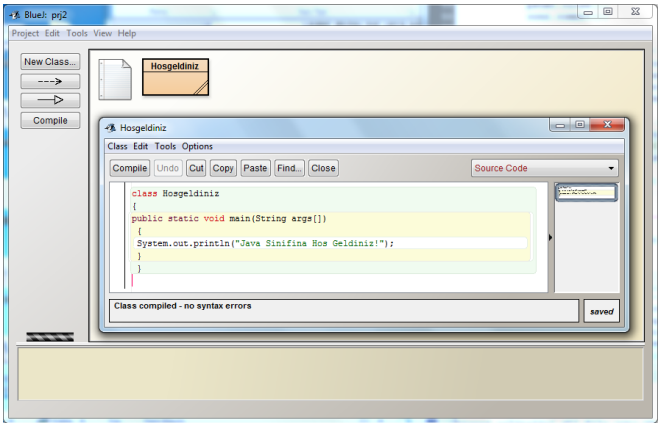
Netbeans is an advanced java editör (http://netbeans.org/downloads/index.html)



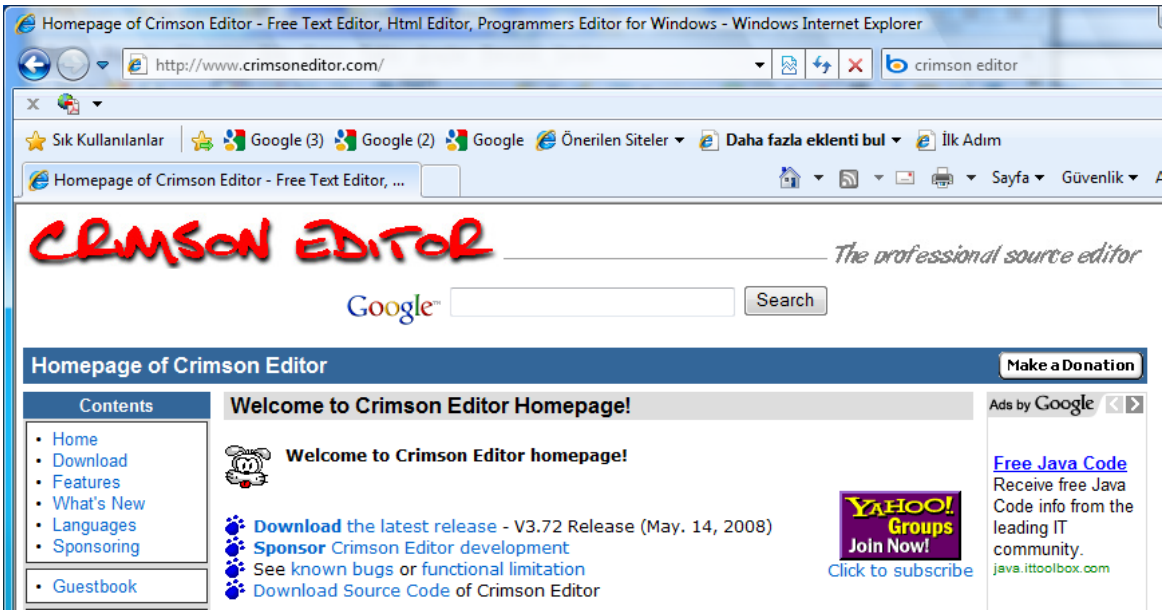
Another simple nice freware java editör is JCreator. This editör is also designed to be used only with java language. It can be downloaded from the adress : http://jcreator.en.softonic.com/



A recent editor that easy to use with java is BlueJThis editör can be downloaded from http://www.bluej.org/ adress

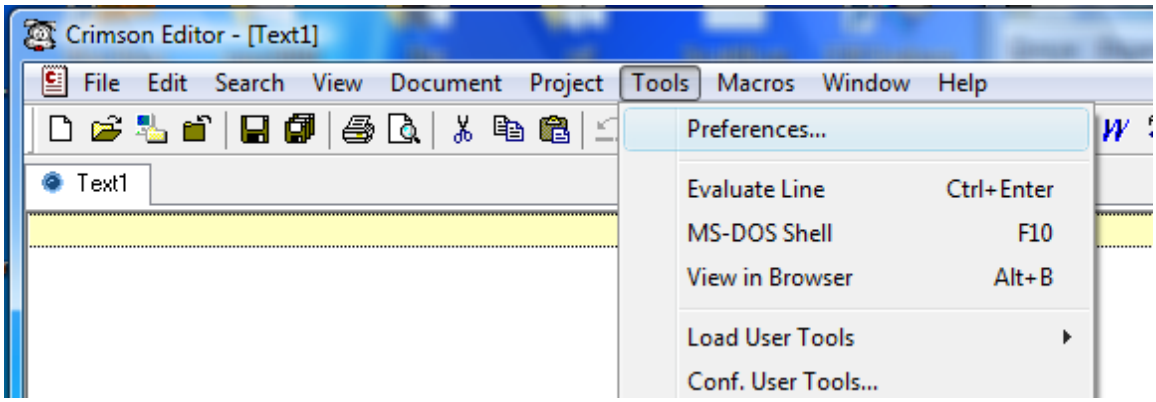


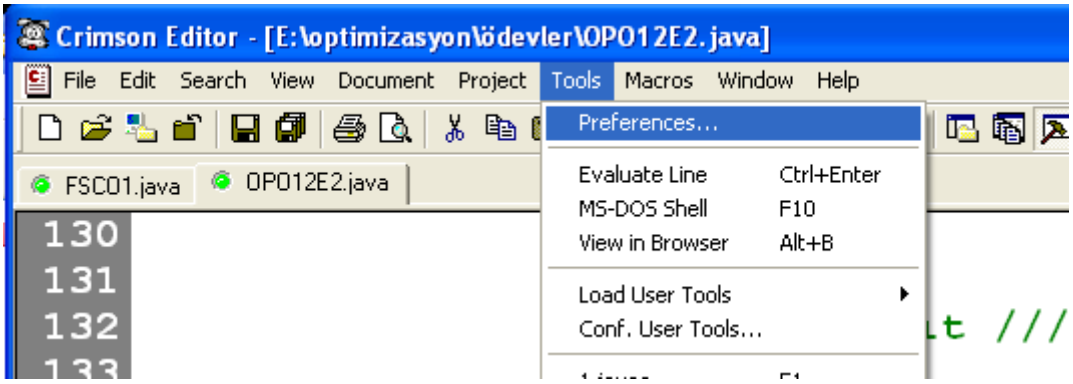
I do recommend a simpler editor called crimson editor for starting programmers. This is also a free program, you can obtain a copy from internet adress http://www.crimsoneditor.com/ .This is a multilanguage editor, which java compiling can be established as well as other programming languages.



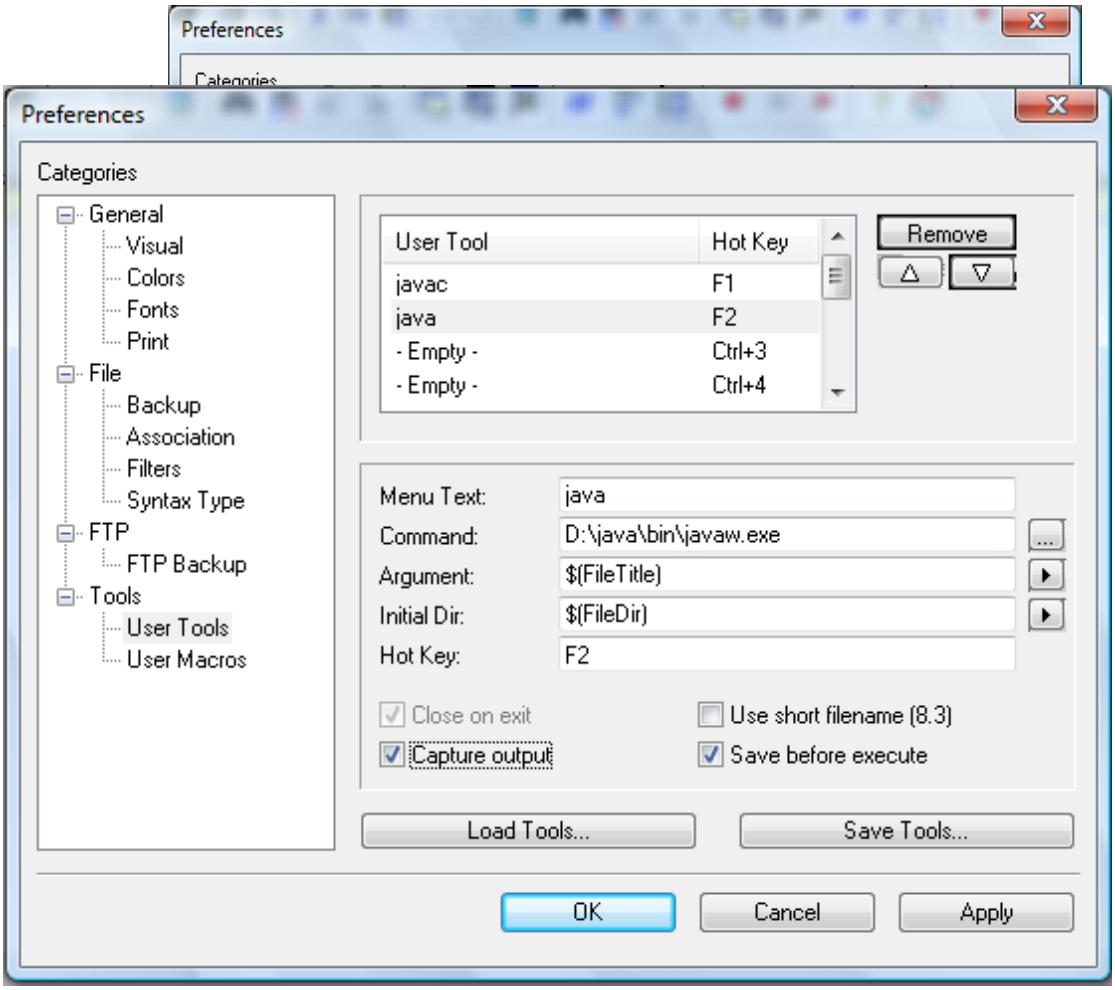
The latest version of Crimson editor when ı looked at was Version 3.72 the file it loads named as : cedt-241-setup.exe. By double click you can load it up to your computer.

In order to compile and run java programs in crimson editor select Tools, preferences, User tools. Cedit programında java dosyanızı çalıştırmak istiyorsanız: Tools, preferences, User Tools düğmelerine basarak javac ve java komutlarını editöre aşağıdaki pencerelerde görüldüğü gibi girebiliriz. (burada java d:\java dosya sistemine yüklenmiştir.)

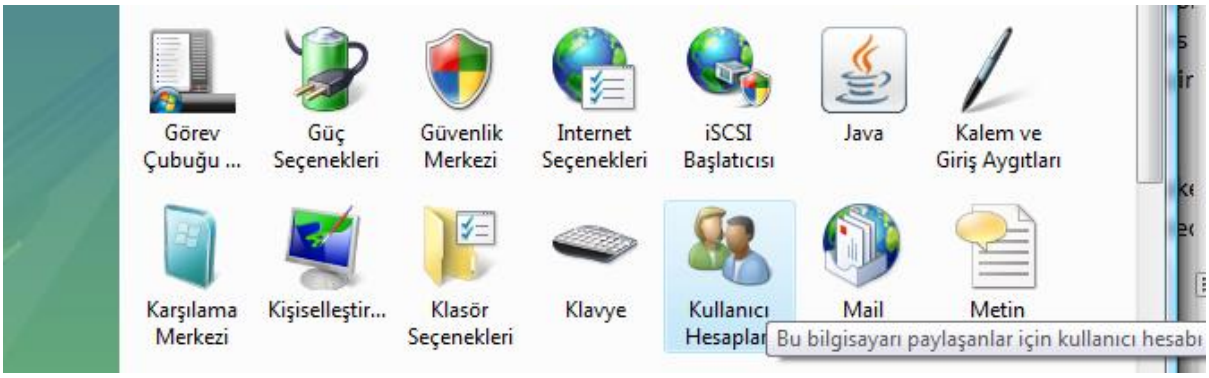


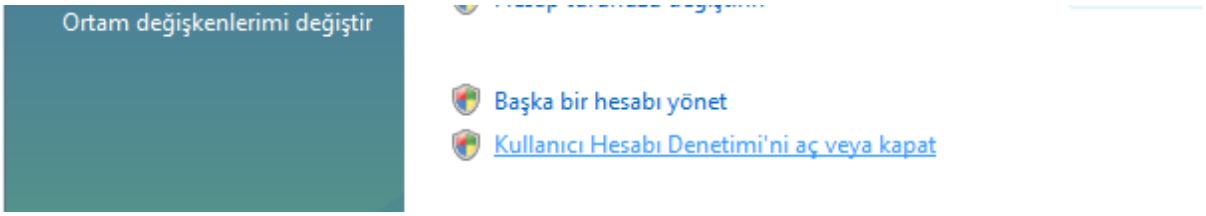


then enter java compile statement javac

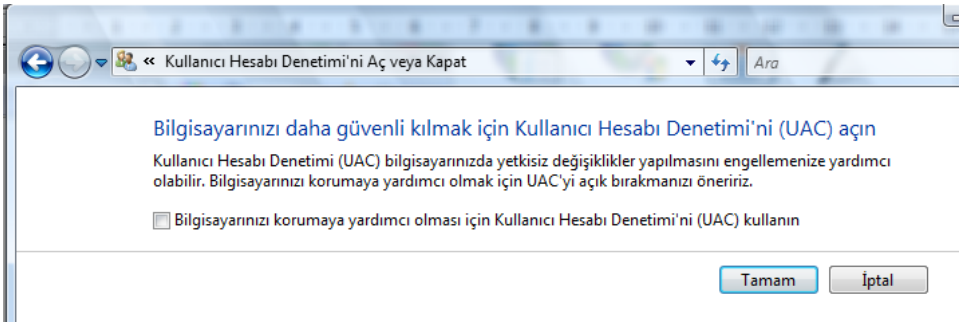


Select javac statement under bin directory, select FileName as argument and Select File Directory as Initial directory. Select F1 as hot key, select Capture output button. In order to enter java runtime enterpretation statement java enter java as Menu Text, As command enter javaw or java command under java compilers bin directory. Select FileTitle as Argument, select FileDir as initial directory, and select F2 as hot key. After doing this you can able to work with java programs, compile it by pressing F1 and run them by pressing F2. One note for MS Vista users, In order to protect you from nasty programs vista does not allow you to make changes in your editor settings, iIf you want this settings remains in its place, go to control panel and select user accounts, then open and closed the user account button





and turn off the user acount security control button.



In addition , I do use important public access java packages such as visad (3 D graphic package) . If you like to use it, obtain a copy of visad.jar from the adress http://www.java2s.com/Code/Java/Advanced-Graphics/GraphLibraryVisad.htm, the load up java 3D package and as well from the adress http://java3d.java.net/binary-builds.html, and load java3D and copy visad.jar into ../jre/lib/ext directory.

Excell spread sheet read and write package xls.jar from the adress http://jexcelapi.sourceforge.net/ and again copy jxl.jar into the ../jre/lib/ext directory

If you like to use SQLite database with java you can get necassary connection file sqlitejdbc-v056.jar from http://www.zentus.com/sqlitejdbc/ and again load it into the ../jre/lib/ext directory

1.2 INTRODUCTION TO JAVA PROGRAMING LANGUAGE

Now we can start to write our first program. You can use your crimson editor or any other environment to write your program. We will assume that crimson editor is used. Our first example is a welcome message. It is traditional in programming language teaching to give a welcome message as the first application

Program 1.2.1 BPIH1E1.java programı

```
public class BPIH1E1
{
```

```
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      System.out.println( "Welcome to Java Programming!" );
    } // end method main

} // end class BPIH1E1
```

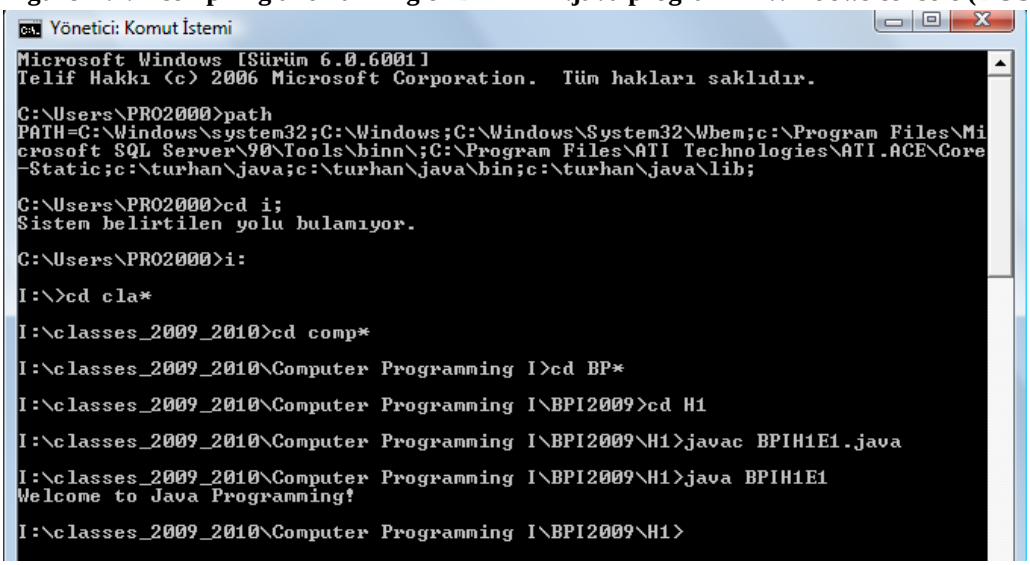
Let us assume that we write our program in any editor and save it. All Java programs has an external envelope called class. The name of the class should be same as the name of the program file. The class name fort his first example is given as BPIH1E1, therefore we will save the file as BPIH1E1.java

In this stage, we will save it in the same directory as we are working. In order to compile the program**, javac** statement is used. It can be carry out in console environment (dos for window system). After compilation, program create a class file, in this example a file called **BPIH1E1.class**. This file contains a machine language-like code that is easily convertible to machine language code for the operation systems and processor of actual computer program is running. This conversion process is done through an interpreter specific to the computer system program is running. In order to operate the interpretor (convertion the class bycode to the machine language code **java** command is used. To run the above program in console environment will require then rhe following commands;

>javac BPIH1E1.java
>java BPIH1E1

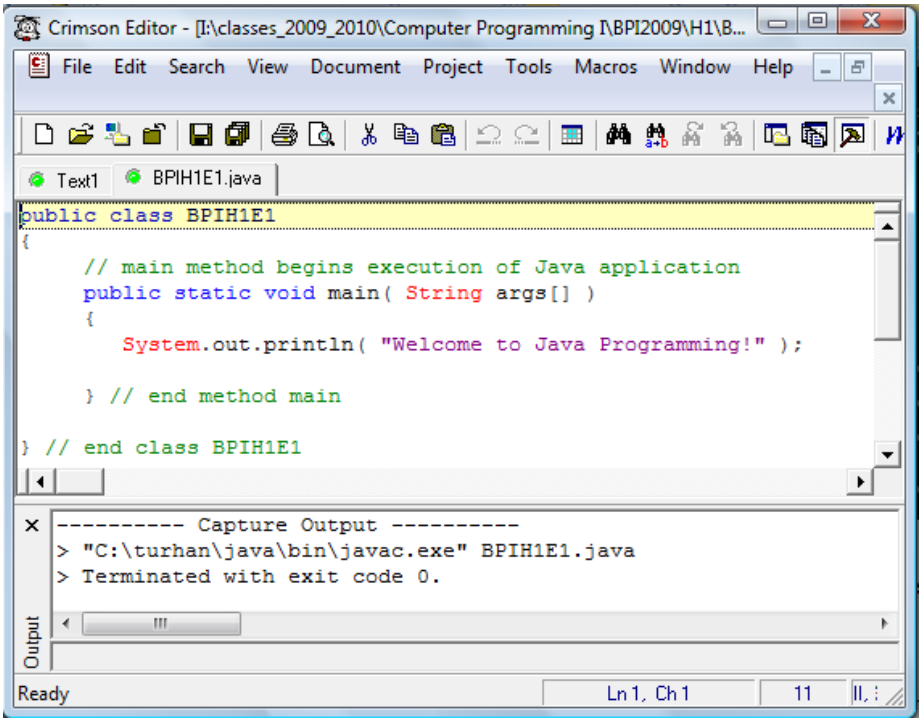
In figure 1.2.1 compiling and running of **BPIH1E1.java** program by using javac and java commands are shown

Figure 1.2.1 compiling and running of BPIH1E1.java program in Windows console (DOS) environment



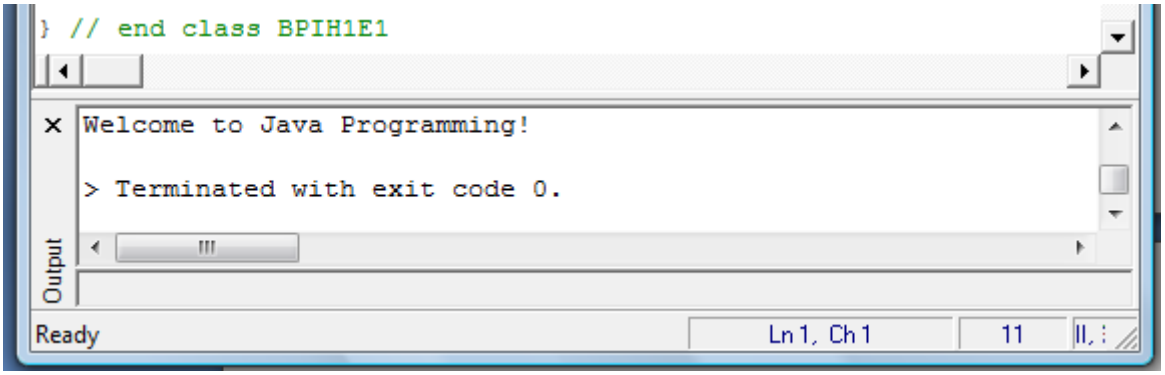
If we would like to do the same by using crimpson editor, all we have to do is to use pre-programmed F1 and F2 buttons . When F1 button pushed (javac compiler command)

Figure 1.2.2 compiling of BPIH1E1.java program in Windows console (DOS) environment by using preprogrammed F1 button in Crimson Editor



Program compiles and **BPIH1E1.class** is created. When F2 button pushed, **BPIH1E1.class** file interpreted the machine language and program runs in the computer. Because of this dual system java codes can run free of the operating systems or computer, the samer code (java class code) will run in any computer.

Figure 1.2.3 interpreting (running) of BPIH1E1.class bycode in Windows console (DOS) environment by using preprogrammed F2 button in Crimson Editor



Now let us look at the details of our first program, The first line

public class BPIH1E1 is given us basic class name, term public specify that the code of the program will be public (will not be hidden). Classes started with big open paranthesis, {, and ends with big close paranthesis, }. This is also valid for other java statements as well. Big paranthesis are the same as paragraph start and finish signs.

The line **// main method begins execution of Java application** is a command statement, computer does not do anything related to this line, it is for us to understand the program code.

The line **public static void main(String args[])** start a subprogram part called a method. The name of our method is **main,** the word **void** tells us that this method is not given us any output from this program part. **System.out.println("Welcome to Java Programming!");**

print to console(DOS) screen **Welcome to Java Programming!** The sign **“** indicates that it is a writing (String) type of constant. The sign **;** indicates the end end of the programming command **System.out.println** is another method defined in java library **java.io.** With closed big paranthesis main method closed, and class is closed with another big paranthesis.

Second example program , **BPIH1E2.java,** will do the same as the first program, but in this one two output commands are used.

System.out.print("Welcome to ");
System.out.println("Java Programming!");

The difference between System.out.print and System.out.prinln is that the second one includes a carriage return, while the first one stops in the same line, therefore the output of this program is exactly same as the previous one.

Program 1.2.2 BPIH1E2.java programı

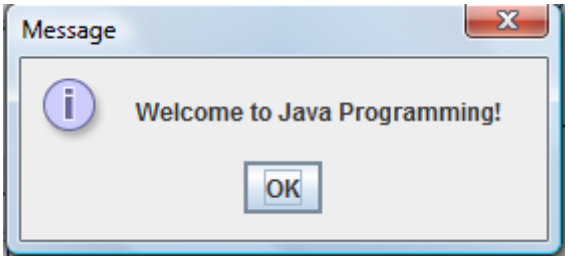
```
import javax.swing.*;
public class BPIH1E2
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
    System.out.print("Welcome to ");
    System.out.println("Java Programming!" );
    } // end method main

} // end class BPIH1E2
```

In the third code, in order to take an output, the command
JOptionPane.showMessageDialog(null,"Welcome to Java Programming!");
Is used. This program uses library method **JOptionPane.showMessageDialog.** This method is defined in java library **javax.swing.** The first line of the program indicates that we are connected to the codes in this library * indicates that all codes under this library is called. The output of this methos is given as a graphic output instead of console text output.

Program 1.2.3 BPIH1E3.java programı

```
import javax.swing.*;// program uses class JOptionPane in swing library
public class BPIH1E3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
    JOptionPane.showMessageDialog(null,"Welcome to Java Programming!" );
    } // end method main

} // end class BPIH1E3
```

Figure 1.2.4 interpreting (running) of BPIH1E3.class bycode in graphic environment



A similar program to BPIH1E1.java is also given as **BPIH1E4.java.** In this program, instead of System.out.println("Welcome to Java Programming!");
System.out.printf("%s\n%s\n","Welcome to", "Java Programming!"); is used. The result is exactly the same. This output structure is taken exactly from C programming language. Java is very similar to c and C++ as command structure. In the latest version of java also some of the same method names are alsıo introduced. Some control terms are used in this statement. Some of the very common control statements in java as follows

\n ctrl-return (next line)

\t ctrl-tab (next tab space)

\r enter key

\\ write \ sign

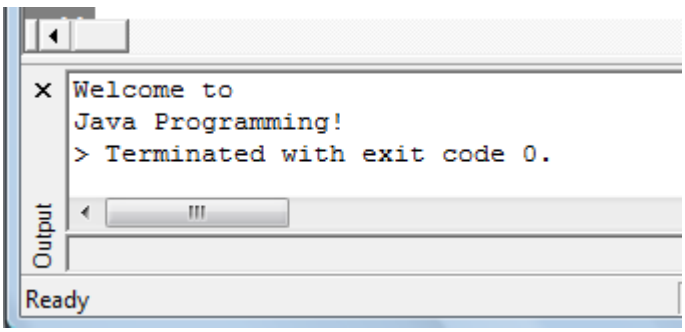
\" write " sign

There are also some printf format structures in printf statement %s means write a String (array of characters) constant or variable.

Program 1.2.4 BPIH1E4.java programı

```
public class BPIH1E4
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
    System.out.printf( "%s\n%s\n","Welcome to", "Java Programming!" );

    } // end method main
} // end class BPIH1E4
```

The program output is



The carriage return (/n) in the middle of the formatting part of printf force the output to the next line

1.3 INTEGER AND REAL VARIABLES AND ARITHMETIC CALCULATIONS

Our next program **BPIH1E5.java** reads two integer numbers from console screen, and then adds this numbers, and output the result to the console screen. In order to read data from console environment, class **java.util.Scanner** is used. A channel to the console opens by using method Scanner. The name of the input channel (Scanner type variable) input. Scanner is basically another class structure (program) defined in java. Actual reading is done throgh a method **input.nextInt();** of class Scanner (located in the library java.util.Scanner)

Program 1.3.1 BPIH1E5.java programı

```
import java.util.Scanner; // program uses class Scanner

public class BPIH1E5
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
      int number2; // second number to add
      int sum; // sum of number1 and number2
      System.out.print( "Enter first int: " ); // prompt
      number1 = input.nextInt(); // read first number from user
      System.out.print( "Enter second integer: " ); // prompt
      number2 = input.nextInt(); // read second number from user
      sum = number1 + number2; // add numbers
      System.out.printf( "Sum is %d\n", sum ); // display sum
    } // end method main

} // end class BPIH1E5
```

Integer variables can be called as byte(8 bit), short(16 bit), int(32 bit) and long(64 bit) according to space they occupy in computer memory. Bits are binary structures that only contains 0 or 1. When bits coming together as byte groups of different sizes they represents integer numbers. For example maximum number can be written to 8 bit byte type of integer variable is 11111111 which represents 256. Usually one of the bits used as +/- sign in a number therefore byte carries numbers between -128 to 127 (total 256 including 0) . ın order to define an integer variable, first variable type defined and a variable name follows. An actual value can be assigned by using an = (equal) sign.

int number1;
number1=32;

or this two statement can be given only as one statement.

int number1=32;

In our program integer value is read from the console as

int number1;

number1 = input.nextInt();

Numbers represents with bits in computer memory. In table 1.3.1 different type of integer variables and their number representation limits are given.

Figure 1.3.1 The bit sizes and 10 based limits of the integer type variables.

Integer type	**Byte size**	**Number represantation limits**
Byte	8 bit	From -128 to 127
Short	16 bit	From -32768 to 32767
int	32 bit	From -2157483648 to 2147483647
long	64 bit	From -9223372036854775808 to 9223372036854775808

In Table 1.3.2 representation of number 15 in byte type integer variable in computer bit structure.

Table 1.3.2 writing 15 in a 8 bit integer variable (byte)

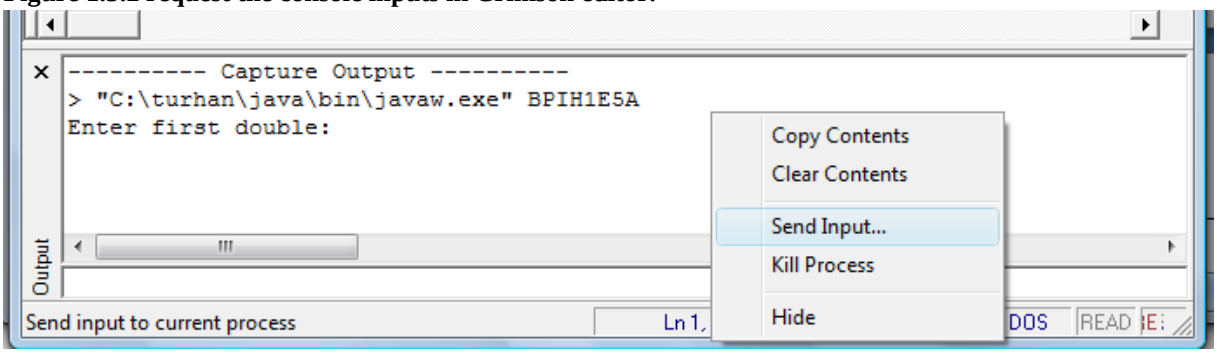
+/-	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+	**64**	**32**	**16**	**8**	**4**	**2**	**1**
0	**0**	**0**	**0**	**1**	**1**	**1**	**1**

The following arithmetic proseses are defined in java for integer variables:
+ addition - substraction * multiplication / division and % module (remaining number) in integer division. For example an addition process can be defined as

sum = number1 + number2;

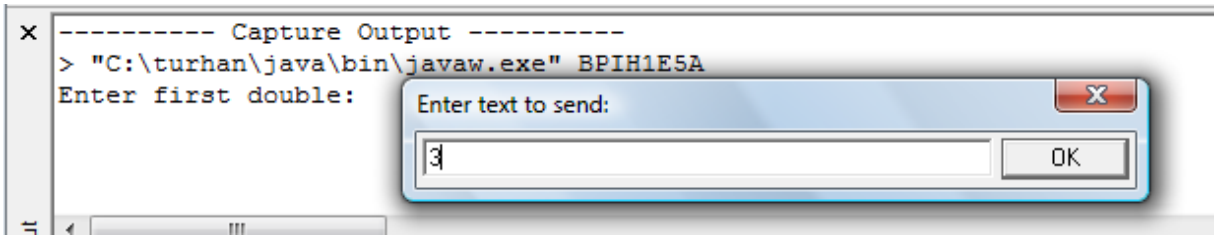
To get output of the example program in console environment by using Crimson editor, first we should send inputs through pushing send Input buton

Figure 1.3.2 request the console inputs in Crimson editor.



Then you can enter the number through the opening boxes

Figure 1.3.3 sending the console inputs in Crimson editor.



Then you can enter the second number the same way and result will apear in the capture output window as

Figure 1.3.4 Crimson editor output of BPIH1E5.java

```
> "C:\turhan\java\bin\javaw.exe" BPIH1E5
Enter first int: 3
Enter second integer: 5
Sum is 8

> Terminated with exit code 0.
```

If we would like to do the same thing directly in dos windos we can enter the numbers directly

Figure 1.3.5 Console(DOS) output of BPIH1E5.java

```
I:\classes_2009_2010\Computer Programming I\BPI2009\H1>java BPIH1E5
Enter first int: 3
Enter second integer: 5
Sum is 8

I:\classes_2009_2010\Computer Programming I\BPI2009\H1>
```

Instead of integer arithmetic, if real number arithmetic is required, float(32 bit) or double(64 bit) real variables can be used. In order to write in computer memory of binary bit structure a real number, sum digits are assumed to carry negative powers of 2. For example to write **22.625e-17** into computer memory, following bit/byte structure can be assumed.

Table 1.3.3 writing 22.625×10^{-17} in a 16 bit real variable

+/-	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	+\-	2^4	2^3	2^2	2^1	2^0
+	16	8	4	2	1	0.5	0.25	0.125	0.0625	-	16	8	4	2	1
0	1	0	1	1	0	1	0	1	0	1	1	0	0	0	1

Number representaion limits of real variables are given in Table 1.3.4.

Table 1.3.4 Real variable types, bit sizes and number representation limits

Real variable type	Bit size	Number represantation limits
float	32	From -3.40292347e+38 to 3.40292347e+38
double	64	From -1.7976931348623157e+308 to 1.7976931348623157e+308 e

Now, a program similar to **BPIH1E5A.java and BPIH1E5B.java** with double variable can be investigated.

Program 1.3.2 BPIH1E5A.java program

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH1E5A
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
```

```
        Locale us=new Locale("US");
        input.useLocale(us);
        double number1; // first number to add
        double number2; // second number to add
        double sum; // sum of number1 and number2
        System.out.print( "Enter first double: " ); // prompt
        number1 = input.nextDouble(); // read first number from user
        System.out.print( "Enter second integer: " ); // prompt
        number2 = input.nextDouble(); // read second number from user
        sum = number1 + number2; // add numbers
        System.out.printf(us,"Sum is %10.5e\n", sum ); // display sum
    } // end method main

} // end class BPIH1E5A
```

Program 1.3.3 BPIH1E5B.java program

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH1E5B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        // create Scanner to obtain input from command window
        Scanner input = new Scanner( System.in );
        Locale tr=new Locale("TR");
        input.useLocale(tr);
        double number1; // first number to add
        double number2; // second number to add
        double sum; // sum of number1 and number2
        System.out.print( "Enter first double: " ); // prompt
        number1 = input.nextDouble(); // read first number from user
        System.out.print( "Enter second integer: " ); // prompt
        number2 = input.nextDouble(); // read second number from user
        sum = number1 + number2; // add numbers
        System.out.printf(tr,"Sum is %10.5e\n", sum ); // display sum
    } // end method main

} // end class BPIH1E5B
```

The basic difference of this programs compare to the previous integer version is the
Locale us=new Locale("US");
Statement, with this statement and the with the following
input.useLocale(us);
statement you can define the country program is running and that specification of that countries number divisor. For example US use period (.) as number divisor, but in Turkey comma(,) is used as number divisor.

In program **BPIH1E5C.java** formatting with **String s=String.format(us,"Sum is %10.5f\n", sum);** term is used to get the formatted output. Please not that %d for integer format, %10.5e and %10.5f for double format is used.

Program 1.3.4 BPIH1E5C.java program

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH1E5C
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      Locale us=new Locale("US");
      input.useLocale(us);
      double number1; // first number to add
      double number2; // second number to add
      double sum; // sum of number1 and number2
      System.out.print( "Enter first double: " ); // prompt
      number1 = input.nextDouble(); // read first number from user
      System.out.print( "Enter second integer: " ); // prompt
      number2 = input.nextDouble(); // read second number from user
      sum = number1 + number2; // add numbers
      String s=String.format(us,"Sum is %10.5f\n", sum );
      System.out.println(s); // display sum
    } // end method main

} // end class BPIH1E5C
```

In the next program graphic type String (character array variable) defined a file name and data is read by using Scanner channel from a data file

String filename=JOptionPane.showInputDialog("Enter file name (a.txt): ");

Program 1.3.5 BPIH1E5D.java program

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;  // international format library Locale
import javax.swing.*;     //Java swing graphic library
import java.io.*;         //Java input library for File class

public class BPIH1E5D
{
    // main method begins execution of Java application
    public static void main( String args[] ) throws FileNotFoundException
    {
       // create Scanner to obtain input from a file
      String filename=JOptionPane.showInputDialog("Enter file name (a.txt): ");
      Scanner input = new Scanner(new File(filename));
      Locale us=new Locale("US");
      input.useLocale(us);
      double number1; // first number to add
      double number2; // second number to add
      double sum; // sum of number1 and number2
      number1 = input.nextDouble(); // read first number from user
```

```
        number2 = input.nextDouble(); // read second number from user
        sum = number1 + number2; // add numbers
        String s=String.format(us,"Sum is %10.5f\n", sum );
        System.out.println(s); // display sum
    } // end method main
} // end class BPIH1E5D
```

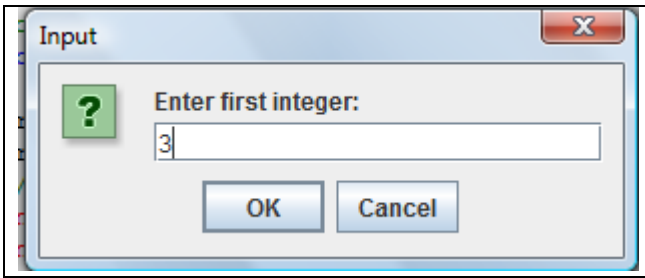
In the last program of first week, two integer numbers are read through the graphic window method through statement

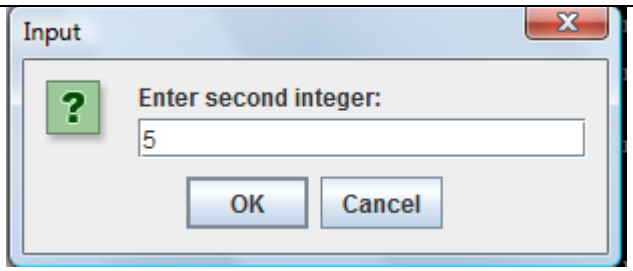
number1=Integer.parseInt(JOptionPane.showInputDialog("Enter first integer: "));

Program 1.3.6 BPIH1E6.java program

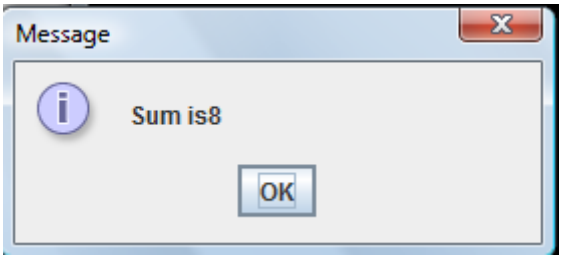
```
import javax.swing.*;// program uses class JOptionPane in swing library

public class BPIH1E6
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        int number1; // first number to add
        int number2; // second number to add
        int sum; // sum of number1 and number2
        number1=Integer.parseInt(JOptionPane.showInputDialog("Enter first integer: "));
        number2=Integer.parseInt(JOptionPane.showInputDialog("Enter second integer: "));
        sum = number1 + number2; // add numbers
        JOptionPane.showMessageDialog(null,"Sum is"+sum ); // display sum
    } // end method main

} // end class BPIH1E4
```

The input Windows of this program will look like this :





The result will be



Not that instead of integer if we would like to read double value, the statement will be

double number1=Double.parseDouble(JOptionPane.showInputDialog("Enter first reqal number: "));

In the next program we will input a name (String variable) by using Scanner class. In order to read the variable **input.next()** statement is used.

Program 1.3.7 BPIH1E7.java program

```
import java.util.Scanner; // program uses class Scanner

public class BPIH1E7
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner(System.in );
      System.out.print( "Enter your name: " ); // prompt
      System.out.println("Welcome to java "+input.next()); // display sum
    } // end method main

} // end class BPIH1E4
```

In the last program String type input variable will be read by using
String isim=JOptionPane.showInputDialog("Enter first integer: ");
statement

```
import javax.swing.*;// program uses class JOptionPane in swing library

public class BPIH1E8
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      String isim=JOptionPane.showInputDialog("Enter first integer: ");
      JOptionPane.showMessageDialog(null,"Welcome to java "+isim ); // display sum
    } // end method main

} // end class BPIH1E4
```

WEEK 1 CLASS EXERSIZES

EX 1. Write a program to print in console screen “hello my name” and your name. For example if your name is ali, it will be **Hello my name is Ali.** The name of the program and class will be **BPIH1CEX1.java**

```
public class BPIH1CEX2
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      System.out.println( "Hello my name is Ali!" );
    } // end method main
} // end class BPIH1E1
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH1CEX2
Hello my name is Ali!
> Terminated with exit code 0.
```

EX 2. Write a program to print in graphic screen “hello my name” and your name. For example if your name is ali, it will be **Hello my name is Ali.** The name of the program and class will be **BPIH1CEX2.java**

```
import javax.swing.*;// program uses class JOptionPane in swing library
public class BPIH1CEX3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
    JOptionPane.showMessageDialog(null,"Hello my name is Ali!" );
    } // end method main
} // end class BPIH1E3
```

EX 3. Write a program to enter your name in console screen and write "hello my name" and your name. For example if your name is ali, it will be **Hello my name is Ali.** The name of the program and class will be **BPIH1CEX3.java**

```
import javax.swing.*;// program uses class JOptionPane in swing library

public class BPIH1CEX3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      String isim=JOptionPane.showInputDialog("Enter your name: ");
      JOptionPane.showMessageDialog(null,"Hello my name is "+isim+"!" );
    } // end method main

} // end class BPIH1E4
```

EX 4. Write a program to enter two integer in console screen and write the division of this two integer. in console screen . The name of the program and class will be **BPIH1CEX4.java**

```
import java.util.Scanner; // program uses class Scanner

public class BPIH1CEX4
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
```

```
        int number2; // second number to add
        int division; // division of number1 and number2
        System.out.print( "Enter first int: " ); // prompt
        number1 = input.nextInt(); // read first number from user
        System.out.print( "Enter second integer: " ); // prompt
        number2 = input.nextInt(); // read second number from user
        division = number1/number2; // add numbers
        System.out.printf( "Division is %d\n", division ); // display division
    } // end method main

} // end class BPIH1CEX4
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH1CEX4
Enter first int: 10
Enter second integer: 3
Division is 3

> Terminated with exit code 0.
```

EX 5. Write a program to enter two double in graphic screen and write the division of this two double variables in double in graphic screen. The name of the program and class will be **BPIH1CEX5.java**

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH1CEX5
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        // create Scanner to obtain input from command window
        Scanner input = new Scanner( System.in );
        Locale us=new Locale("US");
        input.useLocale(us);
        double number1; // first number to add
        double number2; // second number to add
        double division; // division of number1 and number2
        System.out.print( "Enter first double: " ); // prompt
        number1 = input.nextDouble(); // read first number from user
        System.out.print( "Enter second integer: " ); // prompt
        number2 = input.nextDouble(); // read second number from user
        division = number1/number2; // add numbers
        System.out.printf(us,"division is %10.5e\n", division ); // display division
    } // end method main

} // end class BPIH1E4
```

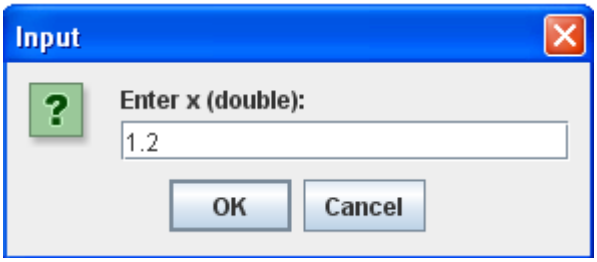
```
---------- Capture Output ----------

> "D:\java\bin\javaw.exe" BPIH1CEX5
Enter first double: 10.2
Enter second integer: 3.0
division is 3.40000e+00
```

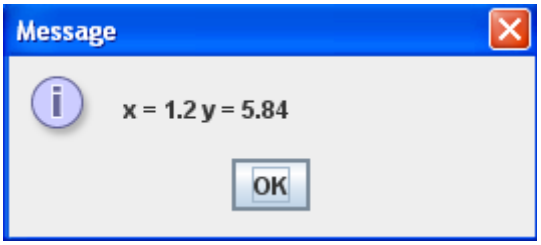
```
> Terminated with exit code 0.
```

WEEK 1 HOMEWORKS

HW 1 Write a program to enter a double variable named x in graphic screen and write the results of y=x*x+2*x+2 in graphic screen. The name of the program and class will be **BPIH1HW1.java**

```
import javax.swing.*;// program uses class JOptionPane in swing library

public class BPIH1HW1
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      double x;
      double y;
      x=Double.parseDouble(JOptionPane.showInputDialog("Enter x (double): "));
      y=x*x+2*x+2;
      JOptionPane.showMessageDialog(null,"x = "+x+" y = "+y ); // display sx and y
    } // end method main

} // end class BPIH1HW1
```





HW 2 Write a program to enter an integer variable named x in console screen and write the results of y=x*x*x+2/x+1 in console screen. The name of the program and class will be **BPIH1HW2.java**

```
import java.util.Scanner; // program uses class Scanner

public class BPIH1HW2
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int x;
      int y;
      System.out.print( "x (integer) =  " ); // prompt
```

```
        x = input.nextInt(); // read x from user
        y=x*x*x+2/x+1;
        System.out.printf( "x = %d y = %d\n", x,y ); // display x and y
    } // end method main

} // end class BPIH1HW2
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH1HW2
x (integer) =  3
x = 3 y = 28
> Terminated with exit code 0.
```

HW3 Write a program that will write in console screen the following text including the carriage returns(\n)

"Hello my name is (Your name). I am taking computer class the first time"

"It is fun to study a computer language, specially Java"

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH1CEX2
Hello my name is Turhan.  I am taking computer class the first time
It is fun to study a computer language, specially Java

> Terminated with exit code 0.
```

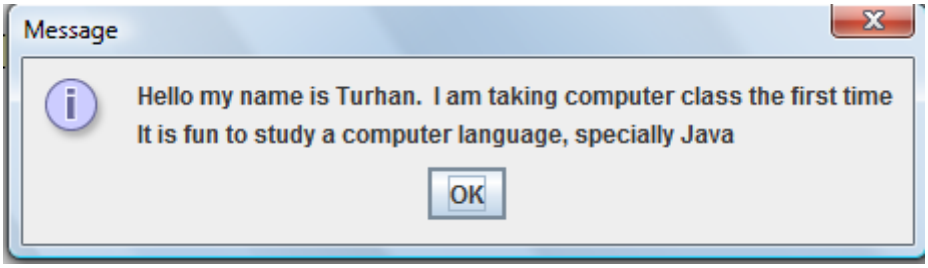
```
public class BPIH1HW3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      System.out.println( "Hello my name is Turhan. I am taking computer class the first tim\nIt is fun to study a computer
language, specially Java" );

    } // end method main

} // end class BPIH1HW3
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH1HW3
Hello my name is Turhan. I am taking computer class the first time
It is fun to study a computer language, specially Java

> Terminated with exit code 0.
```

HW4 Write a program that will write in graphic screen the following text including the carriage returns
"Hello my name is (Your name). I am taking computer class the first time"
"It is fun to study a computer language, specially Java"

```
import javax.swing.JOptionPane;
public class BPIH1HW4
```

```
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      JOptionPane.showMessageDialog( null,"Hello my name is Turhan. I am taking computer class the first time\nIt is fun to study a computer language, specially Java" );

    } // end method main

} // end class BPIH1HW4
```



2. INTRODUCTION TO JAVA LANGUAGE

2.1 STRİNG VARIABLES

In this section we will continue to learn integer and doubles. We will also look at a new kind of variable called String type String type variables can be assign as:

String s=”Ali”;
or
String s=new String(“Ali”);
Strings can be added up by using + sign; for example
String s=”Ali”;
s=s+” Veli” will give us string value
“Ali Veli”

In java arithmetic proceses can be written in shorthand form as well (A notation from c language)
for example instead of **int c=c+1; int c++** can be written and Instead of int **c=c+5; c+=5;** can be written, If we give long and shorthanded form of this statements :

Long notation	**Short notation**
c=c+1;	c++;
c=c-1;	c--;
c=c+a;	c+=a;
c=c-a;	c-=a;
c=c/a;	c/=a;
c=c*a;	c*=a;
c=c%a;	c%=a;

The same short notation can be used in String type of variables as well. Instead of
String s=”Ali”;
the same statement can be written in shorthand form
String s=”Ali”;
s+=” Veli”;

The next example program will be summation of two integer values. In this program, despite reading the data from screen, they are assigned directly in the program. “\n” String format is utilised to advance lines. Note that integer or double values can be added to the String value as well. The results will be String. For example

int number=2;

String s=”number =”+number;

will give the String value:

“number = 2”

Remaining strings can directly written by an output method.

Program 2.1.1 BPIH2E1.java program

```
import java.io.*; //calls java input output class io
class BPIH2E1
 {   // main method begins execution of Java application
     public static void main(String args[])
     {
        int number1,number2;
        int sum,substract,multiply,divide,remain;
        number1=2;
        number2=3;
        sum = number1+number2;
        substract = number1-number2;
        multiply = number1*number2;
        divide=number1/number2;
        remain=number1%number2;
        String s=number1+" +  "+number2+" = "+sum+"\n";
        s+=number1+" -  "+number2+" = "+substract+"\n";
        s+=number1+" x  "+number2+" = "+multiply+"\n";
        s+=number1+" /  "+number2+" = "+divide+"\n";
        s+=number1+" %  "+number2+" = "+remain+"\n";
        System.out.println(s);
      } // end method main
 }
```

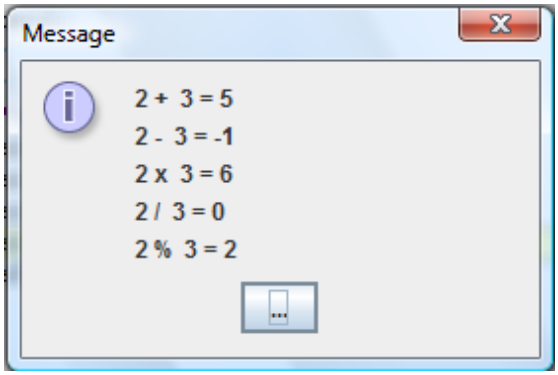
```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH2E1
2 +  3 = 5
2 -  3 = -1
2 x  3 = 6
2 /  3 = 0
2 %  3 = 2

> Terminated with exit code 0.
```

The second example is the same as the first example except output is send to a graphic window

Program 2.1.2 BPIH2E1A.java program

```
import javax.swing.*;// program uses class JOptionPane in swing library

class BPIH2E1A
 {    // main method begins execution of Java application
     public static void main(String args[])
     {
        int number1,number2;
        int sum,substract,multiply,divide,remain;
        number1=2;
        number2=3;
        sum = number1+number2;
```

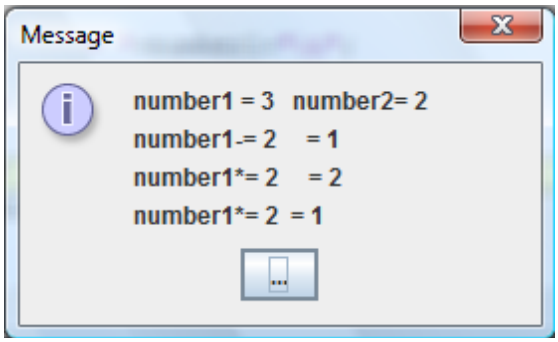
```
        substract = number1-number2;
        multiply = number1*number2;
        divide=number1/number2;
        remain=number1%number2;
        String s=number1+" +  "+number2+" = "+sum+"\n";
        s+=number1+" -  "+number2+" = "+substract+"\n";
        s+=number1+" x  "+number2+" = "+multiply+"\n";
        s+=number1+" /  "+number2+" = "+divide+"\n";
        s+=number1+" %  "+number2+" = "+remain+"\n";
        JOptionPane.showMessageDialog(null,s);
    }// end method main
}
```



In the third example, shorthand notations are also used in integer addition processes

Program 2.1.3 BPIH2E2.java program

```
import javax.swing.JOptionPane;

public class BPIH2E2
{  public static void main(String[] args)
        {
        int number1=1;
        int number2=2;
        number1+=number2;
        String s="number1 = "+number1+"   number2= "+number2+"\n";
        number1-=number2;
        s+="number1-= "+ number2+"     = "+number1+"\n";
        number1*=number2;
        s+="number1*= "+ number2+"     = "+number1+"\n";
        number1/=number2;
        s+="number1*= "+ number2+"  = "+number1+"\n";
        JOptionPane.showMessageDialog(null,s);
        }
}
```



The fourth example is similar the third one except, for printout printf console output is used.

Program 2.1.4 BPIH2E2B.java program

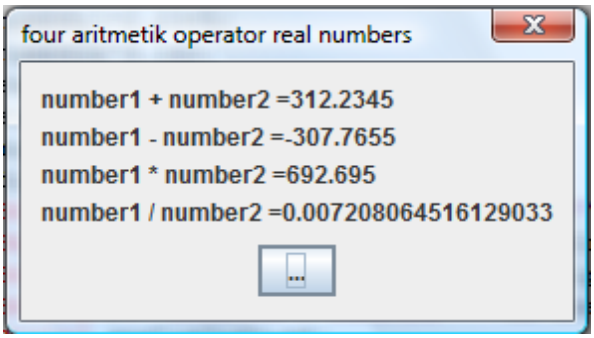
```
public class BPIH2E2B
{  public static void main(String[] args)
        {
        int number1=1;
        int number2=2;
        number1+=number2;
        String s="number1+= : "+ number1+"\n";
        number1-=number2;
        s+="number1-= : "+ number1+"\n";
        number1*=number2;
        s+="number1*= : "+ number1+"\n";
        number1/=number2;
        s+="number1/= : "+ number1;
        //\n newline position the screen curser
        //   at the beginning of the new line
        //\t Horizontal tab, Move the screen cursor
        //   to the next tab stop
        //\r Carriage return. Position the screen cursor
        //   at the beginning of the current line
        //%s String format
        //%d int format
        System.out.printf("%s",s);
        }
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH2E2B
number1+= : 3
number1-= : 1
number1*= : 2
number1/= : 1
> Terminated with exit code 0.
```

A similar program is given by using a double type real numbers

Program 2.1.5 BPIH2E3.java program

```
import javax.swing.*;
class BPIH2E3
 {
     public static void main(String args[])
     {
        double number1,number2;
        double sum,substract,multiply,divide;
        number1=2.2345;
        number2=3.1e2;
        sum = number1+number2;
        substract = number1-number2;
        multiply = number1*number2;
        divide=number1/number2;
        String s1="number1 + number2 ="+sum+"\n";
        String s2="number1 - number2 ="+substract+"\n";
        String s3="number1 * number2 ="+multiply+"\n";
        String s4="number1 / number2 ="+divide+"\n";
        String s=s1+s2+s3+s4;
        JOptionPane.showMessageDialog(null,s,"four aritmetik operator real numbers",JOptionPane.PLAIN_MESSAGE);
      }
 }
```

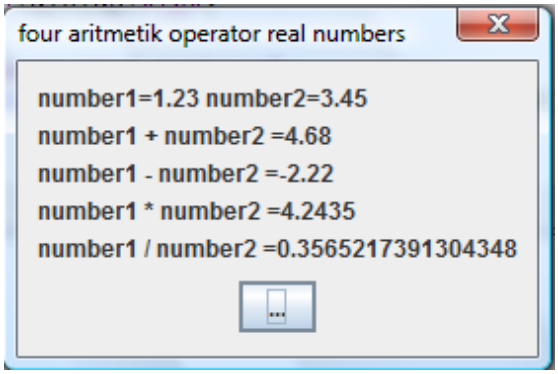


In this last program, A heading for graphic window is also given as : "four aritmetik operator real numbers"

In the next example two real numbers read from graphic screen as real numbers and loaded to different String statements and at the end String statements are added into a new String.

Program 2.1.6 BPIH2E3A.java program

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;
class BPIH2E3A
 {
    public static void main(String args[])
    {
       double number1,number2;
       double sum,substract,multiply,divide;
       number1=Double.parseDouble( JOptionPane.showInputDialog("Enter first number: "));
       number2=Double.parseDouble( JOptionPane.showInputDialog("Enter second number: "));
       sum = number1+number2;
       substract = number1-number2;
       multiply = number1*number2;
       divide=number1/number2;
       String s0="number1="+number1+" number2="+number2+"\n";
       String s1="number1 + number2 ="+sum+"\n";
       String s2="number1 - number2 ="+substract+"\n";
       String s3="number1 * number2 ="+multiply+"\n";
       String s4="number1 / number2 ="+divide+"\n";
       String s=s0+s1+s2+s3+s4;
       String sg="four aritmetik operator real numbers";
       JOptionPane.showMessageDialog(null,s,sg,JOptionPane.PLAIN_MESSAGE);
     }
 }
```



The last program is similar to the previous one, except the data is read by using Scanner class.

Program 2.1.7 BPIH2E3B.java program

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

class BPIH2E3B
 {
     public static void main(String args[])
     {
        double number1,number2;
        double sum,substract,multiply,divide;
       // create Scanner to obtain input from command window
        Scanner input = new Scanner( System.in );
        Locale us=new Locale("US");
        input.useLocale(us);
        System.out.print("enter the first number : ");
        number1=input.nextDouble();
        System.out.print("enter the second number : ");
        number2=input.nextDouble();
        sum = number1+number2;
        substract = number1-number2;
        multiply = number1*number2;
        divide=number1/number2;
        String s1="number1 + number2 ="+sum+"\n";
        String s2="number1 - number2 ="+substract+"\n";
        String s3="number1 * number2 ="+multiply+"\n";
        String s4="number1 / number2 ="+divide+"\n";
        String s=s1+s2+s3+s4;
        System.out.println(s);
      }
 }
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" BPIH2E3B
enter the first number : 1.234
enter the second number : 2.345
number1 + number2 =3.579
number1 - number2 =-1.1110000000000002
number1 * number2 =2.89373
number1 / number2 =0.5262260127931769
```

2.2 CHARACTER VARIABLES

Character (char) variable type define actual characters in computers. In order to see characters in java **ISO Unicode** character standards are used. Unicode character code consist of 4 hexagonal (base 16) number. In case you are not familiar with orthogonal number system, a table of equiavalancy is given

Tablo 2.2.1 Heksagonal(sixteen base), and 10 base and binary number systems equvalency

Heksagonal	10 base	2 base (binary)
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110

7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

in ISO unicode '\u0041' defines character 'A'. or '\u03E1'. Or code '\u03E1' defines 'a'. When the first two number of unicode equals to 0, it defines ASCII characte codes. You can find very detailed list of characters in ISO unicode by looking the internet site **http:\\unicode.org**

In the small code sample below character A and 'α' is loaded to computer as char variables in Java

```
char A1,A2;
char alpha1,alpha2;
A1='\u0041';
A2='A';
alpha1='\u03B1';
alpha2='a';
```

It should be noted that A1 and alpha1 can be seen in any environment, But A2 and alpha2 can be seen only if the editor program is written recognise the correct unicode characters. Some of the control characters are listed in Table 2.2.2

Table 2.2.2 Some special control characters

Chracter sysmbol	**action**	**Unicode symbol**
\b	**Back one character**	**\u0008**
\t	**tab**	**\u0009**
\n	**ctrl-return**	**\u000A**
\"	**“ sign**	**\u0022**
\'	**‘ sign**	**\u0027**
****	**\ sign**	**\u005C**

Some additional unicode characters are given in table 2.2.3. By using control characters several actions in print can be carried out such as back space, carriage-return, tab etc. When character variables are used, they can merge in strings, In fact strings are character arrays. When something with a character that does not available in your editor you can add that character into your String variable as char variable character. Note that some characters might not be seen properly in console environment, but can always be seen correctly in graphic environment. Because of this, **JoptionPane.showMessageDialog** method will be used for output in the example programs. The first example, characterA.java prints some characters (ügçabc) by using Unicode characters and char variable. The second program characterB.java prints a set of greek letters, The third program characterC.java prints Turkish character set. The fourth and fifth programs prints mathematical equations.

Program 2.2.1 characterA.java program

```
import java.io.*;
import javax.swing.*;

class characterA
 {
    public static void main(String args[])
    {
       char b1,b2,b3,b4,b5,b6;
       b1='ü';
       b2='ğ';
       b3='ç';
       b4='a';
       b5='b';
       b6='c';
       String s=""+b1+b2+b3+b4+b5+b6;
        JOptionPane.showMessageDialog(null,s," character
procesing",JOptionPane.PLAIN_MESSAGE);
     }
 }
```

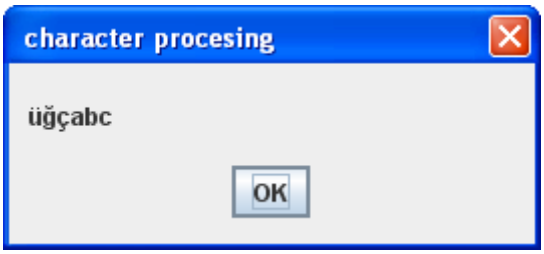


Table 2.2.3 Some unicode character tables

	000	001	002	003	004	005	006	007
0	NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
1	SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
2	STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
3	ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
4	EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
5	ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
6	ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
7	BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
8	BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
9	HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
A	LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
B	VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
C	FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
D	CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
E	SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
F	SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

	008	009	00A	00B	00C	00D	00E	00F
0	XXX 0080	DCS 0090	NB SP 00A0	° 00B0	À 00C0	Ð 00D0	à 00E0	ð 00F0
1	XXX 0081	PU1 0091	¡ 00A1	± 00B1	Á 00C1	Ñ 00D1	á 00E1	ñ 00F1
2	BPH 0082	PU2 0092	¢ 00A2	² 00B2	Â 00C2	Ò 00D2	â 00E2	ò 00F2
3	NBH 0083	STS 0093	£ 00A3	³ 00B3	Ã 00C3	Ó 00D3	ã 00E3	ó 00F3
4	IND 0084	CCH 0094	¤ 00A4	´ 00B4	Ä 00C4	Ô 00D4	ä 00E4	ô 00F4
5	NEL 0085	MW 0095	¥ 00A5	µ 00B5	Å 00C5	Õ 00D5	å 00E5	õ 00F5
6	SSA 0086	SPA 0096	¦ 00A6	¶ 00B6	Æ 00C6	Ö 00D6	æ 00E6	ö 00F6
7	ESA 0087	EPA 0097	§ 00A7	· 00B7	Ç 00C7	× 00D7	ç 00E7	÷ 00F7
8	HTS 0088	SOS 0098	¨ 00A8	¸ 00B8	È 00C8	Ø 00D8	è 00E8	ø 00F8
9	HTJ 0089	XXX 0099	© 00A9	¹ 00B9	É 00C9	Ù 00D9	é 00E9	ù 00F9
A	VTS 008A	SCI 009A	ª 00AA	º 00BA	Ê 00CA	Ú 00DA	ê 00EA	ú 00FA
B	PLD 008B	CSI 009B	« 00AB	» 00BB	Ë 00CB	Û 00DB	ë 00EB	û 00FB
C	PLU 008C	ST 009C	¬ 00AC	¼ 00BC	Ì 00CC	Ü 00DC	ì 00EC	ü 00FC
D	RI 008D	OSC 009D	SHY 00AD	½ 00BD	Í 00CD	Ý 00DD	í 00ED	ý 00FD
E	SS2 008E	PM 009E	® 00AE	¾ 00BE	Î 00CE	Þ 00DE	î 00EE	þ 00FE
F	SS3 008F	APC 009F	¯ 00AF	¿ 00BF	Ï 00CF	ß 00DF	ï 00EF	ÿ 00FF

0180 **Latin Extended-B** **024F**

	018	019	01A	01B	01C	01D	01E	01F	020	021	022	023	024
0	ƀ 0180	Ɛ 0190	Ơ 01A0	ư 01B0	ǀ 01C0	ǐ 01D0	Ǎ 01E0	ǰ 01F0	Ȁ 0200	Ȑ 0210	ɳ 0220	Ȏ 0230	ɀ 0240
1	Ɓ 0181	Ƒ 0191	ơ 01A1	Ʊ 01B1	ǁ 01C1	Ǒ 01D1	ǡ 01E1	DZ 01F1	ȁ 0201	ȑ 0211	ɖ 0221	ȏ 0231	ʔ 0241
2	Ƃ 0182	ƒ 0192	Ƣ 01A2	Ʋ 01B2	ǂ 01C2	ǒ 01D2	Ǽ 01E2	Dz 01F2	Ȃ 0202	Ȓ 0212	ȣ 0222	Ȑ 0232	ɂ 0242
3	ƃ 0183	Ɠ 0193	ƣ 01A3	Ʒ 01B3	! 01C3	Ǔ 01D3	ǽ 01E3	dz 01F3	ȃ 0203	ȓ 0213	ȣ 0223	ȳ 0233	Ƀ 0243
4	Ƅ 0184	Ɣ 0194	Ƥ 01A4	ʒ 01B4	DŽ 01C4	ǔ 01D4	Ǥ 01E4	Ǵ 01F4	Ȅ 0204	Ȕ 0214	Ȝ 0224	ɫ 0234	Ʉ 0244
5	ƅ 0185	ƕ 0195	ƥ 01A5	Ƶ 01B5	Dž 01C5	Ǖ 01D5	ǥ 01E5	ǵ 01F5	ȅ 0205	ȕ 0215	ȝ 0225	ɲ 0235	Ʌ 0245
6	Ɔ 0186	ƛ 0196	Ʀ 01A6	ƶ 01B6	dž 01C6	ǖ 01D6	Ǧ 01E6	Ƕ 01F6	Ȇ 0206	Ȗ 0216	Ȟ 0226	ɬ 0236	Ɇ 0246
7	Ƈ 0187	Ɨ 0197	Ƨ 01A7	Ʒ 01B7	LJ 01C7	Ǘ 01D7	ǧ 01E7	Ƿ 01F7	ȇ 0207	ȗ 0217	ȟ 0227	ɭ 0237	ɇ 0247
8	ƈ 0188	Ƙ 0198	ƨ 01A8	Ƹ 01B8	Lj 01C8	ǘ 01D8	Ǩ 01E8	Ǹ 01F8	Ȉ 0208	Ș 0218	Ƞ 0228	ɮ 0238	Ɉ 0248
9	Ɖ 0189	ƙ 0199	Ʃ 01A9	ƹ 01B9	lj 01C9	Ǚ 01D9	ǩ 01E9	ǹ 01F9	ȉ 0209	ș 0219	ȡ 0229	ɯ 0239	ɉ 0249
A	Ɗ 018A	ƚ 019A	ƪ 01AA	ƺ 01BA	NJ 01CA	ǚ 01DA	Ǫ 01EA	Ǻ 01FA	Ȋ 020A	Ț 021A	Ȣ 022A	Ɤ 023A	ɐ 024A
B	ƛ 018B	ƛ 019B	ƫ 01AB	Ƽ 01BB	Nj 01CB	Ǜ 01DB	ǫ 01EB	ǻ 01FB	ȋ 020B	ț 021B	ȣ 022B	Ⱦ 023B	ɑ 024B
C	ƌ 018C	Ɯ 019C	Ƭ 01AC	ƽ 01BC	nj 01CC	ǜ 01DC	Ǫ 01EC	Ǽ 01FC	Ȍ 020C	Ʒ 021C	Ȣ 022C	ȼ 023C	Ɽ 024C
D	ȸ 018D	Ɲ 019D	ƭ 01AD	ƾ 01BD	Ǎ 01CD	ǝ 01DD	ǫ 01ED	ǽ 01FD	ȍ 020D	ȝ 021D	ȣ 022D	Ł 023D	ɾ 024D
E	Ǝ 018E	ƞ 019E	Ʈ 01AE	ƿ 01BE	ǎ 01CE	Ǎ 01DE	Ǯ 01EE	Ø 01FE	Ȏ 020E	Ȟ 021E	Ȣ 022E	Ⱦ 023E	Ɏ 024E
F	Ə 018F	Ɵ 019F	Ư 01AF	ǀ 01BF	Ǐ 01CF	ǎ 01DF	ǯ 01EF	ø 01FF	ȏ 020F	ȟ 021F	ȣ 022F	ȿ 023F	ɏ 024F

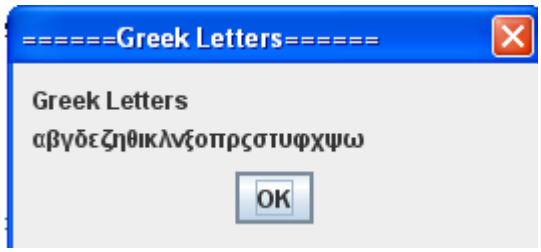
0370 **Greek and Coptic** **03FF**

	037	038	039	03A	03B	03C	03D	03E	03F
0	Ͱ 0370		ΐ 0390	Π 03A0	ΰ 03B0	π 03C0	ϐ 03D0	Ϣ 03E0	ϰ 03F0
1	ͱ 0371		Α 0391	Ρ 03A1	α 03B1	ρ 03C1	ϑ 03D1	ϣ 03E1	ϱ 03F1
2	Ͳ 0372		Β 0392		β 03B2	ς 03C2	ϒ 03D2	Ϥ 03E2	ϲ 03F2
3	ͳ 0373		Γ 0393	Σ 03A3	γ 03B3	σ 03C3	ϒ 03D3	ϥ 03E3	ϳ 03F3
4	ʹ 0374	΄ 0384	Δ 0394	Τ 03A4	δ 03B4	τ 03C4	ϓ 03D4	Ϧ 03E4	Θ 03F4
5	͵ 0375	΄΄ 0385	Ε 0395	Υ 03A5	ε 03B5	υ 03C5	ϕ 03D5	ϥ 03E5	ϵ 03F5
6	Ͷ 0376	Ά 0386	Ζ 0396	Φ 03A6	ζ 03B6	φ 03C6	ϖ 03D6	ϧ 03E6	϶ 03F6
7	ͷ 0377	· 0387	Η 0397	Χ 03A7	η 03B7	χ 03C7	ϗ 03D7	ϩ 03E7	Ϸ 03F7
8		Έ 0388	Θ 0398	Ψ 03A8	θ 03B8	ψ 03C8	Ϙ 03D8	Ϫ 03E8	ϸ 03F8
9		Ή 0389	Ι 0399	Ω 03A9	ι 03B9	ω 03C9	ϙ 03D9	ϫ 03E9	Ϲ 03F9
A	ͺ 037A	Ί 038A	Κ 039A	Ϊ 03AA	κ 03BA	ϊ 03CA	Ϛ 03DA	Ϭ 03EA	Ϻ 03FA
B	ͻ 037B		Λ 039B	Ϋ 03AB	λ 03BB	ϋ 03CB	ϛ 03DB	ϭ 03EB	ϻ 03FB
C	ͼ 037C	Ό 038C	Μ 039C	ά 03AC	μ 03BC	ό 03CC	Ϝ 03DC	Ϯ 03EC	ϼ 03FC
D	ͽ 037D		Ν 039D	έ 03AD	ν 03BD	ύ 03CD	ϝ 03DD	ϯ 03ED	Ͻ 03FD
E	; 037E	Ύ 038E	Ξ 039E	ή 03AE	ξ 03BE	ώ 03CE	Ϟ 03DE	ϰ 03EE	Ͼ 03FE
F		Ώ 038F	Ο 039F	ί 03AF	ο 03BF	Ϗ 03CF	ϟ 03DF	ϱ 03EF	Ͽ 03FF

	220	221	222	223	224	225	226	227	228	229	22A	22B	22C	22D	22E	22F
0	∀ 2200	∐ 2210	∠ 2220	∭ 2230	≀ 2240	≐ 2250	≠ 2260	≮ 2270	≺ 2280	⊐ 2290	⊠ 22A0	≺ 22B0	∧ 22C0	⋐ 22D0	≮ 22E0	⋰ 22F0
1	∁ 2201	∑ 2211	∡ 2221	∫ 2231	≁ 2241	≑ 2251	≡ 2261	≯ 2271	≻ 2281	⊑ 2291	⊡ 22A1	≻ 22B1	∨ 22C1	⋑ 22D1	≯ 22E1	⋱ 22F1
2	∂ 2202	− 2212	∢ 2222	∮ 2232	≃ 2242	≒ 2252	≢ 2262	≤ 2272	⊂ 2282	⊒ 2292	⊢ 22A2	◁ 22B2	∩ 22C2	⋒ 22D2	⋢ 22E2	∈ 22F2
3	∃ 2203	∓ 2213	∣ 2223	∯ 2233	≃ 2243	≓ 2253	≡ 2263	≥ 2273	⊃ 2283	⊓ 2293	⊣ 22A3	▷ 22B3	∪ 22C3	⋓ 22D3	⋣ 22E3	⋲ 22F3
4	∄ 2204	∔ 2214	∤ 2224	∴ 2234	≠ 2244	≔ 2254	≤ 2264	≮ 2274	⊄ 2284	⊔ 2294	⊤ 22A4	⊴ 22B4	⋄ 22C4	⋔ 22D4	⋤ 22E4	⋳ 22F4
5	∅ 2205	∕ 2215	∥ 2225	∵ 2235	≅ 2245	≕ 2255	≥ 2265	≯ 2275	⊅ 2285	⊕ 2295	⊥ 22A5	⊵ 22B5	⋅ 22C5	⋕ 22D5	⋥ 22E5	⋴ 22F5
6	∆ 2206	∖ 2216	∦ 2226	∶ 2236	≇ 2246	≠ 2256	≦ 2266	≦ 2276	⊆ 2286	⊖ 2296	⊢ 22A6	⊶ 22B6	⋆ 22C6	⋖ 22D6	≤ 22E6	⋵ 22F6
7	∇ 2207	∗ 2217	∧ 2227	∷ 2237	≉ 2247	≗ 2257	≧ 2267	≧ 2277	⊇ 2287	⊗ 2297	⊨ 22A7	⊷ 22B7	⋇ 22C7	⋗ 22D7	≥ 22E7	⋶ 22F7
8	∈ 2208	∘ 2218	∨ 2228	∸ 2238	≈ 2248	≙ 2258	≨ 2268	≩ 2278	⊈ 2288	⊘ 2298	⊨ 22A8	⊸ 22B8	⋈ 22C8	⋘ 22D8	≤ 22E8	⋷ 22F8
9	∉ 2209	∙ 2219	∩ 2229	∹ 2239	≉ 2249	≚ 2259	≩ 2269	≩ 2279	⊉ 2289	⊙ 2299	⊩ 22A9	⊹ 22B9	⋉ 22C9	⋙ 22D9	≥ 22E9	⋸ 22F9
A	∈ 220A	√ 221A	∪ 222A	∺ 223A	≅ 224A	≚ 225A	≪ 226A	≺ 227A	⊊ 228A	⊙ 229A	⊪ 22AA	⊤ 22BA	⋊ 22CA	≦ 22DA	⋠ 22EA	⋹ 22FA
B	∋ 220B	∛ 221B	∫ 222B	÷ 223B	≋ 224B	≜ 225B	≫ 226B	≻ 227B	⊋ 228B	⊛ 229B	⊫ 22AB	⊻ 22BB	≺ 22CB	≧ 22DB	⋡ 22EB	⋺ 22FB
C	∌ 220C	∜ 221C	∬ 222C	∼ 223C	≌ 224C	≜ 225C	≬ 226C	≼ 227C	⊎ 228C	⊜ 229C	⊬ 22AC	⊼ 22BC	≻ 22CC	≦ 22DC	⋢ 22EC	∋ 22FC
D	∋ 220D	∝ 221D	∭ 222D	∽ 223D	≍ 224D	≝ 225D	≭ 226D	≽ 227D	⊎ 228D	⊖ 229D	⊭ 22AD	⊽ 22BD	≏ 22CD	≧ 22DD	⋣ 22ED	⋻ 22FD
E	■ 220E	∞ 221E	∮ 222E	≈ 223E	≑ 224E	≝ 225E	≮ 226E	≲ 227E	⊎ 228E	⊞ 229E	⊯ 22AE	⊾ 22BE	∨ 22CE	≼ 22DE	⋮ 22EE	∋ 22FE
F	∏ 220F	∟ 221F	∯ 222F	∿ 223F	≏ 224F	≟ 225F	≯ 226F	≳ 227F	⊏ 228F	⊟ 229F	⊰ 22AF	⊿ 22BF	∧ 22CF	≽ 22DF	⋯ 22EF	⋸ 22FF

Program 2.2.2 characterB.java program

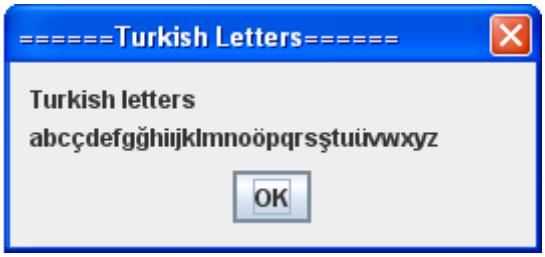
```
import java.io.*;
import javax.swing.*;
class characterB
 {
     public static void main(String args[])
     {   //Greek letters
        char b1,b2,b3,b4,b5,b6,b7,b8,b9,b10,b11,b12,b13,b14;
        char b15,b16,b17,b18,b19,b20,b21,b22,b23,b24;
        String s="Greek Letters \n";
        b1='\u03B1';  // alfa
        b2='\u03B2';  // beta
        b3='\u03B3';  // gamma
        b4='\u03B4';  // delta
        b5='\u03B5';  // epsilon
        b6='\u03B6';  // fi
        b7='\u03B7';  // eta
        b8='\u03B8';  // teta
        b9='\u03B9';  // ypsilon
        b10='\u03BA'; // kappa
        b11='\u03BB'; // lambda
        b12='\u03BC'; // mü
        b12='\u03BD'; // nü
        b13='\u03BE'; // ksi
        b14='\u03BF'; // o
        b15='\u03C0'; // pi
        b16='\u03C1'; // ro
        b17='\u03C2'; //
        b18='\u03C3'; //
        b19='\u03C4'; //
        b20='\u03C5'; //
        b21='\u03C6'; //
        b22='\u03C7'; //
        b23='\u03C8'; //
        b24='\u03C9'; //

s+=""+b1+b2+b3+b4+b5+b6+b7+b8+b9+b10+b11+b12+b13+b14+b15+b16+b17+b18+b19+b20+b21+b22+b23+b24;
         JOptionPane.showMessageDialog(null,s,"======Greek Letters======",JOptionPane.PLAIN_MESSAGE);
      }
 }
```



Program 2.2.3 characterC.java program

```
import java.io.*;
import javax.swing.*;
class charaterC
```

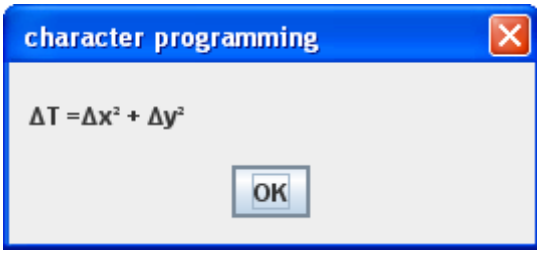
```
{   public static void main(String args[])
    {
        char b1,b2,b3,b3a,b4,b5,b6,b7,b7a,b8,b9,b9a,b10,b11,b12,b13,b14,b15,b15a,b16,b17,b18,b19,b20;
        char b21,b22,b23,b24,b25,b25a,b26,b27,b27a,b28,b29,b30,b31,b32,b33;
        String s="Turkish letters\n";
        b1='\u0061';   //a
        b2='\u0062';   //b
        b3='\u0063';   //c
        b3a='\u00E7';  //ç
        b4='\u0064';   //d
        b5='\u0065';   //e
        b6='\u0066';   //f
        b7='\u0067';   //g
        b7a='\u011F';  //ğ
        b8='\u0068';   //h
        b9a='\u0131';  //ı
        b9='\u0069';   //i
        b10='\u006A';  //j
        b11='\u006B';  //k
        b12='\u006C';  //l
        b13='\u006D';  //m
        b14='\u006E';  //n
        b15='\u006F';  //o
        b15a='\u00F6'; //ö
        b22='\u0070';  //p
        b23='\u0071';  //q
        b24='\u0072';  //r
        b25='\u0073';  //s
        b25a='\u015F'; //ş
        b26='\u0074';  //t
        b27='\u0075';  //u
        b27a='\u00FC'; //ü
        b28='\u0076';  //v
        b29='\u0077';  //w
        b30='\u0078';  //x
        b31='\u0079';  //y
        b33='\u007A';  //z
        s+=""+b1+b2+b3+b3a+b4+b5+b6+b7+b7a+b8+b9+b9a+b10+b11+b12+b13+b14+b15+b15a+b22+b23+b24+b25+b25a+b26
        JOptionPane.showMessageDialog(null,s,"======Turkish Letters======",JOptionPane.PLAIN_MESSAGE);
    }
}
```



Program 2.2.4 characterD.java program

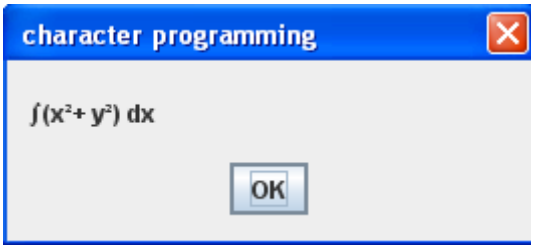
```
import java.io.*; //java girdi cikti sinifini cagir
```

```
import javax.swing.*;
class characterD
 {
    public static void main(String args[])
    {
       char b1,b2;
       b1='\u0394';   //big delta
       b2='\u00B2';   //Square sign
       String s=""+b1+"T ="+b1+"x"+b2+" + "+b1+"y"+b2;
       JOptionPane.showMessageDialog(null,s," character programming",JOptionPane.PLAIN_MESSAGE);
     }
 }
```



Program 2.2.5 characterE.java program

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;
class characterE
 {
    public static void main(String args[])
    {
       char b1,b2;
       b1='\u222B';   //integral
       b2='\u00B2';
       char b4='\u221E';
       String s=""+b1+"(x"+b2+"+ y"+b2+") dx";
       JOptionPane.showMessageDialog(null,s," character programming",JOptionPane.PLAIN_MESSAGE);
     }
 }
```



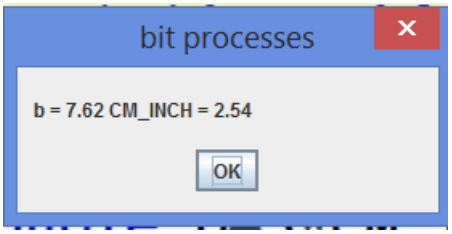
2.3 THE TERM FINAL AND CONSTANTS IN JAVA

In Java language, in order to use constants instead of variables the term **final** comes in front of variable term. A constant defined with final can not be changed. Can be defined only ones in the program. For example

final double pi=3.14159;
defines pi as a constant.

Program 2.3.1 finalA.java program : an example of usage term final

```
 import java.io.*;
import javax.swing.*;

class finalA
 {    public static void main(String args[])
      {
        final double CM_INCH = 2.54;
         double b=3*CM_INCH;
         //CM_INCH=4;
         String s="b = "+b+" CM_INCH = "+CM_INCH;
         JOptionPane.showMessageDialog(null,s,"bit processes",JOptionPane.PLAIN_MESSAGE);
    }
 }
```

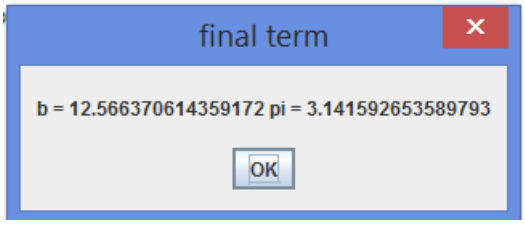



```
import java.io.*;
import javax.swing.*;

class finalA
 {    public static void main(String args[])
      {
        final double CM_INCH = 2.54;
         double b=3*CM_INCH;
         CM_INCH=4;
         String s="b = "+b+" CM_INCH = "+CM_INCH;
         JOptionPane.showMessageDialog(null,s,"bit processes",JOptionPane.PLAIN_MESSAGE);
    }
 }
```

```
---------- Capture Output ----------
> "C:\java\bin\javac.exe" finalA.java
finalA.java:9: error: cannot assign a value to final variable CM_INCH
         CM_INCH=4;
         ^
1 error

> Terminated with exit code 1.
```

Program 2.3.2 finalC.java program : an example of usage term final

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class finalC
 {   public static void main(String args[])
     {
                final double pi = 3.141592653589793;
        double b=4*pi;
        //pi=4;
        String s="b = "+b+" pi = "+pi;
        JOptionPane.showMessageDialog(null,s,"final term",JOptionPane.PLAIN_MESSAGE);
    }
 }
```




```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class finalC
 {   public static void main(String args[])
     {
                final double pi = 3.141592653589793;
        double b=4*pi;
        pi=4;
        String s="b = "+b+" pi = "+pi;
        JOptionPane.showMessageDialog(null,s,"final term",JOptionPane.PLAIN_MESSAGE);
    }
 }
```

```
---------- Capture Output ----------
> "C:\java\bin\javac.exe" finalC.java
finalC.java:9: error: cannot assign a value to final variable pi
        pi=4;
        ^
1 error

> Terminated with exit code 1.
```

WEEK 2 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX 1. By using **characterE.java** (and changing it) program, write the following equation in graphic screen

$$\int(\alpha^2+\beta^2)\,d\alpha$$

EX 2. Write the following code, compile and run it

```
public class doubleA1
{       public static void main(String[] args)
        {
                double A=1;
                A++;
                double A1=A;
                A1+=3;
                double A2=A1;
                A2--;
                double A3=A2;
                A3*=5;
                System.out.println("A="+A+"A1="+A1+"A2="+A2+"A3="+A3);
        }
}
```

EX3. Write the following code, compile and run it. Explain why it is not working, correct the error.

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class finalC
 {    public static void main(String args[])
      {
        final double pi = 3.141592653589793;
         double b=4*pi;
         pi=2*pi;
         String s="b = "+b+" pi = "+pi;
         JOptionPane.showMessageDialog(null,s,"final deyimi",JOptionPane.PLAIN_MESSAGE);
    }
 }
```

WEEK 2 HOMEWORKS

HW 1 By using ISO Unicode character codes write

$$\iint(\alpha^2+\beta^2)\,\partial\alpha\partial\beta$$

into graphic screen by using **JOptionPane.showMessageDialog** method.

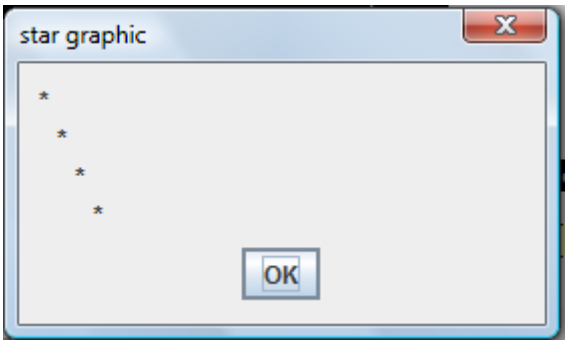
The name of the program and class will be **BPIH2HW1.java**

HW2 Investigate the folloving program

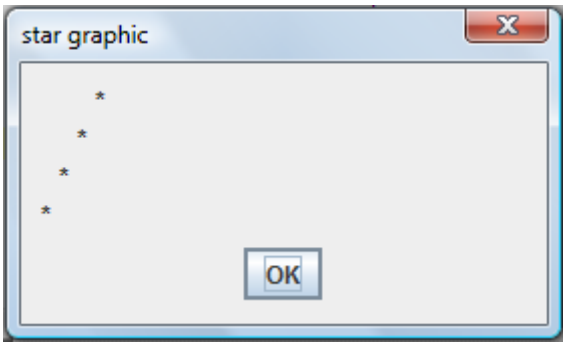
```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class star
```

```
{   public static void main(String args[])
    {
      final String star = "*";
       String s1=star+"\n";
       String s2="   "+s1;
       String s3="   "+s2;
       String s4="   "+s3;
       String s=s1+s2+s3+s4;
       JOptionPane.showMessageDialog(null,s,"star graphic",JOptionPane.PLAIN_MESSAGE);
   }
}
```

Output of this program is :



Now change the program to create the following output:



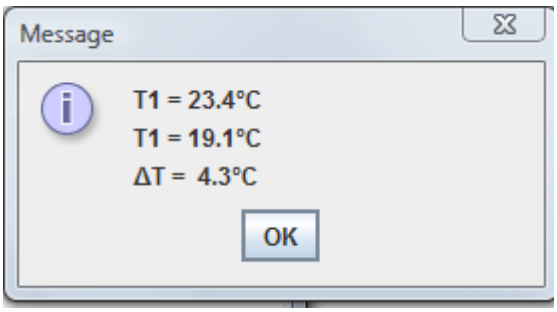
HW3. Write the following code, compile it and run it

```
import javax.swing.JOptionPane;

public class intA1
{       public static void main(String[] args)
        {
                int A=1;
                A++;
                int A1=A;
                A1+=3;
                int A2=A1;
                A2--;
                int A3=A2;
                A3*=5;
                int A4=A3;
                A4/=5;
               String s="A = "+A;
               s+=" A1 = "+A1;
               s+=" A2 = "+A2;
```

```
            s+=" A3 = "+A3;
            s+=" A4 = "+A4;
           String s1="Integer and String addition";
          JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
      }
}
```

HW4. Write the following code, compile it and run it

```
import javax.swing.JOptionPane;
import java.util.Locale;

public class BPIH2HW4A
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {  Locale us=new Locale("US");
      double T1;
      double T2; //
      double dT; // sum of number1 and number2
      String a1="Enter first temperature "+'\u000B'+"C : ";
      T1 = Double.parseDouble(JOptionPane.showInputDialog(a1));
      String a2="Enter second temperature "+'\u000B'+"C : ";
      T2 = Double.parseDouble(JOptionPane.showInputDialog(a2));
      dT = T1 - T2; // difference of temperatures
      String s1="T1 = %4.1f"+'\u00B0'+"C \n";
      String s2="T1 = %4.1f"+'\u00B0'+"C \n";
      String s3='\u0394'+"T = %4.1f"+'\u00B0'+"C \n";
      String s=s1+s2+s3;
      String s4=String.format(us,s,T1,T2,dT );
      JOptionPane.showMessageDialog(null,s4); // display sum
    } // end method main

} // end class BPIH2HW4A
```



3. INTRODUCTION TO JAVA LANGUAGE

3.1 BOOLEAN VARIABLES

Boolean type variable is for logical processes. Key word for the boolean type variable is **boolean**. Boolean variable is only one bit long and taken value **true** or **false**

boolean x=true;

boolean y=false;

The available operators for the boolean variables are:

Table 3.1.1 Boolean operators that operate directly in boolean variables

Boolean operator	Meaning
&&	and
||	or
!	not

Usage of this variables are given in

Program 3.1.1 booleanex1.java : an example of boolean variable

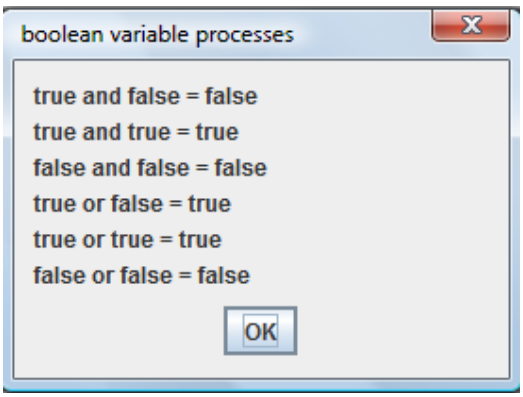
```
import java.awt.Graphics;

public class booleanex1
{
        public static void main(String arg[])
        {
         boolean b1=true;
         boolean b2=false;
         boolean b3=b1&&b2;
         boolean b4=b1||b2;
         boolean b5=!b1;
         String s="b1="+b1+"\n";
         s+="b2="+b2+"\n";
         s+="b3="+b3+"\n";
         s+="b4="+b4+"\n";
         s+="b5="+b5+"\n";
         System.out.println(s);
        }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" booleanex1
b1=true
b2=false
b3=false
b4=true
b5=false
> Terminated with exit code 0.
```

The second example creates a complete boolean process table of all possible outputs of **boolean and(&&)** and **boolean or(||)** statements

Program 3.1.2 booleanA.java : an example of boolean variable, creates boolean table

```
class booleanA
 {
    public static void main(String args[])
    {
       boolean b1,b2,b3,b4;
       boolean x1,x2,x3,x4,x5,x6;
       b1=true;
       b2=false;
       b3=true;
       b4=false;
       x1 = b1&&b2;
       x2 = b1&&b3;
       x3 = b2&&b4;
       x4 = b1||b2;
       x5 = b1||b3;
       x6 = b2||b4;
       String s1=b1+" and "+b2+" = "+x1+"\n";
       String s2=b1+" and "+b3+" = "+x2+"\n";
       String s3=b2+" and "+b4+" = "+x3+"\n";
       String s4=b1+" or "+b2+" = "+x4+"\n";
       String s5=b1+" or "+b3+" = "+x5+"\n";
       String s6=b2+" or "+b4+" = "+x6;
       String s=s1+s2+s3+s4+s5+s6;
       JOptionPane.showMessageDialog(null,s," boolean variable processes",JOptionPane.PLAIN_MESSAGE);
     }
 }
```



Boolean variables can interact with real and integer numbers through a second set of operators.

Table 3.1.3 Boolean operators that operates on integer and real variables variables

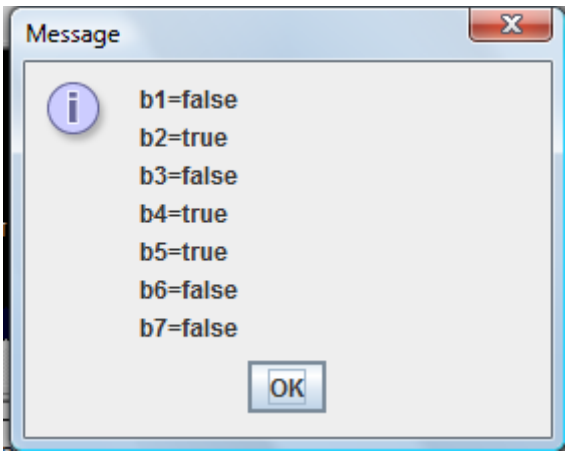
Boolean operator	Meaning
>	Grater than
<	Smaller than
==	Equal to
>=	Greater than and equal to
<=	Smaller than and equal to
!=	Not equal to

This operators operates with integer and real variables but result will be in boolean type, for example

int x=3;
int y=5;
boolean b=x>y;
will give the boolean value of false. In the second example, an example of number boolean operators. Note that due to getting result in boolean, after first process the result can be reprocessed by using operators in Table 3.1.1. Boolean Values can be added up like other type of variables to Strings to take the printput of results.

Program 3.1.4 booleanex2.java : an example of boolean variable

```
import javax.swing.JOptionPane;
public class booleanex2
{
        public static void main(String arg[])
        {
         int x=3;
         int y=4;
         boolean b1=x>y;
         boolean b2=x<y;
         boolean b3=(x==y);
         boolean b4=(x!=y);
         boolean b5=!(x==y);
         boolean b6=(x==y) && (x!=y);
         boolean b7=b3 && b4;
         String s="b1="+b1+"\n";
         s+="b2="+b2+"\n";
         s+="b3="+b3+"\n";
         s+="b4="+b4+"\n";
         s+="b5="+b5+"\n";
         s+="b6="+b6+"\n";
         s+="b7="+b7+"\n";
         JOptionPane.showMessageDialog(null,s);
        }
}
```

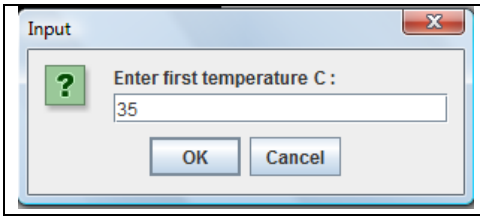


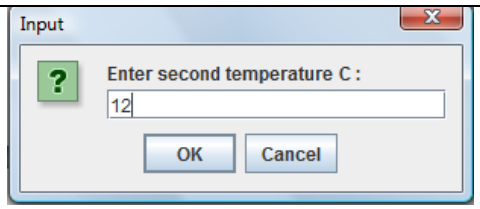
A second example will be comparison of a temperature difference and write the results booleans.

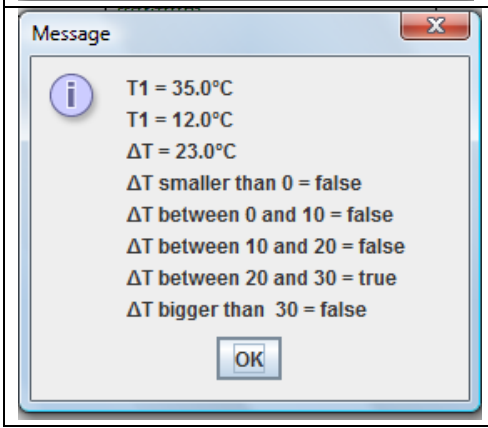
Program 3.1.5 booleanex3.java : an example of boolean variable

```
import javax.swing.JOptionPane;
import java.util.Locale;
```

```
public class booleanex3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {  Locale us=new Locale("US");
       char d='\u0394';
       char o='\u00B0';
       double T1;
       double T2; //
       double dT; // sum of number1 and number2
       String a1="Enter first temperature "+'\u000B'+"C : ";
       T1 = Double.parseDouble(JOptionPane.showInputDialog(a1));
       String a2="Enter second temperature "+'\u000B'+"C : ";
       T2 = Double.parseDouble(JOptionPane.showInputDialog(a2));
       dT = T1 - T2; // difference of temperatures
       boolean b0=(dT<0);
       boolean b1=(dT>=0) && (dT<10);
       boolean b2=(dT>=10) && (dT<20);
       boolean b3=(dT>=20) && (dT<30);
       boolean b4=(dT>=30);
       String s1="T1 = %4.1f"+o+"C \n";
       s1+="T1 = %4.1f"+o+"C \n";
       s1+=d+"T = %4.1f"+o+"C \n";
       s1+= d+"T smaller than 0 = "+b0+"\n";
       s1+= d+"T between 0 and 10 = "+b1+"\n";
       s1+= d+"T between 10 and 20 = "+b2+"\n";
       s1+= d+"T between 20 and 30 = "+b3+"\n";
       s1+= d+"T bigger than  30 = "+b4+"\n";
       String s4=String.format(us,s1,T1,T2,dT );
       JOptionPane.showMessageDialog(null,s4); // display sum
    } // end method main

} // end class booleanex3
```







3.2 IF CONTROL STRUCTURE

When a decision should be made by a computer program, if control structure used. If structure has the following form :

Program 3.2.1 booleanex3.java : an example of boolean variable

```
if(boolean variable 1)
{ executes statements here if boolean variable 1 is true}
else if (boolean variable 2)
{ executes statements here if boolean variable 2 is true}
else if (boolean variable 3)
{ executes statements here if boolean variable 3 is true}
…… (more else if statements are possible)
else
{ executes statements here if none of the above boolean variables are true}
```

In our first example program name and grade of a students read and process the grade according to number grade.

```
import javax.swing.JOptionPane;
 class if1
{
    public static void main (String args[])
    {
    double grade;
    String name;
    name=JOptionPane.showInputDialog("Enter Student\'s name : ");
    grade=Double.parseDouble(JOptionPane.showInputDialog("Enter Student\'s grade(0-100) : "
    String s1="Grade of "+name+" is  ";
      if( grade >= 90)    { s1+="A";}
      else if(grade >=75) { s1+="B";}
      else if(grade >=60) { s1+="C";}
      else if(grade >=50) { s1+="D";}
      else if(grade >=40) { s1+="E";}
      else            { s1+="F";}
      s1+="("+grade+")";
    JOptionPane.showMessageDialog(null,s1,"Studen\'s grade : ",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);
    }
}
```

Two double numbers compared in the second example

Program 3.2.2 if2.java : an example of boolean variable

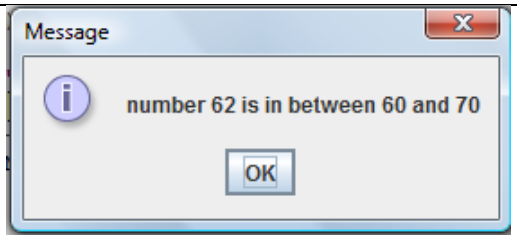
```
public class if2
{
public static void main (String args[])
{
 double x=1.245;
 double y=3.235;
 String s1="";
 if(x<y)
 { s1=x+" is smaller than "+y;}
 else if(x>y)
 { s1=x+" is bigger than "+y;}
  else if(x==y)
```

```
 { s1=x+" is equal to "+y;}
  System.out.println(s1);
 }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" if2
1.245 is smaller than 3.235

> Terminated with exit code 0.
```

Program 3.2.3 if3.java : an example of boolean variable

```
import java.awt.Graphics;
import javax.swing.JOptionPane;
import javax.swing.JApplet;

public class if3
{
public static void main (String args[])
{
 int i=62;
 int i1=60;
 int i2=70;
 String s1="";
 boolean b1=(i>=i1) && (i<i2);
 if(b1)
 { s1="number "+i+" is in between "+i1+" and "+i2;}
 else
  { s1="number "+i+" is not in between "+i1+" and "+i2;}
  JOptionPane.showMessageDialog(null,s1);
}
}
```

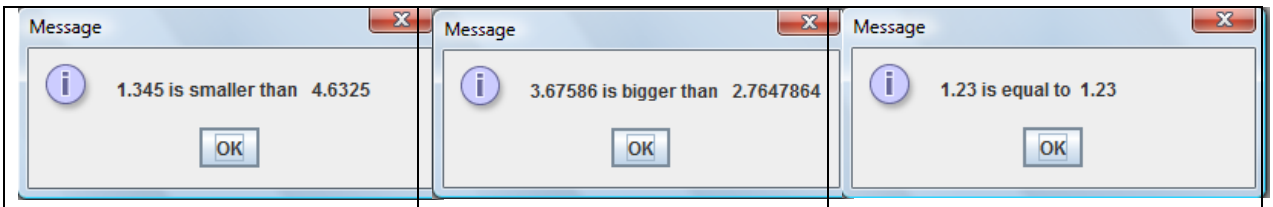


In the next example two double variables are compared

Program 3.2.4 if_double.java : an example of double variable comparison

```
import java.io.*;
import javax.swing.JOptionPane;

class if_double
{
public static void main(String args[])
 {
 double x;
 double y;
 x=Double.parseDouble(JOptionPane.showInputDialog(" x = "));
 y=Double.parseDouble(JOptionPane.showInputDialog(" y = "));
 if(x<y)
 { JOptionPane.showMessageDialog(null,x+" is smaller than   "+y);}
```

```
 else if(x>y)
 { JOptionPane.showMessageDialog(null,x+" is bigger than   "+y);}
  else if(x==y)
 { JOptionPane.showMessageDialog(null,x+" is equal to  "+y);}
}
}
```

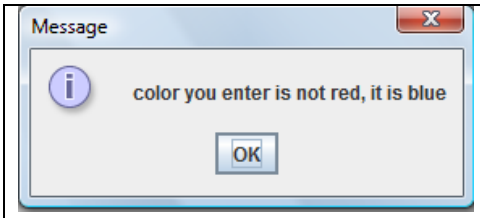


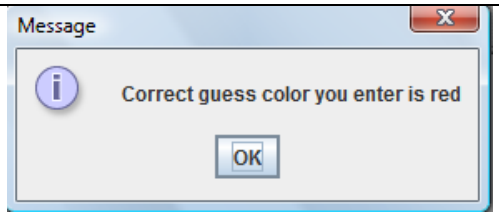
In the next example String variables compared. In order to check if two String variables are equal or not String comparision method equals can be used.

String s1=”John”;
String s2=”Jane”;
Boolean b1=s1.equals(s2);

Program 3.2.5 if_String.java : an example of String variable comparison

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.JOptionPane;

class if_String
{
public static void main(String args[])
 {
 //Guess the color
 String s1;
 String s=JOptionPane.showInputDialog(" Enter a color = ");
 boolean b1=s.equals("red");
 if(b1) {s1="  Correct guess color you enter is red";}
 else   {s1="  color you enter is not red, it is "+s;}
 JOptionPane.showMessageDialog(null,s1);
}
}
```





When a complex decision making is required it can be achived as nested if structures (and if structure inside another if structure) or composite if structure(more than one boolean is combined together by joining and(&&), or(||) type boolean processors). In the next example problem 3.2.6 a nested if structure example is given. In the second example problem 3.2.6 a composite if structure is carried out exactly the same program output.

Program 3.2.6 nested if.java : an example of nested if structure

```
import java.io.*;
import javax.swing.*;

class nestedif
{
public static void main(String args[])
{
double x = 7.0;
double y = 3.0;
if(x > 5)
{
    if(y > 5) { System.out.println("x and y are greater than 5.");}
    else if (y==5) { System.out.println("x is greater than 5  and y is equal to 5");}
    else { System.out.println("x is greater than 5 and y is smaller than 5");}
}
else
{
   if(y > 5) { System.out.println("x is smaller or equal to five  y is greater than 5.");}
   else if(y==5) { System.out.println("x is smaller or equal to five  y is equal to 5.");}
   else { System.out.println("x is smaller or equal to five  y is smaller than 5."); }
}
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" nestedif
x is greater than 5 and y is smaller than 5

> Terminated with exit code 0.
```

Program 3.2.56 composite if.java : an example of composite if structure

```
import java.io.*;
import javax.swing.*;

class compositeif
{
public static void main(String args[])
{
double x = 7.0;
double y = 3.0;
if(x>5 && y>5) { System.out.println("x and y are greater than 5.");}
else if(x>5 && y==5) { System.out.println("x and y are greater than 5.");}
else if(x>5 && y<5) { System.out.println("x is greater than 5 and y is smaller than 5");}
else if(x<=5 && y>5) { System.out.println("x is smaller or equal to five  y is greater than 5.");}
else if(x<=5 && y==5) { System.out.println("x is greater than 5  and y is equal to 5");}
else if(x <= 5 && y<5) { System.out.println("x is smaller or equal to five  y is smaller than 5."); }
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" compositeif
x is greater than 5 and y is smaller than 5

> Terminated with exit code 0.
```

As you observed from the example, any complex logical processes can be achived by using if structural blocks. If structure is one of the widely use computer structure.

3.3 BIT PROCESSING IN JAVA

From time to time it could be necassary to proceed directly with the bits in computer. In java a set of processors are given to manupulate directly the bits. In order to understand the bir processes we should understand the basic way of data representation. A byte type integer data representation in bit structure in computer was given in Table 1.3.2

Table 3.3.1 writing 15 in a 8 bit integer variable (byte)

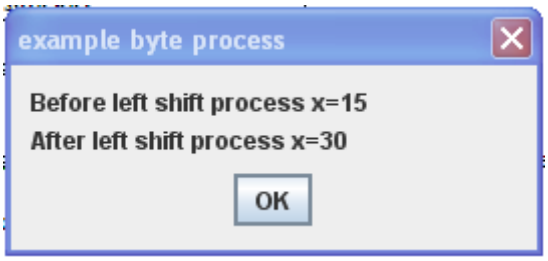
+/-	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+	64	32	16	8	4	2	1
0	0	0	0	1	1	1	1

Table 3.3.2 Bit processors and their meaning

Bit processor	Meaning
&	AND
|	OR
^	XOR
~	NOT
>>	Shift one bit to the right
<<	Shift one bit to the left

Program 3.3.1 bit.java : an example of bit proses << (shift one bit to the left)

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class bit
{   public static void main(String args[])
    {
            int x=15;
            String s="Before left shift process x="+x+"\n";
        x=x<<1;
        s+="After left shift process x="+x;
        JOptionPane.showMessageDialog(null,s,"example byte
process",JOptionPane.PLAIN_MESSAGE);
    }
}
```



What happened to the number. It is beter to explain it with the bit table again (table in byte type is given to avoid 32 bit table of int type integer variable)

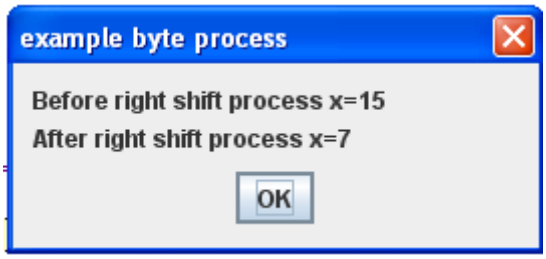
Table 3.3.2 shifting the bits of Table 1.3.2 one to the left a 8 bit integer variable (byte)

	+/-	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	+	64	32	16	8	4	2	1
Before (15)	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
After(30)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**

As you see from the table, number become 30. If left shift apply to the same number(15), number becomes 7.

Program 3.3.2 bit1.java : an example of bit proses >> (shift one bit to the right)

```
import javax.swing.*;
class bit1
{   public static void main(String args[])
    {          int x=15;
       String s="Before right shift process x="+x+"\n";
       x=x>>1;
       s+="After right shift process x="+x;
       JOptionPane.showMessageDialog(null,s,"example byte process",JOptionPane.PLAIN_MESSAGE);
    }
}
```



If this process shown in the bit table

Table 3.3.2 shifting the bits of Table 1.3.2 one to the right a 8 bit integer variable (byte)

	+/-	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	+	64	32	16	8	4	2	1
Before (15)	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
After(7)	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**

In reality int has 32 bit, therefore
int x=15;
will assign bit value as:
00000000000000000000000000001111
And after left shift (<<) operation it will be:
00000000000000000000000000011110

Several bit processes are shown in the next 3 example problems bitA.java, bitB.java and bitD.java

Program 3.3.3 bitA.java : an example of bit proseses

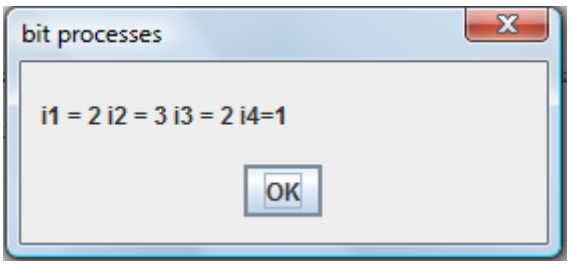
```
class bitA
{   public static void main(String args[])
    {
       //or | bit işlemi
```

```
                int i1=1;
        int i2=2;
        int i3=i1|i2;
        int i4=~i3;
        String s="i1 = "+i1+" i2 = "+i2+" i3 = "+i3+"i4="+i4;
        System.out.println(s);
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" bitA
i1 = 1 i2 = 2 i3 = 3i4=-4

> Terminated with exit code 0.
```

Program 3.3.4 bitB.java : an example of bit proseses

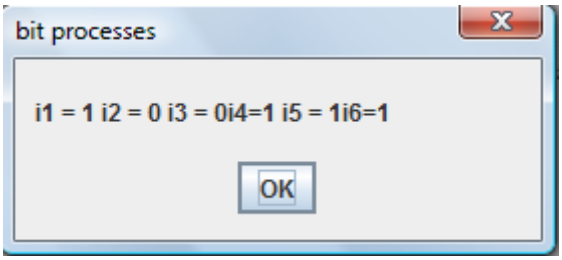
```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class bitB
{   public static void main(String args[])
    {
        //xor  bit işlemi
        int i1=2;
        int i2=3;
        int i3=i1&i2;
        int i4=i3>>1;
        String s="i1 = "+i1+" i2 = "+i2+" i3 = "+i3+" i4="+i4;
        JOptionPane.showMessageDialog(null,s,"bit processes",JOptionPane.PLAIN_MESSAGE);
    }
}
```



Program 3.3.4 bitD.java : an example of bit proseses

```
import java.io.*;
import javax.swing.*;

class bitD
{   public static void main(String args[])
    {
        //or | bit process
        int i1=1;
        int i2=0;
        int i3=i1&i2;
        int i4=i1|i2;
        int i5=i1^i2;
        int i6=~i1;
```

```
        String s="i1 = "+i1+" i2 = "+i2+" i3 = "+i3+"i4="+i4+" i5 = "+i5+"i6="+i4;
        JOptionPane.showMessageDialog(null,s,"bit processes",JOptionPane.PLAIN_MESSAGE);
    }
}
```



WEEK 3 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX 1. By using boolean variables create an AND(&&), and an OR(||) table . the values should be as follows :

true && true = true
true && false = false
false && true = false
false && false = false

true || true = true
true || false = true
false || true = true
false || false = false

take a screen output in graphic format(JOptionPane.showMessegeDialog)

EX 2. Select a color (String variable). If color is blue, black, Brown, grey write "you select a cold color". If color is yellow, green, red write "you select a cold color"

EX 3. Enter 4 numbers. For each numbers if the number less than avarage value, write "you failed". If the number is bigger or equal to avarage, write "you passed"

HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) Enter Two double number from the screen. Compare two numbers and write the results. For example if the numbers you entered 1.23 and 2.34 Write 1 > 2 false
1.23 < 2.34 true
1.23 < 2.34 false
1.23 == 2.34 false

HW2) For a student read name, homework grade, midterm examination grade and final grade. Calculate class grade by equation
Class grade = 0.25*homework grade + 0.35*midterm grade+0.4*final grade
if total grade is less tan 60 write "you failed", if it is equal or bigger than 60, write "you passed"

HW3) Enter a name if the name is Mehmet or Ahmet or Hüseyin or ali or veli or hasan or deniz write "you are lucky, you win a million TL". If not write "sorry you lost".

4. INTRODUCTION TO JAVA LANGUAGE : LOOP STRUCTURES

4.1 WHILE LOOP STRUCTURE

In the if structure of the previous section, it is seen that if the given boolean value is true, computer carries out instruction inside of the paranthesis. In the while structure it is similar to if structure, with one difference. In the if statement inside of the paranthesis instructions will be carried out once. In while statements, it will go into the paranthesis as repeatedly, as long as the boolean is true.

while(boolean variable)
{ processes while boolean variable is true}
For example while loop in the following structure will continue infinitely
{ System.out.println("This loop will be infinite");}
In order to stop the loop the value of the boolean should change somewhere in the loop

```
int number=0;
while(number <= 100)
{ number += 1; }
```

This loop will turn 100 times and when number is bigger than 100, it will stop. The same loop could be written as

```
int number=0;
boolean b= number <= 100;
while(b)
{ number += 1;
 b= number <= 100;
 }
```

In our first example numbers from 1 to number1 is added and sum is calculated.

Program 4.1.1 BPIH4E1.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E1
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
      int sum=0;  // sum of numbers
      System.out.print( "Enter the number: " ); // prompt
      number1 = input.nextInt(); // read first number from user
      int i=1;
      while(i<=number1)
      { sum += i; // add numbers
        i++;
      }
      System.out.printf( "Sum is %d\n", sum ); // display sum
    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E1
Enter the number: 4
Sum is 10
> Terminated with exit code 0.
```

In the second example numbers from number 1 to number 2 is added and sum is calculated.

Program 4.1.2 BPIH4E2.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E2
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
      int number2;  //last number to add
      int sum=0; // sum of number1 and number2
       System.out.print( "Enter the first number: " ); // prompt
      number1 = input.nextInt(); // read first number from user
       System.out.print( "Enter the second number: " ); // prompt
      number2=input.nextInt();
      int i=number1;
      while(i<=number2)
      { sum += i; // add numbers
        i++;
      }
      System.out.printf( "Sum from %d to %d = %d \n", number1,number2,sum ); // display sum
    } // end method main
} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E2
Enter the first number: 1
Enter the second number: 100
Sum from 1 to 100 = 5050

> Terminated with exit code 0.
```

In the third example we will process Strings instead of just numbers by using while statement

Program 4.1.3 BPIH4E3.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E3
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
      System.out.print( "Enter the number: " ); // prompt
      number1 = input.nextInt(); // read first number from user
      int i=0;
      String star="*";
      String space=" ";
      String sum=star;
      while(i<=number1)
      {  sum = space+sum; // add Strings
        System.out.println(sum ); // display sum
        i++;
```

```
        }

    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E3
Enter the number: 4
 *
  *
   *
    *
     *
> Terminated with exit code 0.
```

In the next example number and stars are written in on th same line

Program 4.1.4 BPIH4E4.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E4
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int number1; // first number to add
      System.out.print( "Enter the number: " ); // prompt
      number1 = input.nextInt(); // read first number from user
      int i=0;
      String star="*";
      String space=" ";
      String sum=star;
      while(i<=number1)
      {  sum = star+i+space; // add Strings
        System.out.print(sum); // display sum
        i++;
      }
    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E4
Enter the number: 5
*0 *1 *2 *3 *4 *5
> Terminated with exit code 0.
```

In the next example a primitice type of plot (line plot) is drawn by usıng while loop and * sign

Program 4.1.5 BPIH4E5.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E5
```

```
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      int x1; // first x to plot
      int x2; // last x to plot
      int y=0;
      System.out.print( "Enter the first number x1=" ); // prompt
      x1=input.nextInt(); // read first number from user
      System.out.print( "Enter the first number x1=" ); // prompt
      x2=input.nextInt(); // read second number from user
      String star="*";
      String space=" ";
      String sum;
      String s="";
      int x=x1;
      while(x<=x2)
      {   y=x*x+2*x+1;  // function to be plotted
         int i=0;
         sum=star;
         while(i<y)
         { sum = space+sum;
           i++;
         }
         s+=sum+"\n";

         x++;
      }
      System.out.println(s);
    } // end method main
} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E5
Enter the first number x1=-4
Enter the first number x1=4
       *
   *
 *
*
 *
   *
      *
           *
                *


> Terminated with exit code 0.
```

When a complex numbers such as π=3.14159… , series solution is used. A series

can be used. Let us look at how a series like this can be put into the prgram by using a while loop.

Program 4.1.6 BPIH4E6.java : an example of while loop

```
public class BPIH4E6
```

```
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      //pi = 4 - 4/3 + 4/5 -4/7+4/9-4/11+...+
      String s;
      double pi=0;
      int artieksi=-1;
      int i=1;
      while(i < 100000)
      {
      artieksi*=-1;
      pi+=4/(2.0*i-1.0)*artieksi;
      i++;
      }
      s="pi = "+pi;
      System.out.println(s);
    } // end method main
} // end class BPIH4E1
```

In the next example factorial will be calculated. Factorial is defined as :
n!=n*(n-1)*(n-2)….4.3.2.1 4!= 4*3*2*1 3!=3*2*1

Program 4.1.7 BPIH4E7.java : an example of while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E7
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
              int number; // number to get
      int y=0;
      System.out.print( "Enter the number n=" ); // prompt
      number=input.nextInt(); // read first number from user
      long factorial=1;
      String s="";
      int i=1;
      while(i<=number)
      {  factorial*=i;
        i++;
      }
      s=""+number+"! = "+factorial;
      System.out.println(s);
    } // end method main
} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E7
Enter the number n=4
```

```
4! = 24
> Terminated with exit code 0.
```

In the last example a more complex series will be calculated

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

note that e=2.7182818…

Program 4.1.8 BPIH4E8.java : an example of while loop

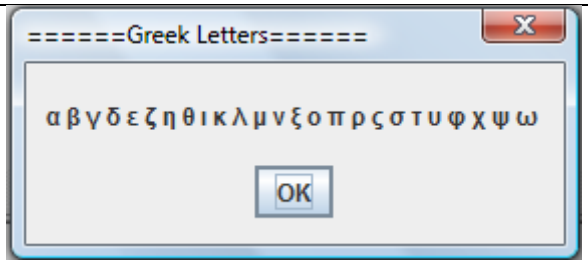
```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH4E8
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // number e = 2.718...
      // exp(x) = 1 + x /1! + x2 / 2! + x3 / 3! + x4 / 4! + ..
      String s;
      // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      Locale us=new Locale("US");
      input.useLocale(us);
      System.out.print( "Enter x : " ); // prompt
      double x = input.nextDouble(); // read first number from user
      double power=1;
      double exponent=1;
      double factorial=1;
      double number=1;
      while(number<=200)
      {
      factorial*=number;
      power*=x;
      exponent+=power/factorial;
      number++;
      }
      s="exp("+x+") = "+exponent;
      System.out.println(s);
    } // end method main
} // end class BPIH4E1
```

It is possible to use also char variable in while loop counter

Program 4.1.9 BPIH4E12.java : an example of while loop

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;
```

```
class BPIH4E12
 {    public static void main(String args[])
      {
         String s="";
         char b='\u03B1';
         while(b<='\u03C9')
         {s+=b+" ";
         b++;}
         JOptionPane.showMessageDialog(null,s,"======Greek
Letters======",JOptionPane.PLAIN_MESSAGE);
      }
 }
```



4.2 DO-WHILE LOOP STRUCTURE

Do-while structure is very similar to while structure. Except in do-while statements in the paranthesis will be carried out at least ones, in while statement if the boolean statement is false, it will not go inside of the loop at all.

do
{ processes while boolean variable is true}
while(boolean variable);

In order to stop the loop the value of the boolean should change somewhere in the loop
For example in the first while loop the value of number will be 11 (it will not enter to the loop)

```
int number=11;
while(number <= 10)
{ number += 1; }
```

In the next do-while statement, the value of number will be 12. It will go into the loop once.

```
int number=11;
do
{ number += 1; }
while(number <= 10);
```

In our first example numbers from 1 to number1 is added and sum is calculated.
We will list the few codes, similar to while statements code here, but the all codes similar to while codes are available in exercise folder.

Program 4.2.1 BPIH4E1A.java : an example of do-while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E1A
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
```

```
        // create Scanner to obtain input from command window
        Scanner input = new Scanner( System.in );
        int number1; // first number to add

        int sum=0; // sum of number1 and number2
        System.out.print( "Enter the number: " ); // prompt
        number1 = input.nextInt(); // read first number from user
        int i=1;
        do
        { sum += i; // add numbers
          i++;
        }  while(i<=number1);
        System.out.printf( "Sum is %d\n", sum ); // display sum
    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E1A
Enter the number: 100
Sum is 5050

> Terminated with exit code 0.
```

Program 4.2.2 BPIH4E4A.java : an example of do-while loop

```
import java.util.Scanner; // program uses class Scanner

public class BPIH4E4A
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        // create Scanner to obtain input from command window
        Scanner input = new Scanner( System.in );
        int number1; // first number to add
        System.out.print( "Enter the number: " ); // prompt
        number1 = input.nextInt(); // read first number from user
        int i=0;
        String star="*";
        String space="  ";
        String sum=star;
        String s="";
        do
        {  s += star+i+space; // add Strings
           i++;
        }  while(i<=number1);
        System.out.printf("%s",s);
    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E4A
Enter the number: 20
*0  *1  *2  *3  *4  *5  *6  *7  *8  *9  *10  *11  *12  *13  *14  *15  *16  *17  *18  *19  *20
> Terminated with exit code 0.
```

In this example an equation y=f(x)= 5.67*x*x+ 2.25*x-3.25 is calculated in and result is printed out.

Program 4.2.3 BPIH4E9A.java : an example of do-while loop

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;

public class BPIH4E9A
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      Scanner input = new Scanner( System.in );
      Locale us=new Locale("US");
      input.useLocale(us);
      System.out.print( "Enter first double x1: " ); // prompt
      double x1 = input.nextDouble(); // read first number from user
      System.out.print( "Enter last double x2: " ); // prompt
      double x2 = input.nextDouble(); // read first number from user
      System.out.print( "Enter increment step dx: " ); // prompt
      double dx = input.nextDouble(); // read first number from user
      double x=x1;
      String s="";
      double y=0;
      do
      { y=5.67*x*x+ 2.25*x-3.25; // add numbers
        x+=dx;
        s+="x = "+x+" y = "+y+"\n";
      }  while(x<=x2);
      System.out.println(s); // display x and y
    } // end method main
} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E9A
Enter first double x1: 0
Enter last double x2: 10
Enter increment step dx: 0.2
x = 0.2 y = -3.25
x = 0.4 y = -2.5732
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E9A
Enter first double x1: 0
Enter last double x2: 2
Enter increment step dx: 0.2
x = 0.2 y = -3.25
x = 0.4 y = -2.5732
x = 0.6000000000000001 y = -1.4427999999999999
x = 0.8 y = 0.14120000000000088
x = 1.0 y = 2.1788000000000007
x = 1.2 y = 4.67
x = 1.4 y = 7.614799999999999
x = 1.5999999999999999 y = 11.0132
x = 1.7999999999999998 y = 14.865199999999998
x = 1.9999999999999998 y = 19.170799999999996
x = 2.1999999999999997 y = 23.929999999999993

> Terminated with exit code 0.
```

4.3 FOR LOOP STRUCTURE

For loop structure is also similar to while structure. Except initial condition and incrimental conditions are also given in the structure

for(initial statements, boolean condition, incremental statements)
{ processes while boolean variable is true}

Some example statements:

for(int number=0,int sum=0;number <= 10;number++)
{sum+=number; }

for(double number=0,double sum=0;number <= 10.0;number+=0.1)
{sum+=number; }

For loops are usually used as a counter (similar tos um or product statements in math), but for and while can always be replaced each other. To show this fact the example problems given in this subchapter will be the same as while examples except graphic screen input and output is used in here.
In our first example numbers from 1 to number1 is added and sum is calculated.

Program 4.3.1 BPIH4E1B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E1B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      int number1; // first number to add
      int sum=0; // sum of number1 and number2
      number1=Integer.parseInt(JOptionPane.showInputDialog("Enter first integer: ")); // read number
      for(int i=1;i<=number1;i++)
      { sum += i; // add numbers
      }
      System.out.printf( "Sum is %d\n", sum ); // display sum
    } // end method main

} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E1B
Sum is 5050
> Terminated with exit code 0.
```

In the second example numbers from number 1 to number 2 is added and sum is calculated.

Program 4.3.2 BPIH4E2B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E2B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      int number1; // first number to add
      int number2;  //last number to add
      int sum=0; // sum of number1 and number2
      number1=Integer.parseInt(JOptionPane.showInputDialog("Enter first integer: ")); // read number
      number2=Integer.parseInt(JOptionPane.showInputDialog("Enter last integer: ")); // read number
```

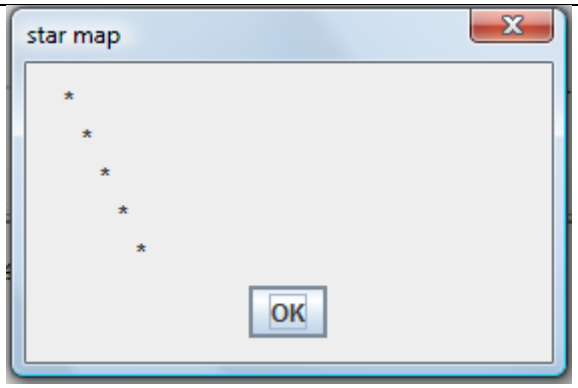
```
        for(int i=number1;i<=number2;i++)
        { sum += i; // add numbers
        }
        System.out.printf( "Sum from %d to %d = %d \n", number1,number2,sum ); // display sum
    } // end method main
} // end class BPIH4E1
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" BPIH4E2B
Sum from 1 to 100 = 5050
> Terminated with exit code 0.
```

In the third example we will process Strings instead of just numbers by using while statement

Program 4.3.3 BPIH4E3B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E3B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        int number1; // first number to add
        number1=Integer.parseInt(JOptionPane.showInputDialog("Enter number of stars: ")); // read number
        String star="*";
        String space="   ";
        String sum=star;
        String s="";
        for(int i=0;i<number1;i++)
        {  sum = space+sum; // add Strings
           s+=sum+"\n";
        }
        JOptionPane.showMessageDialog(null,s,"star map",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E1
```

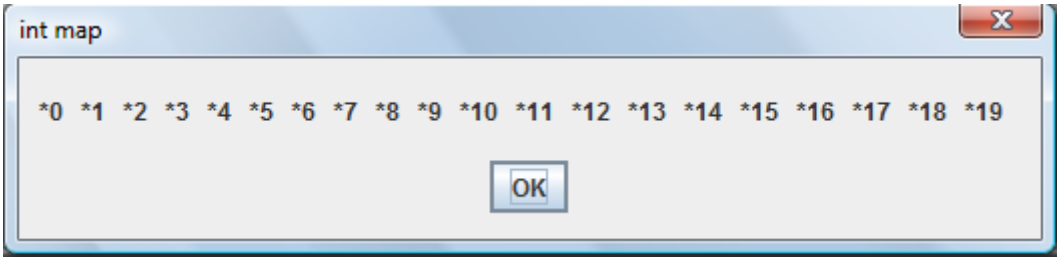


In the next example number and stars are written in on th same line

Program 4.3.4 BPIH4E4B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E4B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        int number1; // first number to add
```

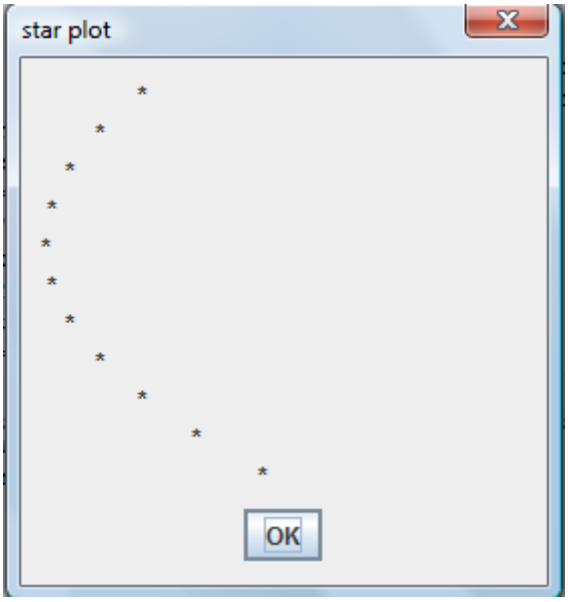
```
        number1=Integer.parseInt(JOptionPane.showInputDialog("Enter number of starts: ")); // read number
        String star="*";
        String space="   ";
        String s="";
        for(int i=0;i<number1;i++)
        {  s += star+i+space; }
        JOptionPane.showMessageDialog(null,s,"int map",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E1
```



In the next example a primitive type of plot (line plot) is drawn by using while loop and * sign

Program 4.3.5 BPIH4E5B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E5B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        int x1; // first x to plot
        int x2; // last x to plot
        int y=0;
        x1=Integer.parseInt(JOptionPane.showInputDialog("Enter first number: ")); // read number
        x2=Integer.parseInt(JOptionPane.showInputDialog("Enter last number: "));  // read number
        String star="*";
        String space=" ";
        String sum=star;
        String s="";
        for(int x=x1;x<=x2;x++)
        {   y=x*x+2*x+1;  //function to be plotted
            for(int i=0;i<y;i++)
            { sum = space+sum;}
            s+=sum+"\n";
            sum=star;
        }
        JOptionPane.showMessageDialog(null,s,"star plot",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E5B
```

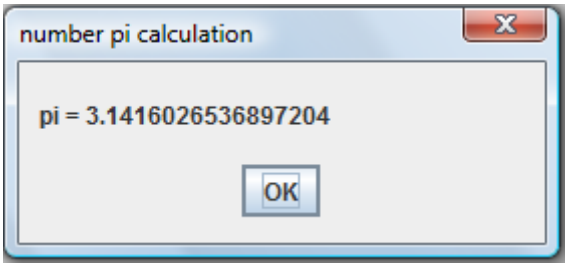


When a complex numbers such as π=3.141592653589793, series solution is used. A series

can be used. Let us look at how a series like this can be put into the program by using a while loop.

Program 4.3.6 BPIH4E6.java : an example of for loop

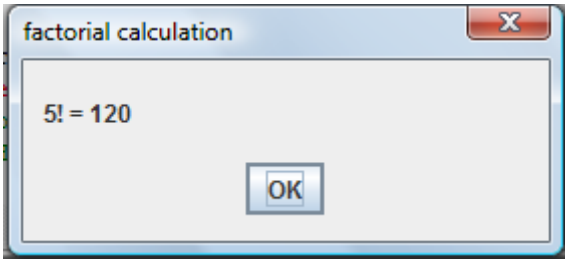
```
import javax.swing.*; // program uses class Scanner

public class BPIH4E6B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      //pi = 4 - 4/3 + 4/5 -4/7+4/9-4/11+...+
      String s;
      int i;
      double pi=0;
      int artieksi=-1;
      for(i=1;i < 100000;i++)
      {
      artieksi*=-1;
      pi+=4/(2.0*i-1.0)*artieksi;
      }
      s="pi = "+pi;
      JOptionPane.showMessageDialog(null,s,"number pi
calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E6B
```



In the next example factorial will be calculated. Factorial is defined as :
n!=n*(n-1)*(n-2)….4.3.2.1 4!= 4*3*2*1 3!=3*2*1

Program 4.3.7 BPIH4E7B.java : an example of for loop

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E7B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // factorial
      String s;
      long factorial=1;
      int number=Integer.parseInt(JOptionPane.showInputDialog("Enter a number: ")); // read number
      for(int i=1;i <=number;i++)
      {
      factorial*=i;
      }
      s=""+number+"! = "+factorial;
      JOptionPane.showMessageDialog(null,s,"factorial calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E7B
```



In the next example a more complex series will be calculated

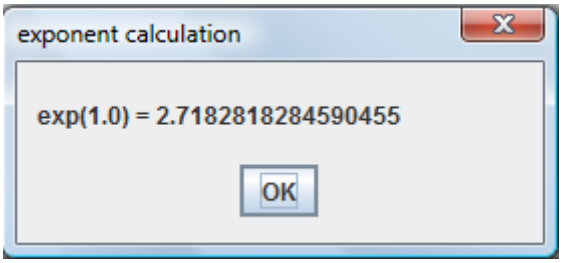
$$e = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

note that e=2.7182818…

Program 4.3.8 BPIH4E8B.java : an example of for loop

```
import javax.swing.*; // program poweres class Scanner

public class BPIH4E8B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // number e = 2.718...
```

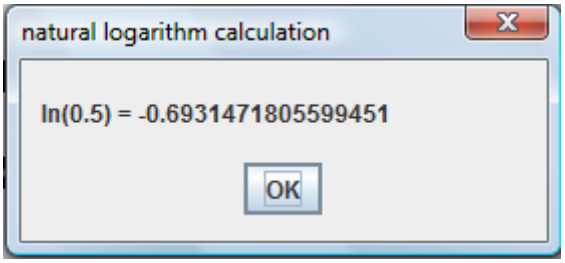
```
        // exp(x) = 1 + x /1! + x2 / 2! + x3 / 3! + x4 / 4! + ..
        String s;
        double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
        double power=1;
        double exponent=1;
        double factorial=1;
        for(double sayi=1;sayi<=200;sayi++)
        {
        factorial*=sayi;
        power*=x;
        exponent+=power/factorial;
        }
        s="exp("+x+") = "+exponent;
        JOptionPane.showMessageDialog(null,s,"exponent calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E8B
```



Another series example will be

Program 4.3.9 BPIH4E13B.java : an example of for loop

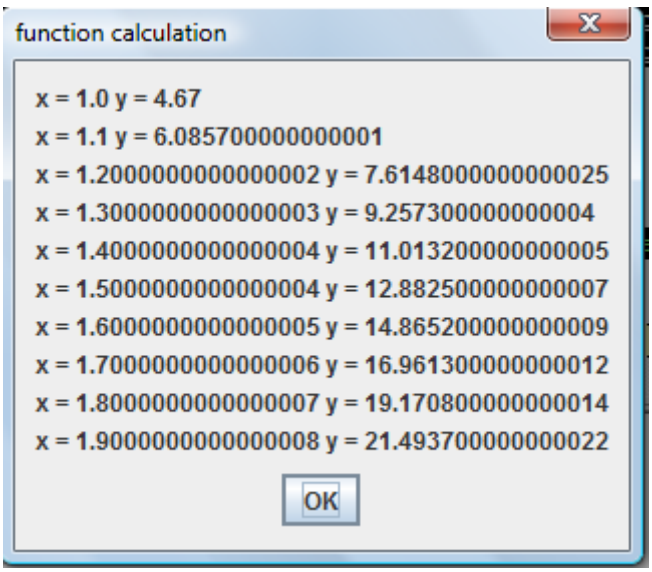
```
import javax.swing.*; // program poweres class Scanner

public class BPIH4E13B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
        // number e = 2.718...
        // ln(x) = (x-1) - (x-1)^2/2+(x-1)^3/3+(x-1)^4/4..
        // for x<2;
        String s;
        double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
        double power=1;
        double ln=0;
        double xm1=x-1;;
        int plusminus=-1;
        for(double n=1;n<=10000000;n++)
        {
                plusminus*=-1;
                power*=xm1;
        ln+=plusminus*power /n;
        }
        s="ln("+x+") = "+ln;
        JOptionPane.showMessageDialog(null,s,"natural logarithm calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E13B
```



In the last example two numbers and a number increment read from the program and a function calculated results are listed.

Program 4.3.10 BPIH4E9B.java : an example of for loop

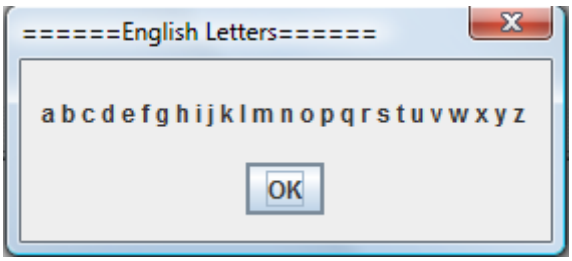
```
import javax.swing.*; // program uses class Scanner

public class BPIH4E9B
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      double x1; // first number to add
      double x2;  //last number to add
      double dx;
      double y;
      x1=Double.parseDouble(JOptionPane.showInputDialog("Enter first double: ")); // read number
      x2=Double.parseDouble(JOptionPane.showInputDialog("Enter last double: ")); // read number
      dx=Double.parseDouble(JOptionPane.showInputDialog("Enter increment step: ")); // read number
      String s="";
      for( double x=x1;x<=x2; x+=dx)
      { y=5.67*x*x+ 2.25*x-3.25;// calculate y
        s+="x = "+x+" y = "+y+"\n";
      }
      JOptionPane.showMessageDialog(null,s,"function calculation",JOptionPane.PLAIN_MESSAGE); //display x and y
    } // end method main
} // end class BPIH4E9B
```



By using for loop structure, also char type varibles can be counted

Program 4.3.11 BPIH4E12B.java : an example of for loop

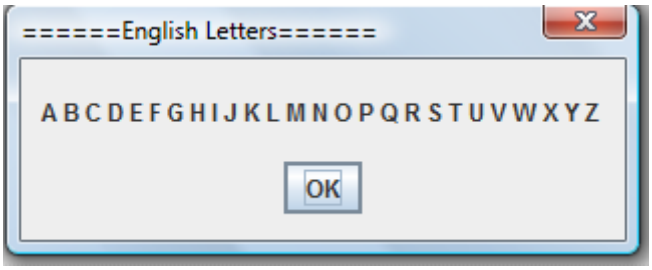
```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class BPIH4E12B
 {   public static void main(String args[])
     {
        String s="";
        for(char b='\u0061';b<='\u007A';b++)
        {s+=b+" ";}
        JOptionPane.showMessageDialog(null,s,"======English Letters======",JOptionPane.PLAIN_MESSAGE);
    }
 }
```



The same program can be given by using char names instead of their Unicode characters

Program 4.3.12 BPIH4E13B.java : an example of for loop

```
import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;

class BPIH4E13B
 {   public static void main(String args[])
     {
        String s="";
        for(char b='A';b<='Z';b++)
        {s+=b+" ";}
        JOptionPane.showMessageDialog(null,s,"======English Letters======",JOptionPane.PLAIN_MESSAGE);
    }
 }
```



4.4 SWITCH-CASE STRUCTURE

Switch-case structure is a structure closer to if structure than the above loop structures. This statement is not widely use among programmers because the same thing can be achived by using if blocks. The structure has the following form :

```
switch( variable name)
{
case the value of the variable :
If the variable has the value above carry out statements here
break; // end of structure
```

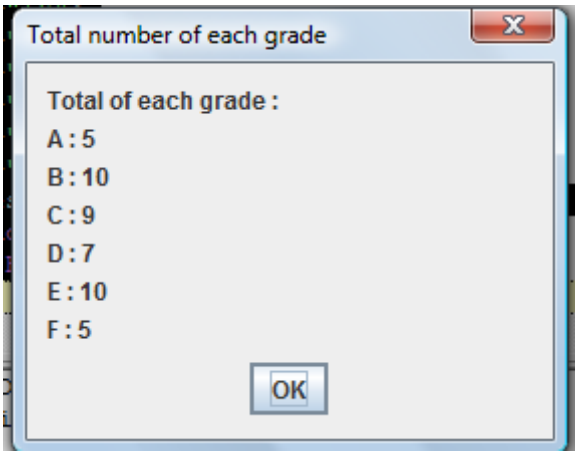
```
case the value of the variable:
If the variable has the value above carry out statements here
break; // end of structure
.................
default :
If non of the variable above is reached, carry out this statement
break; // end of structure
}
```

A sample program of grade counter of students are given

Program 4.4.1 BPIH4E10C.java : an example of switch-case statement

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E10C
{
// main method begins execution of Java application
public static void main( String args[] )
{
String s1="";
char grade;
boolean b=true;
int Anumber=0,Bnumber=0,Cnumber=0,Dnumber=0,Enumber=0,Fnumber=0;
while(b)
{
s1=JOptionPane.showInputDialog("Enter Student\'s grade: "); // read grade
grade=s1.charAt(0);
switch(grade)
{
case 'A': case 'a':
Anumber++;
break;
case 'B': case 'b':
Bnumber++;
break;
case 'C': case 'c':
Cnumber++;
break;
case 'D': case 'd':
Dnumber++;
break;
case 'E': case 'e':
Enumber++;
break;
case 'F': case 'f':
Fnumber++;
break;
default:
b=false;
break;
} // end of switch-case
}//end of while
String s="Total of each grade : \n";
```

```
s+="A : "+Anumber+"\n";
s+="B : "+Bnumber+"\n";
s+="C : "+Cnumber+"\n";
s+="D : "+Dnumber+"\n";
s+="E : "+Bnumber+"\n";
s+="F : "+Fnumber+"\n";
JOptionPane.showMessageDialog(null,s,"Total number of each grade",JOptionPane.PLAIN_MESSAGE);
//display x and y
    } // end method main
} // end class BPIH4E10C
```



If the same program is written by using if blocks instead of switch-case block:

Program 4.4.2 BPIH4E10C.java : switch-case statement replaced with if statement

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E10E
{
// main method begins execution of Java application
public static void main( String args[] )
{
String s1="";
char grade;
boolean b=true;
int Anumber=0,Bnumber=0,Cnumber=0,Dnumber=0,Enumber=0,Fnumber=0;
while(b)
{
s1=JOptionPane.showInputDialog("Enter Student\'s grade: "); // read grade
grade=s1.charAt(0);
if(grade=='A'||grade=='a')
{Anumber++;}
else if(grade=='B'||grade=='b')
{Bnumber++;}
else if(grade=='C'||grade=='c')
{Cnumber++;}
else if(grade=='D'||grade=='d')
{Dnumber++;}
else if(grade=='E'||grade=='e')
{Enumber++;}
else if(grade=='F'||grade=='f')
{Fnumber++;}
```

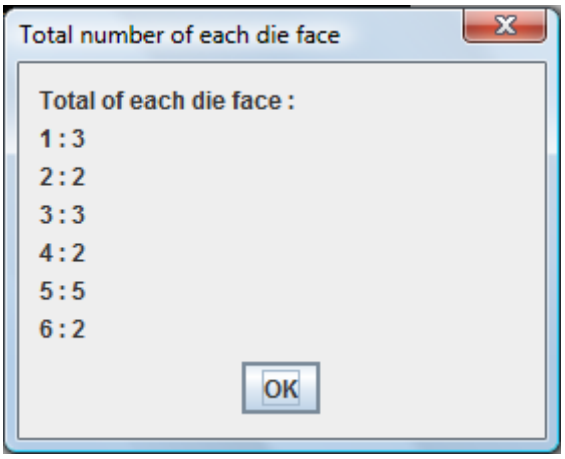
```
else
{b=false;}
}//end of while
String s="Total of each grade : \n";
s+="A : "+Anumber+"\n";
s+="B : "+Bnumber+"\n";
s+="C : "+Cnumber+"\n";
s+="D : "+Dnumber+"\n";
s+="E : "+Bnumber+"\n";
s+="F : "+Fnumber+"\n";
JOptionPane.showMessageDialog(null,s,"Total number of each grade",JOptionPane.PLAIN_MESSAGE); //display x
and y
    } // end method main
} // end class BPIH4E10E
```

The result is the same as the previous program. Now switch case sample will be given for a die throwing example. A die has 6 faces and we will be counting how many times each face is entered.

Program 4.4.3 BPIH4E11C.java : an example of switch-case statement

```
import javax.swing.*; // program uses class Scanner

public class BPIH4E11C
{
// main method begins execution of Java application
public static void main( String args[] )
{
//die throw counting
int die=1;
boolean b=true;
int one=0,two=0,three=0,four=0,five=0,six=0;
while(b)
{
die=Integer.parseInt(JOptionPane.showInputDialog("Enter number in die :")); // read grade
switch(die)
{
case 1:
one++;
break;
case 2:
two++;
break;
case 3:
three++;
break;
case 4:
four++;
break;
case 5:
five++;
break;
case 6:
six++;
break;
default:
```

```
b=false;
break;
} // end of switch-case
}//end of while
String s="Total of each die face : \n";
s+="1 : "+one+"\n";
s+="2 : "+two+"\n";
s+="3 : "+three+"\n";
s+="4 : "+four+"\n";
s+="5 : "+five+"\n";
s+="6 : "+six+"\n";
JOptionPane.showMessageDialog(null,s,"Total number of each die face",JOptionPane.PLAIN_MESSAGE);
//display x and y
    } // end method main
} // end class BPIH4E11C
```



WEEK 4 CLASS EXERSIZES (Wil be shown at the and of Lab class)

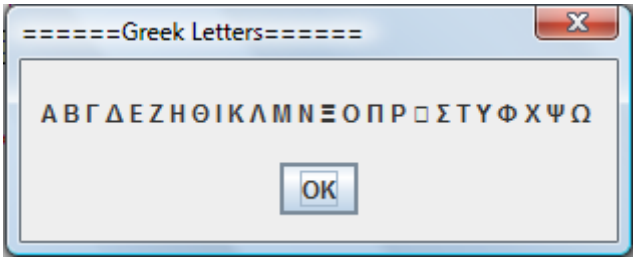
EX1) Write a program to calculate total and avarage of **odd numbers (1,3,5,7,9….99)** from 0 to 100 by using while loop.

EX2) Write a program to calculate the number $A=\sum_{n=1}^{10000}\frac{1}{n^2}$ by using a for loop. They claim that number A is equal to $\frac{\pi^2}{6}=1.6449$ check if it is true.

EX3) Write a program to printout x and y values for the function $y=f(x)=\frac{1}{x^2}$, x from 1 to 20 by using do-while loop

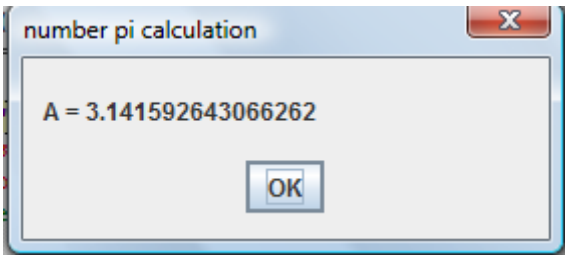
WEEK 4 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) Write a program to list capital greek letters in graphics environment as shown by using while loop



HW2) Write a program to calculate $A=2\prod_{k=1}^{\infty}\frac{(2k)^2}{(2k)^2-1}$ by using a for loop write the result. Note that infinite number is not possible in computer try different finite end numbers and check the result. Note : You can assign a second variable to write Formula easier for example b=(2.0*k)*(2.0*k);

```
import javax.swing.*; // program uses class Scanner

public class BPIH4HW2
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      String s;
      double n;
      double A=2.0;
      double b=1.0;
      for(double k=1.0;k < 100000000.0;k++)
      {
              b=(2.0*k)*(2.0*k);
      A*=b/(b-1);
      }
      s="A = "+A;
      System.out.println(s);
      JOptionPane.showMessageDialog(null,s,"number pi calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E1
```



HW3) A series Formula for ln(x) is given by

$y=\frac{x-1}{x+1}$ write program to calculate series by using a do-while loop. Verify your result. Note that equation can be further simplify if a second conversion z=y*y is made

```
import javax.swing.*; // program poweres class Scanner

public class BPIH4HW3
```

```
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
      // ln(x) = 1+y^2/3+y^4/5+y^6/7+...   y=(x-1)/(x+1)
      String s;
      double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
      double power=1;
      double ln=1;
      double y=(x-1)/(x+1);
      int plusminus=-1;
      double n=1;
      do
      {
              power*=y*y;
      ln+=power /(2.0*n+1.0);
      n++;
      } while(n<=10000000);
      ln*=2*y;
      s="ln("+x+") = "+ln;
      JOptionPane.showMessageDialog(null,s,"natural logarithm calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E1
```

COMPUTER PROGRAMMING I WEEK 6 PROBLEMS (ALGORITHMS)

Q1) Write the following code in the editor and run it. This is the serial solution of the exponential function

$$\exp(x) = \sum_{k=1}^{\infty} \frac{x^k}{k!}$$

```
import javax.swing.*;
public class exp
{
public static void main(String arg[])
{
//f(x)=exp(x)
double x=Double.parseDouble(JOptionPane.showInputDialog("x="));
double factorial=1;
double power=1;
double exp=0;
for(int k=1;k<200;i++)
{exp+=power/factorial;
 power*=x;
 factorial*=k;
}
System.out.println("exp("+x+")="+exp);

}
}
```

Q2 Write the following code in the editor and run it. . This is the serial solution of the natural logarithm (ln) function

$$\ln(x) = 2\sum_{k=1}^{\infty} \frac{1}{2k-1}\left(\frac{x-1}{x+1}\right)^{2k-1}$$

```
 import javax.swing.*;
```

```
public class ln
{
public static void main(String arg[])
{
//f(x)=ln(x)
double x=Double.parseDouble(JOptionPane.showInputDialog("x="));
double X=(x-1.0)/(x+1.0);
double ln=X;
double power=X;
double K=1;
for(int k=2;k<100;k++)
{   power*=X*X;
    K=2.0*k-1.0;
            ln+=power/K;
}
ln*=2.0;
System.out.println("ln("+x+")="+ln);
}
}
```

Q3 Write the following code in the editor and run it. This is the the serial solution of square root function

$$\sqrt{a} = e^{0.5*\ln(a)} = \sum_{k=1}^{\infty} \frac{[0.5\ln(a)]^k}{k!}$$

```
 import javax.swing.*;
public class sqrt
{
public static void main(String arg[])
{
//f(x)=ln(a)
double a=Double.parseDouble(JOptionPane.showInputDialog("x="));
// ln(a)
double X=(a-1.0)/(a+1);
double lna=X;
double power=X;
double K=1;
for(int k=2;k<100;k++)
{   power*=X*X;
    K=2.0*k-1.0;
            lna+=power/K;
}
lna*=2.0;
double x=0.5*lna;
// exp(x*ln(a))
double factorial=1;
power=1;
double sqrt=0;
for(int i=1;i<200;i++)
{sqrt+=power/factorial;
 power*=x;
 factorial*=i;
}

System.out.println("x = "+x+" sqrt("+a+")="+sqrt);
}
}
```

Q4) Write the following code in the editor and run it. This program calculates the roots of a quadratic polynomial

$$y = f(x) = ax^2 + bx + c = 0$$

$$\Delta = \sqrt{b^2 - 4ac}$$

$$x_{1,2} = \frac{-b \pm \Delta}{2a} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

In order to take the root we will use the serial solution given in example 3

```
 public class rootofquadraticpolynomial2
{
public static void main(String arg[])
{
//root of y=f(x)=a*x*x+b*x+c
double a,b,c;
a=2;
b=4;
c=-4;
double delta=b*b-4*a*c;
if(delta<0) System.out.println("Roots are complex");
else
{
//
// ln(delta)
double X=(delta-1.0)/(delta+1);
double lna=X;
double power=X;
double K=1;
for(int k=2;k<100;k++)
{   power*=X*X;
   K=2.0*k-1.0;
            lna+=power/K;
}
lna*=2.0;
double x=0.5*lna;
// exp(x*ln(a))
double factorial=1;
power=1;
double rootdelta=0;
for(int i=1;i<200;i++)
{rootdelta+=power/factorial;
 power*=x;
 factorial*=i;
}


double x1=(-b+rootdelta)/(2*a);
double x2=(-b-rootdelta)/(2*a);
System.out.println("delta="+delta+" root(delta)="+rootdelta+" x1 = "+x1+" x2 = "+x2);
}
}
}
```

Q5) Write the following code in the editor and run it. This program calculates the roots of a quadratic polynomial

$$y = f(x) = ax^2 + bx + c = 0$$

$$\Delta = b^2 - 4ac$$

$$x_{1,2} = \frac{-b \pm \sqrt{\Delta}}{2a} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

In order to take the root we will use Newton-raphson iterative formula:

$$x_{k+1} = x_k - \frac{x_k^{\ 2} - \Delta}{2x_k} \quad k = 1 \ldots n$$

$$\sqrt{\Delta} = x_{n+1}$$

in order to use this formula a first guess should be given

```
public class rootofquadraticpolynomial
{
public static void main(String arg[])
{
//root of y=f(x)=a*x*x+b*x+c
double a,b,c;
a=2;
b=4;
c=-4;
double delta=b*b-4*a*c;
if(delta<0) System.out.println("Roots are complex");
else
{
//calculation of roots of delta
//first gues for x
double x=delta/2.0;
for(int i=0;i<100;i++)
{x=x-(x*x-delta)/(2.0*x);}
double x1=(-b+x)/(2*a);
double x2=(-b-x)/(2*a);
System.out.println("delta="+delta+"root(delta)="+x+"x1="+x1+"x2="+x2);
}}}
```

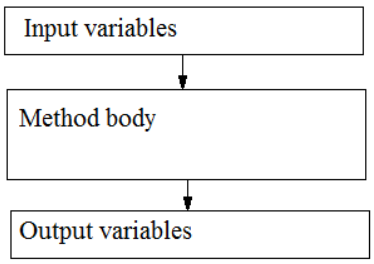
Q6) Develop a program to calculate the following series

$$\sin(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

5. METHODS AND INTRODUCTION TO CLASSES

5.1 METHODS

Methods and **variables** are the main components of java programs. The main componenets of java programs are classes. Methods are program components located in a class. Methods also have input and output variables which makes them to communucate with other program parts.



In general class has the following structure:
(**public**) (**static**) name_of_the_method(method input variables list)
{
Processes to be carried out in method
return Method output variable
}
If an example given:

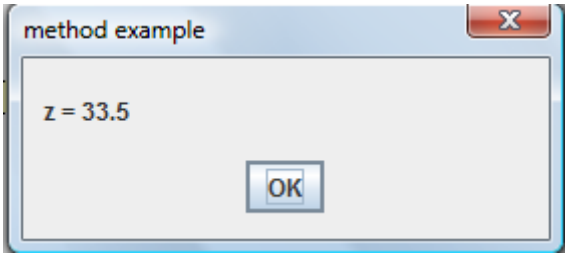
```
public static double z(int x,double y)
{
double w = 3.5;
double f ;
f = w*x*x+y;
return f;
}
```

In this method, x and y is integer type input variables, z method name and f is local variable used inside the method, and because f is used with return, it is also the output variable. The word public declare that the method can be accessed by any program. There is also word private and protected for limited acsess. The word static declares that method can be used as a complete programming piece. Methods can also exist as a part of a complete class, which some of the variable decleration is given in some other part of the class structure. In this case it is not a whole structure by itself, but a subpart of class stucture. When that is the case the word static is not used.

Program 5.1.1 BPIH5E1**: an example of static method call**

```
import javax.swing.*;

    public class BPIH5E1
    {
    public static double z(int x,double y)
    {
    double w = 3.5;
    double f ;
    f = w*x*x+y;
    return f;
    }

   public static void main( String args[] )
   {
```

```
            int i=3;
            double q=2.0;
              String s="z = "+z(i,q);
      JOptionPane.showMessageDialog(null,s,"method example",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH5E1
```



It should be note that in our example program two methods are declared one is **z** the other is **main**. Once a method is declared it can be called from the other programs as well (if it is declared as public)

Program 5.1.2 BPIH5E2**: an example of static method call**

```
  import javax.swing.*; // program uses class Scanner

    public class BPIH5E2
    {

   public static void main( String args[] )
   {
      int i=3;
      double q=2.0;
     String s="z = "+BPIH5E1.z(i,q);
     JOptionPane.showMessageDialog(null,s,"method example",JOptionPane.PLAIN_MESSAGE);
   } // end method main
} // end class BPIH4E2
```

The output of second example is the same as the first one. It is also used the method from the first example by declaring the name as :

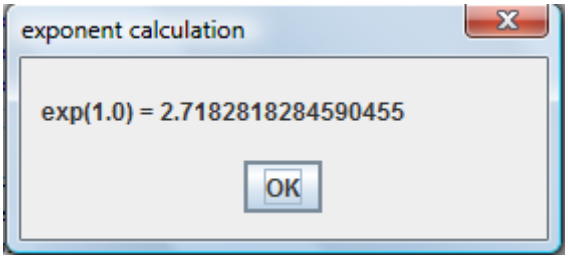
BPIH5E1.z(i,q);

BPIH5E1. in the declaration tells us that method is defined in class **BPIH5E1**. Now that we learn how to write a method, programs previously written as a whole in the main program such as BPIH4E8B can be written now by using a new method.

Program 5.1.3 BPIH5E3**: an example of static method call to calculate function exp**

```
import javax.swing. JOptionPane; // program poweres class Scanner

public class BPIH5E3
{
    // main method begins execution of Java application
    public static double exp(double x)
    {
      // number e = 2.718...
      // exp(x) = 1 + x /1! + x2 / 2! + x3 / 3! + x4 / 4! + ..
      double power=1;
      double exponent=1;
      double factorial=1;
```

```
        for(double sayi=1;sayi<=200;sayi++)
        {
        factorial*=sayi;
        power*=x;
        exponent+=power/factorial;
        }
      return exponent;
      }// end method exp

     public static void main( String args[] )
     {
        String s;
        double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: "));
       // read number
        s="exp("+x+") = "+exp(x);
        JOptionPane.showMessageDialog(null,s,"exponent
calculation",JOptionPane.PLAIN_MESSAGE);
     } // end method main
} // end class BPIH4E1
```



Observe the differences between program BPIH4E8B and BPIH5E3. Once a method such as exp written, it can be used in anywhere. In example BPIH5E4, exp is called from program BPIH5E3

Program 5.1.4 BPIH5E4.java **: an example of static method call to calculate function exp**

```
import javax.swing.*;

public class BPIH5E4
{
    // main method begins execution of Java application
    public static void main( String args[] )
    {
       String s;
       double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: "));
       // read number
       s="exp("+x+") = "+BPIH5E3.exp(x);
       JOptionPane.showMessageDialog(null,s,"exponent calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E4
```

It can also be noted in this point that **JOptionPane.showMessageDialog** is a method called **showMessageDialog** defined in class **JOptionPane.** Because this class definition is not located in our

directory directly, it should also be defined the location of the class. It is given with import statemet as: **import javax.swing.JOptionPane;**

There are other important libraries that we can utilises methods written in. For example, class Math is one of the important libraries for mathematical processes

Table 2.1 some of the static methods defined in class Math

Method	**Definition**	**Example**
abs(x)	Absolute value of double variable x	Math.abs(9.2) = 9.2 Math.abs(-9.2) = 9.2
ceil(x)	Converts double variable x to one upper integer value	Math.ceil(9.2) = 10 Math.ceil(-9.8) = -9
floor(x)	Converts double variable x to one lower integer value	Math.floor(9.2) = 9 Math.floor(-9.8) = -10
cos(x)	Cosine of double value x(in radian)	Math.cos(1.0)=0.54030230568
sin(x)	sine of double value x(in radian)	Math.sin(1.0)=0.8414709840709
tan(x)	Tangent of double value x(in radian)	Math.tan(1.0)=1.557040724655
acos(x)	Inverse cosine (in radian) of double value x	Math.acos(0.54030230568)=1.0
asin(x)	Inverse sine (in radian) of double value x	Math.asin(0.8414709840709)=1.0
atan(x) atan2(x,y)	Inverse Tangent (in radian) of double value x Inverse Tangent (in radian) of double value x and y as in cartesian coordinate system	4.0*Math.atan(1.0)=3.14159265359 4.0*Math.atan2(1.0,1.0)=3.14159265359
exp(x)	Powers of exponent number e(2.7182818284590452354)	Math.exp(1.0) =2.718281828459 Math.exp(2.0) =7.389056098931
log(x)	Natural logarithm for power e	Math.log(2.718281828459)=1.0
pow(x,y)	Y power of x	Math.pow(3,2)=9.0
max(x,y)	Maximum of x and y	Math.max(2.3,12.7)=12.7
min(x,y)	Minimum of x and y	Math.min(2.3,12.7)=2.3
random()	Random number between 0 and y	Math.random() = any double number between 0 and 1 in random

By using Math library, **Program 5.1.4** BPIH5E4.java can be re-written as :

Program 5.1.5 BPIH5E5.java **: an example of static method call to calculate function exp**

```
import javax.swing.JOptionPane; ; // program poweres class Scanner

public class BPIH5E5
{
    // main method begins execution of Java application
    public static void main( String args[] )
```

```
    {
      String s;
      double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
      s="exp("+x+") = "+Math.exp(x);
      JOptionPane.showMessageDialog(null,s,"exponent calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E5
```

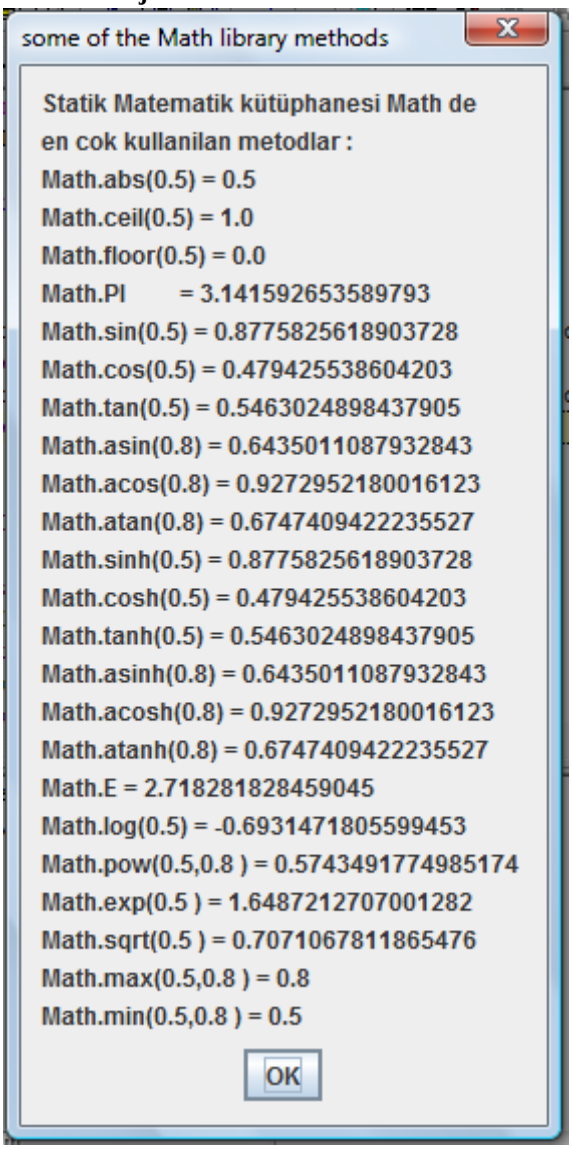
In the next example some of the Math library methods will be used.

Program 5.1.6 BPIH5E6.java **: an example of static method call to calculate function exp**

```
import javax.swing.JOptionPane;
public class BPIH5E6
{
    public static void main (String args[])
    {
    double x,y;
    String s="";
    x=Double.parseDouble(JOptionPane.showInputDialog
    ("enter a real number : "));
    y=Double.parseDouble(JOptionPane.showInputDialog
    ("enter a second real number giriniz : "));

    s +="Statik Matematik kütüphanesi Math de \n";
    s+="en cok kullanilan metodlar :\n" ;
    // Math.abs(3.2)=3.2   Math.abs(-3.2)=3.2
    s+="Math.abs("+x+") = "+Math.abs(x)+"\n";
    // Math.ceil(9.2)=10   Math.ceil(-9.8)=-9
    s+="Math.ceil("+x+") = "+Math.ceil(x)+"\n";
    // Math.floor(9.2)=9   Math.floor(-9.8)=-10
    s+="Math.floor("+x+") = "+Math.floor(x)+"\n";
    s += "Math.PI        = "+Math.PI+"\n";
    //trigonometric function
    s += "Math.sin("+x+") = "+Math.cos(x)+"\n";
    s += "Math.cos("+x+") = "+Math.sin(x)+"\n";
    s += "Math.tan("+x+") = "+Math.tan(x)+"\n";
    s += "Math.asin("+y+") = "+Math.acos(y)+"\n";
    s += "Math.acos("+y+") = "+Math.asin(y)+"\n";
    s += "Math.atan("+y+") = "+Math.atan(y)+"\n";
    //hyperbolic functions
    s += "Math.sinh("+x+") = "+Math.cos(x)+"\n";
    s += "Math.cosh("+x+") = "+Math.sin(x)+"\n";
    s += "Math.tanh("+x+") = "+Math.tan(x)+"\n";
    s += "Math.asinh("+y+") = "+Math.acos(y)+"\n";
    s += "Math.acosh("+y+") = "+Math.asin(y)+"\n";
    s += "Math.atanh("+y+") = "+Math.atan(y)+"\n";
    // Math.E       number e = 2.718281828...
    s += "Math.E = "+Math.E+"\n";
    s += "Math.log("+x+") = "+Math.log(x)+"\n";
    s += "Math.pow("+x+","+y+" ) = "+Math.pow(x,y)+"\n";
    s += "Math.exp("+x+" ) = "+Math.exp(x)+"\n";
    s += "Math.sqrt("+x+" ) = "+Math.sqrt(x)+"\n";
    // Math.max(x,y)   Math.min(x,y)
    s += "Math.max("+x+","+y+" ) = "+Math.max(x,y)+"\n";
    s += "Math.min("+x+","+y+" ) = "+Math.min(x,y)+"\n";
    JOptionPane.showMessageDialog(null,s,
    "some of the Math library methods",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);
    }
```

```
}
```

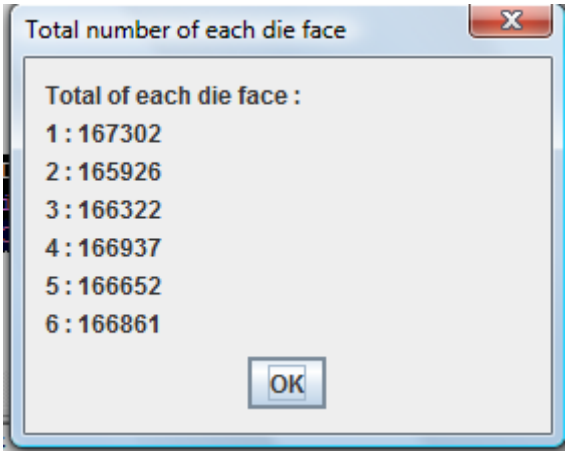
In order to show you the usage of random number, **Program 4.4.3 BPIH4E11C.java is** re-written with a throw_die method. This method will give a random number between 1 to 6 (to simulate die faces) and count how many times each faces comes. We tried 1million die throw in the program. Compare the code with **BPIH4E11C.java.**



Program 5.1.6 BPIH5E7.java **: an example of static method die throwing and counting**

```
import javax.swing.*; // program uses class Scanner

public class BPIH5E7
{
//die method
public static int throw_die()
{return (int)(Math.random()*6+1);}

// main method
public static void main( String args[] )
{
//die throw counting
int die=1;
```

```
boolean b=true;
int one=0,two=0,three=0,four=0,five=0,six=0;
int counter=0;
while(counter<1000000)
{
die=throw_die();
if(die==1) {one++;}
else if (die==2) {two++;}
else if (die==3) {three++;}
else if (die==4) {four++;}
else if (die==5) {five++;}
else if (die==6) {six++;}
counter++;
}//end of while
String s="Total of each die face : \n";
s+="1 : "+one+"\n";
s+="2 : "+two+"\n";
s+="3 : "+three+"\n";
s+="4 : "+four+"\n";
s+="5 : "+five+"\n";
s+="6 : "+six+"\n";
JOptionPane.showMessageDialog(null,s,"Total number of each die face",JOptionPane.PLAIN_MESSAGE); //display x and y
    } // end method main
} // end class BPIH4E11C
```



As seen from the results numbers are similar to each other due to statistical possibility of each face of the the is 1/6th. Theoretically each face should come 166667 times.

In Homework 3 of the last week natural logarithm calculation was given with the Formula :

$$y=\frac{x-1}{x+1}$$

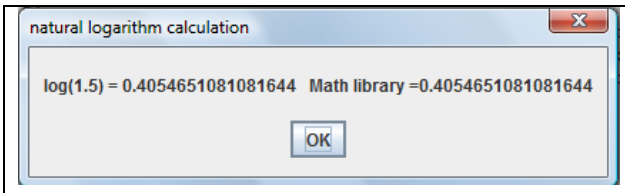
If this Formula written as a static method:

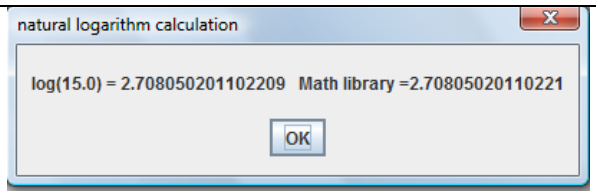
Program 5.1.7 BPIH5E13.java **: an example of static method natural logarithm series calculation**

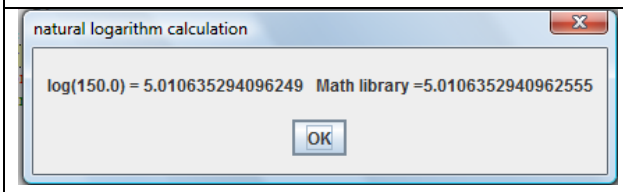
```
import javax.swing.*; // program poweres class Scanner

public class BPIH5E13
```

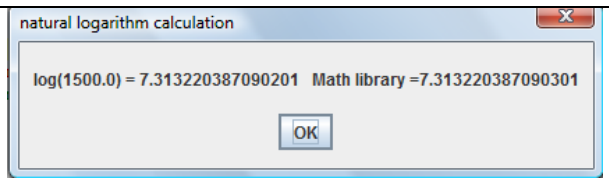
```
{
   public static double log(double x)
   {
     // ln(x) = 1+y^2/3+y^4/5+y^6/7+...   y=(x-1)/(x+1)
      double power=1;
      double ln=1;
      double y=(x-1)/(x+1);
      double n=1;
      do
      { power*=y*y;
      ln+=power /(2.0*n+1.0);
      n++;
      } while(n<=100000);
      ln*=2*y;
      return ln;
    }
   // main method begins execution of Java application
    public static void main( String args[] )
    {
      String s;
      double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
      s="log("+x+") = "+log(x)+"    "+"Math library ="+Math.log(x);
      System.out.println(s);
      JOptionPane.showMessageDialog(null,s,"natural logarithm calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class BPIH4E1
```

The results are also compared with Math library result to see accuracy of the series.









In order to improve the accuracy further, big digit number can be converted small digit number before the calculation is carried out. For example

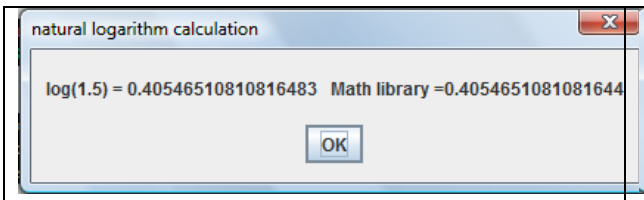
ln(123.456)=ln(1.23456*10^2)= ln(1.23456)+ln(10^2)= ln(1.23456)+2*ln(10)=
ln(1.23456)+2*2.302585092994046
ln(1234.56)= ln(1.23456)+3*2.302585092994046
ln(12345.6)= ln(1.23456)+4*2.302585092994046

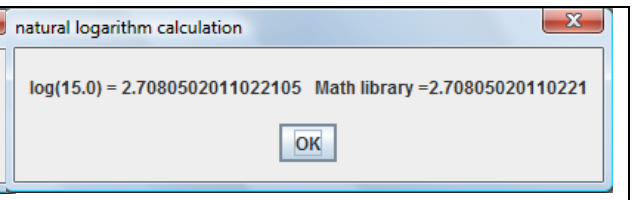
We will write a second version to utilise this property

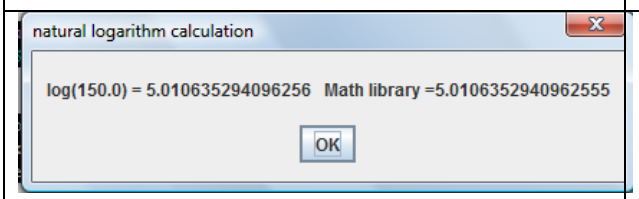
Program 5.1.8 BPIH5E14.java **: an example of static method natural logarithm series calculation**
With digit correction

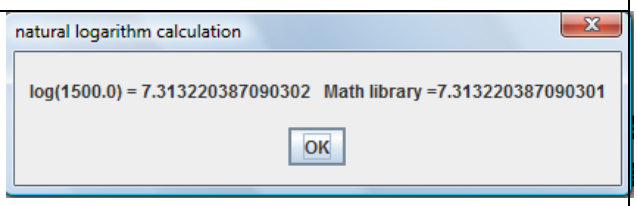
```
import javax.swing.*; // program poweres class Scanner
```

```
public class BPIH5E14
{
   public static double log(double x)
   {
            // ln(x) = 1+y^2/3+y^4/5+y^6/7+...   y=(x-1)/(x+1)
            //
            double ln10=2.302585092994046;
      int m=0;
      int j=1;
      double z=x;
      j=(int)z;
      while(j>0)
      {
            if(j>0) {x=z;m++;}
            z=x/10;
            j=(int)z;
            }
            x=z;

      double power=1;
      double ln=1;
      double y=(x-1)/(x+1);
      double n=1;
      do
      {
            power*=y*y;
      ln+=power /(2.0*n+1.0);
      n++;
      } while(n<=100000);
      ln*=2*y;
      ln+=m*ln10;
            return ln;
   }
   // main method begins execution of Java application
   public static void main( String args[] )
   {
      String s;
      double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
      s="log("+x+") = "+log(x)+"   "+"Math library ="+Math.log(x);
      JOptionPane.showMessageDialog(null,s,"natural logarithm calculation",JOptionPane.PLAIN_MESSAGE);
   } // end method main
} // end class BPIH4E1
```









The next example will plot the integer data by using a given character

Program 5.1.9 BPIH5E18.java **: an example of static method line plotting of an integer function**

```
import java.io.*;     //java girdi cikti sinifini cagir
import javax.swing.*;

class BPIH5E18
 {   public static String emptychar(int emptyspace,char x1)
    {String s="";
        for(int i=0;i<emptyspace;i++)
        {s+=" ";}
        s+=x1+"\n";
        return s;
    }

    public static int y(int x)
    {return x*x;}

    public static void  plot(int x1,int x2,char c1)
    { String s="";
      for(int x=x1;x<=x2;x++)
      {s+=emptychar(y(x),c1);}
      print(s);
    }

            public static void print(String s)
    {
              JOptionPane.showMessageDialog(null,s,"     ",JOptionPane.PLAIN_MESSAGE);
            }

            public static void main(String args[])
    {
       plot(-10,10,'*');
    }
 }
```

5.2 INTRODUCTION TO CLASS CONCEPT AND DYNAMIC METHOD

The basic structure of java programming language is classes. Classes behaves as a new type of variable, defined by the programmer. If a varible in a class is declared, it can be declared similar to declaration of variables

Classname objectname;

The variables declared in classes are called objects. Class based programming is called Object oriented programming. Classes can contain variables, objects of other classes and methods as a part of their structure. If you want to declare an object **new** structure is used. For example

book a=new book();

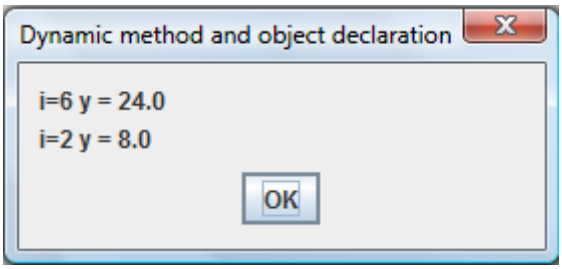
here book is the name of the class and a is the name of the object. In the example dynamic1.java, class name is dynamic1 Class has one variable (int i) and it is declared as 6. Class has also declaration of method **public double y(double x)**. In this method the word **static** is not used, and the class variable i is used inside as a part of the satetement directly. So this type of methods are dynamic and the value of function y is changed is variable i is changed in the program. In order to use class we declared object z as :

dynamic1 z=new dynamic();

After this decleration to call variable **int i** and method **public double y(double x)** is reached through object **z** as **z.i** and **z.y(2.0);**

Program 5.2.1 dynamic1.java : an example of dynamic method and object decleration

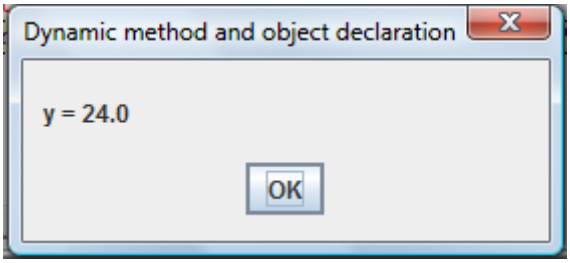
```
 import javax.swing.*; // program uses class Scanner
public class dynamic1
{
int  i=6; //class variable

public double y(double x)
{return x*x*i;}

public static void main(String arg[])
{
dynamic1 z=new dynamic1();
String s="i="+z.i+" y = "+z.y(2.0)+"\n";
z.i=2;
s+="i="+z.i+" y = "+z.y(2.0);
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}}
```



Once Class is declared, its variables and methods can be used from anywhere if they defined as public.

Program 5.2.2 dynamic2.java **: an example of dynamic method and object decleration**

```
 import javax.swing.*; // program uses class Scanner

public class dynamic3
{
public static void main(String arg[])
{
dynamic1 z=new dynamic1();
String s="i="+z.i+" y = "+z.y(2.0)+"\n";
z.i=2;
s+="i="+z.i+" y = "+z.y(2.0);
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}}
```



We can put methods to return string values into the classes. As well note that class dynamic10 Does not have a main method and it's called from class dynamic11

Program 5.2.3 dynamic10.java **: an example of dynamic method and object decleration**

```
public class dynamic10
{
int  i=6; //class variable
public double y(double x)
{return x*x*i;}
public String s1(int j,double x)
{ i=j;
  String s="i="+i+" y = "+y(x)+"\n";
 return s;
}}
```

Program 5.2.4 dynamic11.java **: an example of dynamic method and object decleration**

```
import javax.swing.*; // program uses class Scanner
public class dynamic11
{
public static void main(String arg[])
{
dynamic10 z=new dynamic10();
String s=z.s1(6,2.0)+z.s1(2,2.0);
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}}
```

The next program is the same as the previous, but **JOptionPane.showMessageDialog(null,s, "Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);** statement is given as a seperate method write. Not that the output type of this method is void. Void types does not return any value.

Program 5.2.4 dynamic12.java **: an example of dynamic method and object decleration**

```
import javax.swing.*; // program uses class Scanner

public class dynamic12
{
public static void write(String s)
{ JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}

public static void main(String arg[])
{
dynamic10 z=new dynamic10();
String s=z.s1(6,2.0)+z.s1(2,2.0);
write(s);
}
```

```
}
```

Dynamic methods in the same class can be called each other directly

Program 5.2.5 dynamic3.java **: an example of dynamic method and object decleration**

```
import javax.swing.*; // program uses class Scanner

public class dynamic3
{
int  i=6; //class variable

public double y(double x)
{return x*x*i;}

public double w(double x)
{double f=2.*x*x*y(x);
 return f;
}

public static void main(String arg[])
{
dynamic3 z=new dynamic3();
double xx=2.0;
String s="i="+z.i+" y = "+z.y(xx)+" w="+z.w(xx)+"\n";
z.i=2;
s+="i="+z.i+" y = "+z.y(xx)+" w="+z.w(xx);
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}
}
```

Program 5.2.6 dynamic4.java **: an example of dynamic method and object decleration**

```
port javax.swing.*; // program uses class Scanner

public class dynamic5
{
public static void main(String arg[])
{
dynamic3 z=new dynamic3();
double xx=2.0;
String s="i="+z.i+" y = "+z.y(xx)+" w="+z.w(xx)+"\n";
z.i=2;
s+="i="+z.i+" y = "+z.y(xx)+" w="+z.w(xx);
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}
}
```

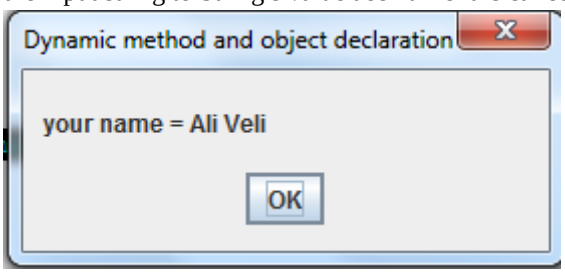
Class variable (variables defined in the class) can be accessed and changed by dynamic classes.

Program 5.2.7 dynamic5.java **: an example of dynamic method and object decleration**

```
 import javax.swing.*; // program uses class Scanner

public class dynamic5
{
```

```
String s="your name  = ";

public void y(String s1)
{s+=s1;}

public static void main(String arg[])
{
dynamic5 d1=new dynamic5();
d1.y("Ali");
d1.y(" Veli");
String s=d1.s;
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}
}
```

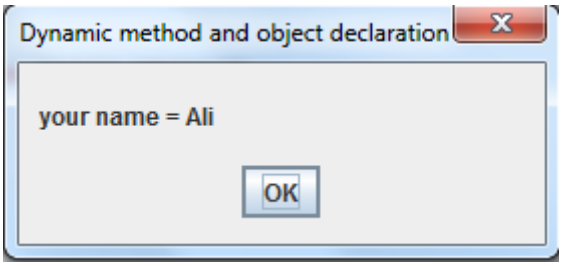
In this program object d1 defined and class variable s is set as “your name = “ String. Dynamic method added the input string to String s value aech time it is called.



When the value of the class variable is required the change a special method called constructor is used. This dynamic class is void type and the name is the same as the class name. The basic purpose of the metod is to load a value into the class variables.

Program 5.2.8 dynamic6.java **: an example of dynamicconstructor method, and object decleration**

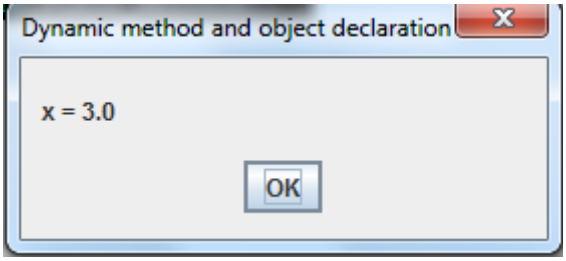
```
 import javax.swing.*; // program uses class Scanner

public class dynamic6
{
String s="your name = ";

public dynamic6(String s1)
{s+=s1;}

public static void main(String arg[])
{
dynamic6 d1=new dynamic6("Ali");
String s=d1.s;
JOptionPane.showMessageDialog(null,s,
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}
}
```



Another simple example of dynamic constructor method class variable decleration

Program 5.2.9 dynamic7.java **: an example of dynamicconstructor method, and object decleration**

```
import javax.swing.*; // program uses class Scanner

public class dynamic7
{
double x;

public dynamic7(double x1)
{x=x1;}

public String toString()
{String s="x = "+x;
return s;
}

public static void main(String arg[])
{
dynamic7 d1=new dynamic7(3.0);
JOptionPane.showMessageDialog(null,d1.toString(),
"Dynamic method and object declaration",JOptionPane.PLAIN_MESSAGE);
}
}
```



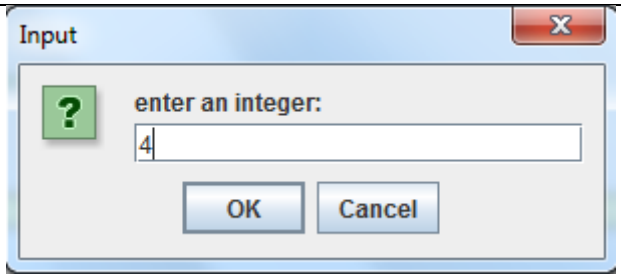
5.3 RECURSIVE METHODS

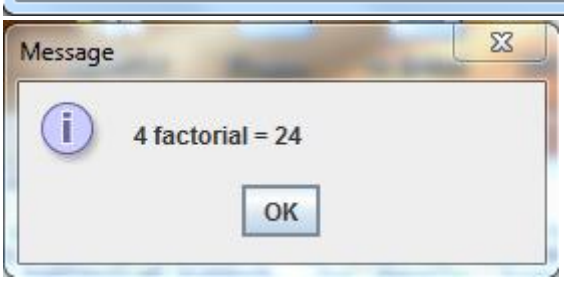
In java methods can call themselves. This will construct a new type of loop structure. This concept can be explained better with an examples. In the next problem factorial process is defined with a static method by using for loop structure.

Program 5.3.1 BPIH5E15.java **: static for loop factorial method**

```
import javax.swing.*;

public class forloop1
{
 public static long factorial(int x)
 {
 long faktoriyel=1;
 for(int sayi=1;sayi<=x;sayi++)
 { faktoriyel*=sayi;}
 return faktoriyel;
 }

 public static int read_int()
 {
     int number1=Integer.parseInt(JOptionPane.showInputDialog("enter an integer:"));
     return number1;
 }
 public static void print(String s)
```

```
 {JOptionPane.showMessageDialog(null,s); }

 // main method begins execution of Java application
 public static void main( String args[] )
 {
 int x=read_int();
 String s=x+" factorial =" + factorial(x);
 print(s);
 } // end of main method

} // end class forloop1
```





In the second example the same process will be achived by recursive loop

Program 5.3.2 recursiveloop1.java : static recursive loop factorial method

```
import javax.swing.*;

public class recursiveloop1
{
public static long factorial(int x)
{
if( x <= 1 )
return 1;
else
return x * factorial( x - 1);
}

  public static int read_int()
  {
       int number1=Integer.parseInt(JOptionPane.showInputDialog("enter an integer:"));
       return number1;
  }
  public static void print(String s)
  {JOptionPane.showMessageDialog(null,s); }

 // main method begins execution of Java application
 public static void main( String args[] )
 {
 int x=read_int();
 String s=x+" factorial =" + factorial(x);
```

```
 print(s);
 } // end of main method

} // end class recursiveloop1
```

As it is seen, method calls itself in here till x is equal to 1. Following up the recursive loops are relatively difficult and memory extensive process, therefore they are nott recommended unless a clear advantage is existed to use them.

5.4 USING METHODS WITH THE SAME NAME IN THE SAME CLASS (OVERLOADING)

We can use the same method name as long as the input parameters are different. If input parameters are the same eventhough return type is different or the variable name it is not allowed. For example:

public double methodB(double variableA)
or
public double methodB(double variableB)
or
public int methodB(double variableC)

can not be distinguished by java , but

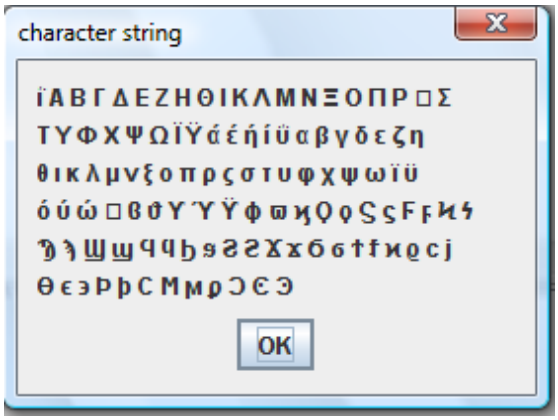
public double methodC(double variableA)
and
public double methodC(int variableB)
or
public double methodC(double variableA, double variableB)

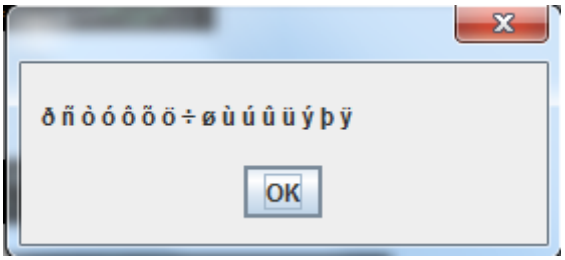
can be distungished by java as different methods
If an example is given, the following program has a static method to list a character list between given two characters. It also has two print methods. These methods are actually calls JOptionPane.showMessageDialog method inside, but one has only output in the input tream and the other has output and header in the input stream. Now system will have a choice between two print method.
public static void print(String s,String heading) and
public static void print(String s)

Program 5.4.1 overloadedmethods.java : using overloaded print method

```
 import java.io.*; //java girdi cikti sinifini cagir
import javax.swing.*;
class overloadedmethods
 {   public static String charstring(char x1,char x2)
    {String s="";
        int i=0;
        if(x2>x1)
        {
        for(char b=x1;b<=x2;b++)
        {s+=b+" ";
         i++;
        // a control-return in every 20 character
        if(i%20==0){s+="\n";}
```

```
        }
        }
        else
        {
        for(char b=x1;b>=x2;b--)
        {s+=b+" ";
        i++;
        if(i%20==0){s+="\n";}
        }
        }
     return s;
    }
    public static void print(String s,String heading)
    {
                JOptionPane.showMessageDialog(null,s,heading,JOptionPane.PLAIN_MESSAGE);
    }

              public static void print(String s)
            {
                JOptionPane.showMessageDialog(null,s,"     ",JOptionPane.PLAIN_MESSAGE);
              }
              public static void main(String args[])
    {
       print(charstring('\u0390','\u03FF'),"character string");
       print(charstring('\u00F0','\u00FF'));
    }
 }
```





WEEK 5 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) Write a static method to calculate the first number $A=6\sum_{n=1}^{10000}\frac{1}{n^2}$ by using a for loop. Then calculate pi by taken square root of A. $\pi=\sqrt{A}$ **Math.sqrt(A)**

EX2) write a program with static method that will calculate avarage of integer numbers starting at x1 and ending at x2

EX3) Write a program to printout x and y values for the function $y=f(x)=\frac{1}{x^2}$, x from 1 to 20 by using do-while loop . f(x) will be written as a static method

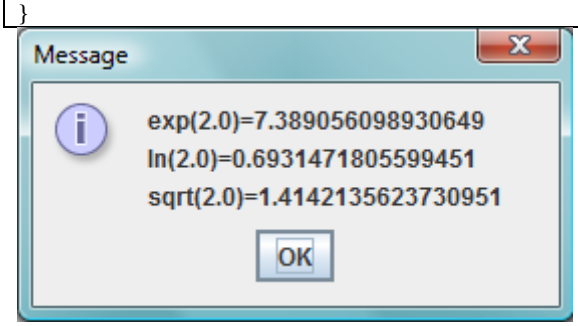
EX4) Fibonnacci numbers are defined with series $F_n=F_{n-1}+F_{n-2}$ where $F_0=0$ and $F_1=1$. Write a method with for loop to calculate Fibonnacci number **public static int F(int n)**
The Fibonacci sequence is named after Leonardo of Pisa, who was known as Fibonacci (a contraction of *filius Bonaccio*, "son of Bonaccio"). Fibonacci's 1202 book Liber Abaci introduced the sequence to Western European mathematics, although the sequence had been previously described in Indian mathematics.

EX5) Write the following code in the editor and run it. This contains the serial solution of the exponential function, logaritm function and power function in the **static method form**

$$\exp(x)=\sum_{k=1}^{\infty}\frac{x^k}{k!} \qquad \ln(x)=2\sum_{k=1}^{\infty}\frac{1}{2k-1}\left(\frac{x-1}{x+1}\right)^{2k-1} \qquad \sqrt{a}=e^{0.5*\ln(a)}=\exp(0.5*\ln(a))=\sum_{k=1}^{\infty}\frac{[0.5\ln(a)]^k}{k!}$$

```
 import javax.swing.*;
public class method1
{

// static method exp
public static double exp(double x)
{
double factorial=1;
double power=1;
double exp=0;
for(int k=1;k<200;k++)
{exp+=power/factorial;
 power*=x;
 factorial*=k;
}
return exp;
}
//static method ln
public static double ln(double x)
{
double X=(x-1.0)/(x+1.0);
double ln=X;
double power=X;
double K=1;
for(int k=2;k<100;k++)
{   power*=X*X;
   K=2.0*k-1.0;
            ln+=power/K;
}
ln*=2.0;
return ln;
}
//static method sqrt
public static double sqrt(double x)
```

```
{return pow(x,0.5);}

//static method pow
public static double pow(double a,double x)
{return exp(x*ln(a));}

//static method print
public static void print(String st1)
{ JOptionPane.showMessageDialog(null,st1);}

//method main
public static void main(String arg[])
{
//f(x)=exp(x)
double x=Double.parseDouble(JOptionPane.showInputDialog("x="));
String s="exp("+x+")="+exp(x)+"\n";
s+="ln("+x+")="+ln(x)+"\n";
s+="sqrt("+x+")="+sqrt(x);
print(s);
}

}
```



EX6) Write the following code in the editor and run it. This program calculates the roots of a quadratic polynomial in method form

$$y = f(x) = ax^2 + bx + c = 0$$

$$\Delta = \sqrt{b^2 - 4ac}$$

$$x_{1,2} = \frac{-b \pm \Delta}{2a} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

In order to take the root we will use the **methods sqrt** and **print** given in example 1

```
 import javax.swing.*;
public class method2
{
//method quadpolyroot1
public static double quadpolyroot1(double a,double b,double c)
{ //y=f(x)=a*x*x+b*x+c=0
double delta=b*b-4*a*c;
double x1=0;
if(delta<0) System.out.println("Roots are complex");
else  { x1=(-b+method1.sqrt(delta))/(2*a);}
return x1;
}

//method quadpolyroot2
public static double quadpolyroot2(double a,double b,double c)
{ //y=f(x)=a*x*x+b*x+c=0
double delta=b*b-4*a*c;
double x2=0;
if(delta<0) System.out.println("Roots are complex");
else { x2=(-b-method1.sqrt(delta))/(2*a);}
return x2;
```

```
}

//method input
public static double input(String s2)
{double y=Double.parseDouble(JOptionPane.showInputDialog(s2));
 return y;
}

//method main
public static void main(String arg[])
{
//f(x)=exp(x)
double a=input("y=f(x)=a*x*x+b*x+c=0, a = ");
double b=input("y=f(x)=a*x*x+b*x+c=0, b = ");
double c=input("y=f(x)=a*x*x+b*x+c=0, c = ");
String s="root1 = "+quadpolyroot1(a,b,c)+"\n";
s+="root2 = "+quadpolyroot2(a,b,c)+"\n";
method1.print(s);
}

}
```

EX7) Write the following code in the editor and run it. This contains the serial solution of the exponential function, logaritm function and power function similar to in the question 1 (serial solutions) , but the actual methods are given in the java library class **Math**

```
import javax.swing.*;
public class method3
{

//main method
public static void main(String arg[])
{
//f(x)=exp(x)
double x=method2.input("x=");
String s="exp("+x+")="+Math.exp(x)+"\n";
s+="ln("+x+")="+Math.log(x)+"\n";
s+="sqrt("+x+")="+Math.sqrt(x);
method1.print(s);
}

}
```

EX8) The program calculates sin(x) and cos(x) series as methods

$$\sin(x)=\sum_{k=0}^{\infty}(-1)^k\frac{x^{2k+1}}{(2k+1)!} \qquad \cos(x)=\sum_{k=0}^{\infty}(-1)^k\frac{x^{2k}}{(2k)!}$$

And compare the results with the Math library methods sin and cos write the code and run it.

```
import javax.swing.*;
public class method4
{

public static double sin(double x)
{int isaret=1;
 double pow=x;
 double fact=1;
 int n=1;
 double sin=0;
 for(int k=1;k<100;k++)
 {
```

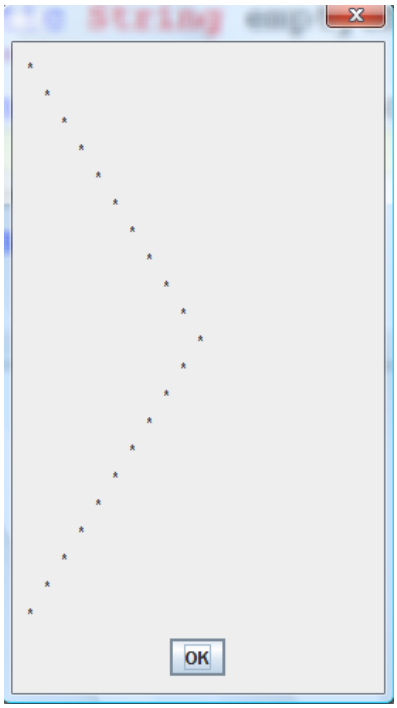
```
  sin+=pow/fact*isaret;
  pow*=x*x;
  n=2*k+1;
  fact*=n*(n-1);
  isaret*=-1;
 }
 return sin;
}

public static double cos(double x)
{int isaret=1;
 double pow=1;
 double fact=1;
 int n=1;
 double cos=1;
 for(int k=1;k<100;k++)
 {
  pow*=x*x;
  n=2*k;
  fact*=n*(n-1);
  isaret*=-1;
  cos+=pow/fact*isaret;
 }
 return cos;
}

public static void main(String arg[])
{
//f(x)=exp(x)
double x=method2.input("x=");
String s="sin("+x+")="+sin(x)+" Math library sin("+x+")="+Math.sin(x)+"\n";
s+="cos("+x+")="+cos(x)+" Math library cos("+x+")="+Math.sin(x)+"\n";
method1.print(s);
}
}
```

WEEK 5 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) Fibonnacci numbers are defined with series $F_n=F_{n-1}+F_{n-2}$ where $F_0=0$ and $F_1=1$. Write a method with **recursive loop** to calculate Fibonnacci number **public static int F(int n)**

HW2) Write a static method to calculate $A=2\prod_{k=1}^{\infty}\frac{(2k)^2}{(2k)^2-1}$ by using a for loop write the result. Note that infinite number is not possible in computer try different finite end numbers and check the result. Note : You can assign a second variable to write Formula easier for example b=(2.0*k)*(2.0*k);

HW3) Program 5.1.9 BPIH5E18.java describes the line plotting, by using this program and changing the function plot the following graphic output:



HW4) Program 5.1.6 BPIH5E7.java found when one die throws possibility of each number(one, two,three...). If we throws two dies in the same time, the total number will be (two,three,...twelve). Calcuate probability of each number. Use static methods.

6. CLASS STRUCTURE

6.1 GENERAL DEFINITION OF CLASS

Class concept and structure are used in all our programs so far, but we only barely mentioned its actual use and potential. The most important thing about classes, is that classes behaves as a new type of variable**.** The variables declared in classes are called objects. Classes are the general definitions of objects. Classes are the general definitions of objects, and objects are the variables defined out of objects. For example when we say pen, it is a class. When we mentione the red pen in Ali's pocket it becomes an object.

The general definition of the class is :

```
class class_name
{
//class variable list
variable type_1  variable_name_1;
variable type_2  variable_name_2;
.........
//class method list

variable_type mrthod_name(lists of method input variables)
{
//calculation and processes done by the method
return output_variable;
}
.............
}
```

Methods (or variables) does not have to be existed in the class definition, if they exist, then they are variables or methods belongs to that class. As the first class example, let us give a box clas that does not contain any methods in it. To define a box geometrically, we should define its three dimension, length, width and height

Program 6.1.1 box.java : a simple class application

```
class box
{
double length;
double width;
double height;
}
```

By this defination, a new type of variable, box, defined. In order to class this in a program,

box boxnumber1= new box();

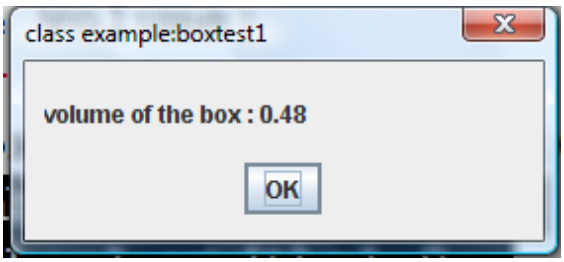
definition is used.

boxnumber1, is a variable(object) defined in class box type. This variable contains three subvariable length, width and height. We can assign values to sub variables in the program as

```
boxnumber1.length=1.0;
boxnumber1.width=1.2;
boxnumber1.height=0.4;
```

Program 6.1.2 boxtest1.java : a test program for class box : a simple class application

```
import javax.swing.JOptionPane;

class boxtest1
{
  public static void main(String args[])
  {
  double volume;
  String s="";
  box boxnumber1= new box();
  boxnumber1.length=1.0;
  boxnumber1.width=1.2;
  boxnumber1.height=0.4;
  volume=boxnumber1.length*boxnumber1.width*boxnumber1.height;
  s=s+"volume of the box : "+volume+"\n";
  JOptionPane.showMessageDialog(null,s,
    "class example:boxtest1",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```



The adress structure for a class variables will be as following :

Term	Computer main object adress	Object class variables adresses
Box1 **boxnumber1**;	null	
boxnumber1=new Box1();	Box1 **boxnumber1** adresss (for example: 23436A3F)	length width height (an adress for each variable, for example 13F23A3B,…)

In the second example, a method is also given in the box class

Program 6.1.3 box1.java : a simple class application

```
class box1
{
double length;
double width;
double height;

box1(double l,double w, double h)
{length=l;
width=w;
height=h;
}
}
```

This method box1 has the same name as the class(box1). This method is called the **constructor** method. By using this method, values to the class variables directly assigned to class variables.

box1 boxnumber2= new box1(1.0,1.2,.4);

Program 6.1.4 boxtest2.java : a test program for class box1 : a simple class application

```
import javax.swing.JOptionPane;
class boxtest2
{
  public static void main(String args[])
  {
  double volume;
  String s="";
  box1 boxnumber2= new box1(1.0,1.2,0.4);
  volume=boxnumber2.length*boxnumber2.width*boxnumber2.height;
  s=s+"volume of the box : "+volume+"\n";
  JOptionPane.showMessageDialog(null,s,
    "class example:boxtest1",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```

In the next example volume calculation of the box2 is taken inside the box class as another method. Note that this method use **class variables** and defined as a **dynamic method** (word static is not existed)

Program 6.1.5 box2.java : a simple class application

```
class box2
{
double length;
double width;
double height;

box2(double l,double w, double h)
{length=l;
width=w;
height=h;
}
double volume()
{return length*width*height;}
}
```

When box2 is called, method volume can be used to calculate volumeof the box

Program 6.1.6 boxtest3.java : a test program for class box2 : a simple class application

```
import javax.swing.JOptionPane;
class boxtest3
{
  public static void main(String args[])
  {
  double volume;
  String s="";
  box2 boxnumber3= new box2(1.0,1.2,0.4);
  volume=boxnumber3.volume();
  s=s+"volume of the box : "+volume+"\n";
  JOptionPane.showMessageDialog(null,s,
    "class example:boxtest3",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```

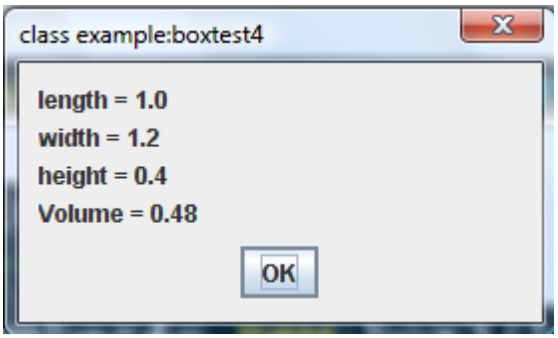
Also output process can be defined as another method inside the class

Program 6.1.7 box3.java : a simple class application

```
class box3
{
double length;
double width;
double height;

box3(double l,double w, double h)
{length=l;
width=w;
height=h;
}

double volume()
{return length*width*height;}

public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n";
  return s;
}

}
```

Program 6.1.8 boxtest4.java : a test program for class box3 : a simple class application

```
import javax.swing.JOptionPane;

class boxtest4
{
   public static void main(String args[])
   {
   double volume;
   String s="";
   box3 boxnumber4= new box3(1.0,1.2,0.4);
   JOptionPane.showMessageDialog(null,boxnumber4.toString(),
     "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
     System.exit(0);}
}
```



The most important thing about classes that they defined a new kind of variable and this variables, with all their methods and class variables can be used like simple variables (double,int, etc). In the next example constructor method of class box4 will be defined with box4 as an input variable. In this way a copy of a given box4 class variable can be obtained.

box4 firstbox= new box4(1.0,1.2,0.4);
box4 secondbox= new box4(firstbox);

Program 6.1.9 box4.java : a simple class application

```
class box4
{
double length;
double width;
double height;
box4(double l,double w, double h)
{length=l;
width=w;
height=h;
}
box4(box4 bi)
{length=bi.length;
width=bi.width;
height=bi.height;
}

double volume()
{return length*width*height;}

public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n";
  return s;
}

}
```

Program 6.1.10 boxtest5.java : a test program for class bo43 : a simple class application

```
import javax.swing.JOptionPane;

class boxtest5
{
```

```
  public static void main(String args[])
  {
  double volume;
  box4 firstbox= new box4(1.0,1.2,0.4);
  box4 secondbox= new box4(firstbox);
  String s="First box : \n"+firstbox.toString()+"\n";
  s+="Second box : \n"+secondbox.toString()+"\n";
  JOptionPane.showMessageDialog(null,s,
    "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```

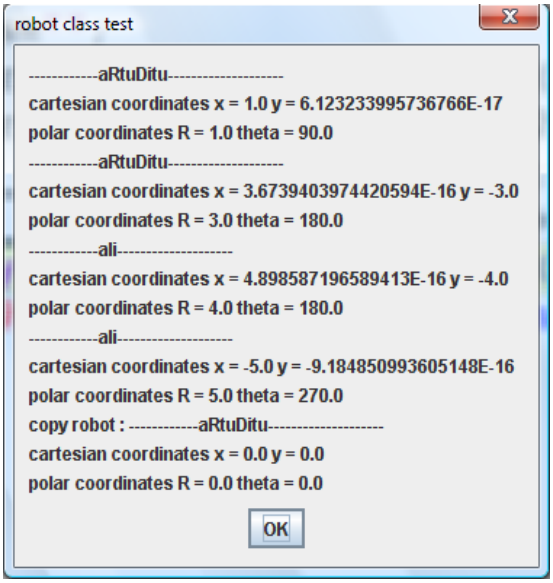
In the next example a robot class is defined. The coordinate of robot is given in the constructor method original coordinates and by using several methods turn to required direction and move in that direction.Fourdifferent overloaded contructor methods are given in this class. The fourth one is defined robot class as input variable

Program 6.1.11 robot.java : a simple class application

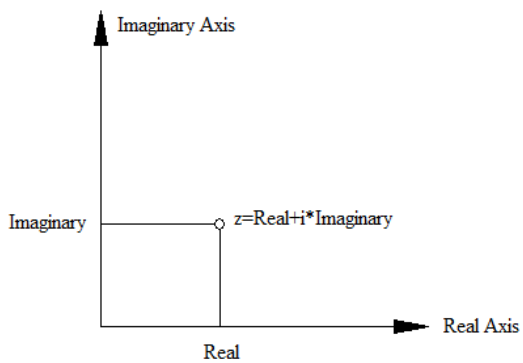
```
public class robot
{
 public String name;
 public double R;
 public double theta;
 public robot(String is) {R=0;theta=0;name=is;}
 public robot(String is,double Ri) {name=is;R=Ri;theta=0;}
 public robot(String is,double Ri,double theta_angle) {name=is;R=Ri;theta=theta_angle*Math.PI/180.0;}
 public robot(robot r1) {name=r1.name;R=r1.R;theta=r1.theta;}
 public void  turn_north() {theta=Math.PI/2.0;}
 public void  turn_south() {theta=3.0*Math.PI/2.0;}
 public void  turn_west() {theta=Math.PI;}
 public void  turn_east() {theta=Math.PI;}
 public void  turn(double angle) {theta+=angle*Math.PI/180.0;}
 public void  forward()  {R+=1;}
 public void  forward(double Ri)  {R+=Ri;}
 public void  backward()  {R-=1;}
 public void  backward(double Ri)  {R-=Ri;}
 public String toString()
 { String s="------------"+name+"--------------------\n";
   s+="cartesian coordinates x = "+R*Math.sin(theta)+" y = "+R*Math.cos(theta)+" \n";
   s+="polar coordinates R = "+R+" theta = "+theta*180/Math.PI+" \n";return s; }
}
```

Program 6.1.12 robottest.java : a test program for class robot : a simple class application

```
import javax.swing.*;public class robottest
{  public static void main(String args[])
   {robot R2D2=new robot("aRtuDitu");
    robot z2=new robot("ali",1.0,90.0);
    robot z3=new robot(R2D2);
    String s="";
    R2D2.turn_north();R2D2.forward();s+=R2D2.toString();
    R2D2.turn_west();R2D2.forward(2.0);s+=R2D2.toString();
    z2.turn_east();z2.forward(3.0);s+=z2.toString();
    z2.turn_south();z2.forward();s+=z2.toString();
    s+="copy robot : "+z3.toString();
    JOptionPane.showMessageDialog(null,s, "robot class test",JOptionPane.PLAIN_MESSAGE);
   }
}
```



In the next example, a complex variable is defined as a class. Complex numbers are expressed with two subnumber a real and an imaginary number. As [redacted], where $i = \sqrt{-1}$.



The first class is complex1. In this class only two variable, real and imaginary is defined.

Program 6.1.13 complex1.java : a simple class application

```
public class complex1
{ public double real;
  public double imaginary;
}
```

Program 6.1.14 complex1test.java : a test program for class complex1 : a simple class application

```
import javax.swing.*;
public class complex1test
{  public static void main(String args[])
   {complex1 z1=new complex1();
    z1.real=2.0;
    z1.imaginary=3.25;
```

```
    String s="complex number = "+z1.real+" + i*"+z1.imaginary+" \n";
    complex1 z2=new complex1();
    z2.real=1.1;
    z2.imaginary=2.0;
    s+="complex number = "+z2.real+" + i*"+z2.imaginary+" \n";
    complex1 z3=new complex1();
    z3.real=z1.real+z2.real;
    z3.imaginary=z1.imaginary+z2.imaginary;
    s+="total of complex number = "+z3.real+" + i*"+z3.imaginary+" \n";
   JOptionPane.showMessageDialog(null,s, "complex1 sınıfı testi",JOptionPane.PLAIN_MESSAGE);
   }
}
```

When class complex1 is used most of the process are still carried out in the main method. The purpose of using classes is to cary out most of the calculations in the class structure. In the next example constructors and methods for adding complex variables are defined in the class complex3.

Program 6.1.15 complex3.java : a simple class application

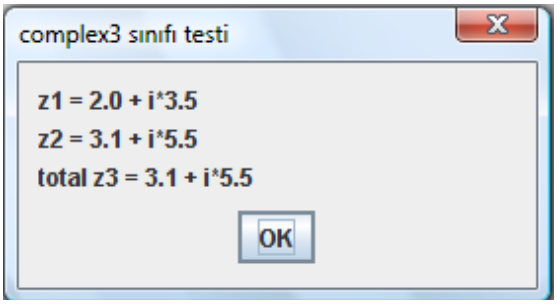
```
public class complex3
{ public double real;
  public double imaginary;
  public complex3(double g,double s)
  {real=g;
   imaginary=s;
  }
  public complex3(complex3 s1)
  {real=s1.real;
   imaginary=s1.imaginary;
  }
  public void add(complex3 s1)
  {real+=s1.real;
   imaginary+=s1.imaginary;
  }
  public  static complex3 add(complex3 s1,complex3 s2)
  {complex3 z3=new complex3((s1.real+s2.real),(s1.imaginary+s2.imaginary));
   return z3;
  }
   public String toString()
  { String s=+real+" + i*"+imaginary+" \n";
    return s;
  }
}
```

Two different test programs are given. In the first one, class complex3test, three complex class object z1,z2 and z3 crated. Z2 is copied into z3 and z1 is added into the z3, terefore we are ended up adding z2 and z3.

Program 6.1.16 complex3test.java : a simple class application

```
import javax.swing.*;
public class complex3test
{  public static void main(String args[])
   { complex3 z1=new complex3(2.0,3.5);
     complex3 z2=new complex3(1.1,2.0);
     complex3 z3=new complex3(z2);
     z3.add(z1);
     String s="z1 = "+z1.toString()+"z2 = "+z3.toString()+"total z3 = "+z3.toString();
```

```
  JOptionPane.showMessageDialog(null,s, "complex3 sınıfı testi",JOptionPane.PLAIN_MESSAGE);
  }
}
```



In the second class, complex3test1, two complex class object z1,z2 are crated. While a third onject z3 is created, the results of addition of z1 and z2 are direcl assigned as vale, by usingsattic add method.

Program 6.1.17 complex3test1.java : a simple class application

```
import javax.swing.*;
public class complex3test1
{  public static void main(String args[])
   { complex3 z1=new complex3(2.0,3.5);
    complex3 z2=new complex3(1.1,2.0);
    complex3 z3=complex3.add(z1,z2);
    String s="z1 = "+z1.toString()+"z2 = "+z3.toString()+"total z3 = "+z3.toString();
   JOptionPane.showMessageDialog(null,s, "complex3 sınıfı testi",JOptionPane.PLAIN_MESSAGE);
   }
}
```

The last example will be a company class. This class keep account balances when uy and sell products.

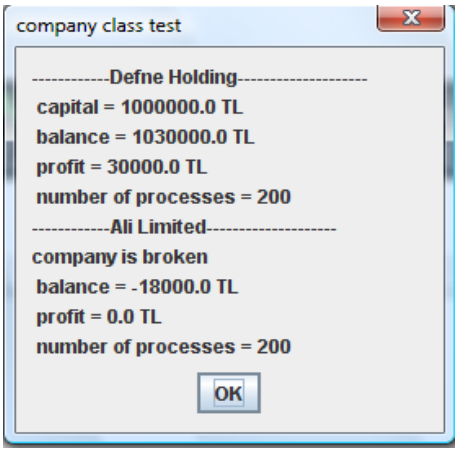
Program 6.1.18 company.java : a simple class application

```
public class company
{ public String name;
  public double capital,balance,profit;
  public int process;
  //this method establish the company

  //constructor method
  public company(String is,double Ri)
  {name=is;
  capital=Ri;
  balance=Ri;
  profit=0;
  process=0;
  }

  public void  buy(double x)
  {balance-=x;
   process++;
   if(balance<=0)
     {System.out.println("company is broken "+process);}
```

```
  }

  public void  sell(double x)
  {balance+=x;
   profit=balance-capital;
   process++;
  }

  public String acount()
  { String s="------------"+name+"--------------------\n";
  if(balance<=0)
     s+="company is broken \n";
  else
     s+=" capital = "+capital+" TL\n";
     s+=" balance = "+balance+" TL\n";
     s+=" profit = "+profit+" TL\n";
     s+=" number of processes = "+process+" \n";
  return s;
  }
}
```

Program 6.1.19 companytest.java : a simple class application test program for class company

```
import javax.swing.*;

public class companytest
{  public static void main(String args[])
   {
   company T=new company("Defne Holding",1.0e6);
   for(int i=0;i<100;i++) {T.buy(100);T.sell(400);}
   String s=T.acount();

   company A=new company("Ali Limited",1000.0);
   for(int i=0;i<100;i++) {A.buy(100);A.buy(90);}
   s+=A.acount();

   JOptionPane.showMessageDialog(null,s, "company class test",JOptionPane.PLAIN_MESSAGE);
   }}
```



6.2 USAGE OF TERM THIS

If the name of class variable and the name of a local variable in a method are the same, to diferentiate both, we can use term this to distinguish the class variables. For example in class box1 constructor method was given as:

```
box1(double l,double w, double h)
{length=l;
width=w;
height=h;
}
```

If instead of l,w and h variable names length, width and height is used they might confuse with class variables of the same name therefore, this method can be written as:

```
box1(double length ,double width, double height)
{this.length=length;
this.width=width;
this.height=height;
}
```

6.3 HIDING OF CLASS VARIABLES FROM OUTSIDE WORLD

Sometimes free access to the class variable outside of the class might not be desirable. For example if the class variable is carried out a bank başance it is desired to be reached to this variable only by the owner of the account if he/she known the access pasword. This kind of application should be done through methods (you can ask password and if password is correct, you can give the further information) . If class box5 is defined with private class variables

Program 6.3.1 box5.java : a simple class application with private class variables

```
class box5
{
private double length;
private double width;
private double height;

box5(double l,double w, double h)
{length=l;
width=w;
height=h;
}
// This methods are written to access private length,
// width and height variables
public double read_length() {return length;}
public double read_width() {return width;}
public double read_height() {return height;}
//This methods are written to change private length,
// width and height variables
public void write_length(double l) {length=l;}
public void write_width(double w) {width=w;}
public void write_height(double h) {height=h;}

box5(box5 bi)
{length=bi.length;
```

```
width=bi.width;
height=bi.height;
}

double volume()
{return length*width*height;}

public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n";
  return s;
}

}
```

We will try to access the class variables through the following test program :

Program 6.3.2 box5test.java : a simple class application test program for class boxtest5

```
import javax.swing.JOptionPane;

class boxtest6
{
   public static void main(String args[])
   {
   double volume;
   box5 firstbox= new box5(1.0,1.2,0.4);
   String s="First box : \n"+firstbox.toString()+"\n";
   firstbox.length=3.0;
   firstbox.width=2.0;
   firstbox.height=1.0;
   s+="value changed first box : \n"+firstbox.toString()+"\n";
   JOptionPane.showMessageDialog(null,s,
     "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
     System.exit(0);}
}
```

The result of compiling the program is :

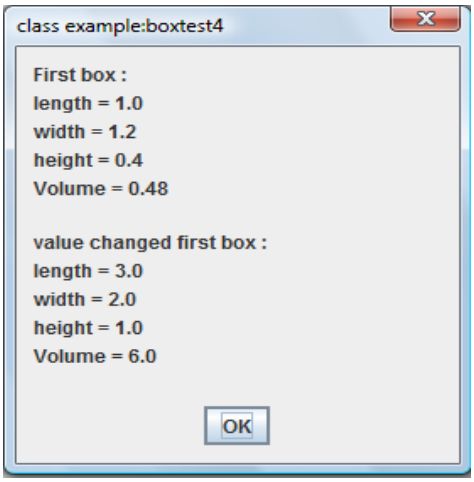
```
---------- Capture Output ----------
> "C:\turhan\java\bin\javac.exe" boxtest6.java
boxtest6.java:10: length has private access in box5
   firstbox.length=3.0;
          ^
boxtest6.java:11: width has private access in box5
   firstbox.width=2.0;
          ^
boxtest6.java:12: height has private access in box5
   firstbox.height=1.0;
          ^
3 errors
```

```
> Terminated with exit code 1.
```

The correct acccess tot the private variables will be as follows:

Program 6.3.4 box5testA.java : a simple class application test program for class company

```
import javax.swing.JOptionPane;

class boxtest6A
{
  public static void main(String args[])
  {
  double volume;
  box5 firstbox= new box5(1.0,1.2,0.4);
  String s="First box : \n"+firstbox.toString()+"\n";
  firstbox.write_length(3.0);
  firstbox.write_width(2.0);
  firstbox.write_height(1.0);
  s+="value changed first box : \n"+firstbox.toString()+"\n";
  JOptionPane.showMessageDialog(null,s,
    "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```



Another example is given as a bank account . In this class varaible accountno is private and can be reached only if a correct password is given. Class also has a counter to aalow only a given number of wrong password entry

Program 6.3.5 bankaccount.java

```
import javax.swing.*;
public class bankaccount
{
private String accountno;
private int code;
private int number;

public bankaccount(String an,int c)
{
accountno=an;
```

```
code=c;
number=0;
}
public String output()
{ while(number<3)
  {         number++;
  int c1=Integer.parseInt(JOptionPane.showInputDialog("enter the code = "));
  {if(c1==code) return accountno;
  }
  }
  return "wrong code is given";
}
}
```

Program 6.3.6 bankaccounttest1.java

```
public class bankaccounttest1
{
public static void main(String arg[])
{
 bankaccount isim=new bankaccount("123A4B5",12345);
 System.out.println(isim.output());
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" bankaccounttest1
123A4B5
> Terminated with exit code 0.
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" bankaccounttest1
wrong password is given 3 times system is locked
> Terminated with exit code 0.
```

WEEK 6 CLASS EXERSIZES (Wil be shown at the and of Lab class)

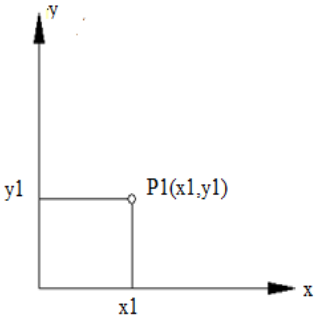
EX1) Multiplication of two complex number z1=Re1+i*Im1 and z2= z1=Re2+i*Im2 is defined as: z1*z2=(Re1*Re2-Im1*Im2) + i*(Re1*Im2+Re2*Im1) add two methods to complex3 class

public void multiply(complex3 s1)

public static complex3 multiply(complex3 s1,complex3 s2)

and test the methods through test programs

EX2) A point in two dimensional cartesian coordinate system is shown as x and y. Write a class to represent the point with output method to show the coordinates of the point



EX3) Add private String variable color of the box to box5 class and add up methods to Access the color and to change the color to to the class
public String read_color() {…….}
public void write_color(String col) {…….}

WEEK 6 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) A Force vector is defined with force components as F=Fx* i+Fy*j where i and j are unit vectors in x and y directions. Write a class to define a Force vector. In your definition also add a method to add two vectors. Note : rule of adding two vectors as F1=Fx1*i+Fy1*j and F2=Fx2*i+Fy2*j is F=F1+F2=(Fx1+Fx2)*i+(Fy1+Fy2)*j

HW2) Add the point method you developed in class exercise Ex2, a method to calculate the distance between to points. For the points P1(x1,y1) and P2(x2,y2) the distance between two points are :

$D=\sqrt{(x2-x1)^2+(y2-y1)^2}$ write a test program to test it

HW3) we would like to give robot we defined in robot class more direction flexibility so we will add
public void turn_north_east()
public void turn_north_west()
public void turn_south_east()
public void turn_south_west()
methods to class. Write the methods and test it with a test program

HW4) In the company class, we would like to give company possibility to borrow Money. Borrowed Money should be add to balance statement, when borrowed Money paid, so we will also define

public void borrowmoney(double x)
public void payborrowedmoney(double x)

After each 10 buying or selling 10% interest should be added to borrowed Money class variable (borrowed Money variable should be multiply with 1.1), but this amount will not be added to balance statement. Not that variable process counts total number of processes and when process%10 is zero, 10 buying or selling is completed regardless of the actual value of process variable
if((process%10)==0) {…….} is the statement to write

7 CLASS STRUCTURE: COMPOSITION AND INHERITANCE

7.1 COMPOSITION AND PROTECTED, PUBLIC AND PRIVATE

It is already mentioned that class structures are defined a new objects, similar but more complex compered to variable types. Since they are similar to variable, they can used as class variables in the definition of the other classes. Our first example consist of three classes point, circle and cylinder. As center of circle, point class is used and as base of cylinder circle class is used, so there is a chain relation between point, circle and cylinder classes.

Program 7.1.1 point.java : a simple class with protected class variables

```
public class point
{
   protected double x, y;

   public point(double a, double b)
   {enterpoint(a,b);}

   public point(point A)
   {enterpoint(A.readX(),A.readY());}

   public void enterpoint(double a, double b)
   { x=a;y=b;}

   public double readX()
   {return x;}

   public double readY()
   {return y;}

   public String toString()
   {return "["+x+","+y+"]";}

   public String name(){return "point";}

}
```

Program 7.1.2 circle.java : a simple class with protected class variables, use class point as class variable

```
public class circle
{
   protected double radius;
   protected point p;

   public circle()
   {   p=new point(0.0,0.0);
       enterradius(0.0);}

   public circle(double r, double a, double b)
   {  p=new point(a,b);
      enterradius(r);
   }
```

```
    public circle(double r, point n)
    {  p=new point(n.readX(),n.readY());
       enterradius(r);
    }

    public circle(circle c)
    {  p=new point(c.readpoint().readX(),c.readpoint().readY());
       enterradius(c.readradius());
    }

    public void enterradius(double r)
    {   if(r >= 0.0)
            radius=r;
        else
            radius=0.0;
    }

    public point readpoint()
    {return p;}

    public double readradius()
    {return radius;}

    public double area()
    {return Math.PI*radius*radius;}

    public String toString()
    { return "Center = "+"["+p.x+","+p.y+"]"+"; radius="+radius;}

    public String name(){return "circle";}
}
```

Program 7.1.3 cylinder.java : a simple class with protected class variables, use class circlr as class variable

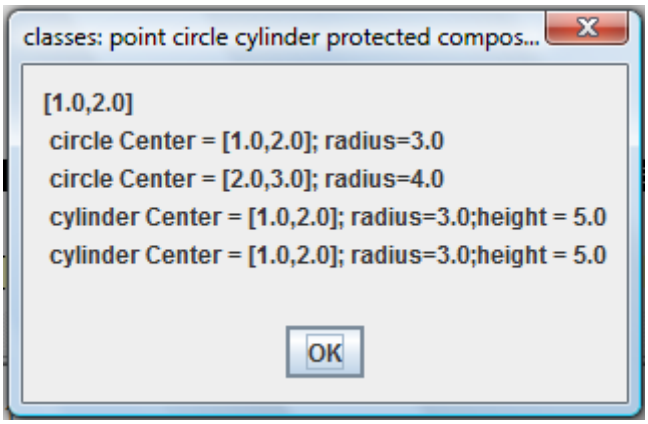
```
public class cylinder
 {
 protected double height;
 protected circle c;

 public cylinder()
 {c=new circle(0.0,0.0,0.0);
  enterheight(0);}

 public cylinder( double h,circle d)
 {c=new circle(d);
  height=h;
 }

 public cylinder( double h, double r, double a, double b)
 {c=new circle(r,a,b);
  height=h;
 }

 public void enterheight (double h )
 { if(h>=0) height=h;
   else height=0;
 }
```

```
 public double readheight(){return height;}

 public double area()
 {return 2*c.area()+2*Math.PI*c.radius*height;}

 public double volume(){return c.area()*height;}

 public String toString()
 {return c.toString()+";height = "+height;}

 public String name(){return "cylinder";}
}
```

Program 7.1.4 test program for classes point, circlr and cylinder with protected class variables

```
import javax.swing.*;

class compositiontest1
{
  public static void main(String args[])
  {
  point n=new point(1.0,2.0);
  circle d1=new circle(3.0,n);
  circle d2=new circle(4.0,2.0,3.0);
  cylinder s1=new cylinder(5.0,d1);
  cylinder s2=new cylinder(5.0,3.0,1.0,2.0);
  String s=n.toString()+"\n "+
  d1.name()+" "+d1.toString()+"\n "+
  d2.name()+" "+d2.toString()+"\n "+
  s1.name()+" "+s1.toString()+"\n "+
  s2.name()+" "+s2.toString()+"\n ";
  JOptionPane.showMessageDialog(null,s,
    "classes: point circle cylinder protected composition test",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```

As it is seen from the example , this classes are used inside another and has similar methods inside as well, They can be called as a family of classes composed from each other. Variables of these classes are also defined as **protected.** The word protected is more public than private but protected variables and methods are stil hard to Access, we usually use methods to get Access to protected variables. Output of the example class is :



Similar methods can be written with public variables as follows:

Program 7.15 point.java : a simple class with public class variables

```
public class point
{
   public double x, y;

   public point(double a, double b)
   {x=a;y=b;}

   public point(point A)
   {x=A.x;
    y=A.y;
   }

   public String toString()
   {return "["+x+","+y+"]";}

   public String name(){return "point";}

}
```

Program 7.1.6 circle.java : a simple class with public class variables, use class point as class variable

```
public class circle
{
   public double radius;
   public point p;

   public circle()
   {   p=new point(0.0,0.0);
       radius=0.0;}

   public circle(double r, double a, double b)
   {  p=new point(a,b);
      radius=r;
   }

   public circle(double r, point n)
   {  p=new point(n.x,n.y);
      radius=r;
   }

   public circle(circle c)
   {  p=new point(c.p.x,c.p.y);
      radius=c.radius;
   }

   public double area()
   {return Math.PI*radius*radius;}

   public String toString()
   { return "Center = "+"["+p.x+","+p.y+"]"+"; radius="+radius;}

   public String name(){return "circle";}
}
```

Program 7.1.7 cylinder.java : a simple class with public class variables, use class circle as class variable

```
public class cylinder
```

```
 {
 public double height;
 public circle c;

 public cylinder()
 {c=new circle(0.0,0.0,0.0);
  height=0.0;}

 public cylinder( double h,circle d)
 {c=new circle(d);
  height=h;
 }

 public cylinder( double h, double r, double a, double b)
 {c=new circle(r,a,b);
  height=h;
 }

 public double area()
 {return 2*c.area()+2*Math.PI*c.radius*height;}

 public double volume(){return c.area()*height;}

 public String toString()
 {return c.toString()+";height = "+height;}

 public String name(){return "cylinder";}
}
```

Results are the same as the previous program, so why we need protected or private, In the next example a class, bank_account, is given. This class can only be written as private due to security concern of electronic money theft from the acount.

Program 7.1.8 class bank_account with private class variables

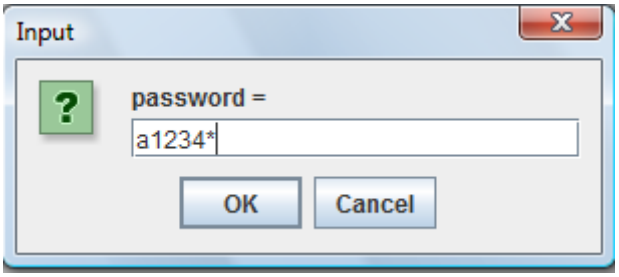
```
import javax.swing.*;

public class bank_account
{
 private String name;
 private int account_number;
 private String password;
 private int password_trial;
 private static int maximum_password_trial=3;
 private double account_balance;

public bank_account(String iname,int i_account_number,String i_password,double pgir)
 {name=iname;
  account_number=i_account_number;
  password=i_password;
  account_balance=pgir;
  password_trial=0;
 }

 public String read_account()
 { String s1="null";
   String password_read;
   int account_number_read=Integer.parseInt(JOptionPane.showInputDialog("account number = "));
```

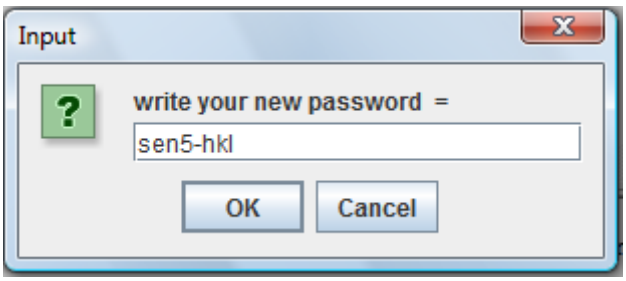
```
  while(password_trial<maximum_password_trial)
  {
  password_read=JOptionPane.showInputDialog("password = ");
  if(password_read.equals(password) ){password_trial=0;return write();}
  else{password_trial++;JOptionPane.showMessageDialog(null,"sorry wrong password, please try again","Wrong
password",JOptionPane.PLAIN_MESSAGE);}
  }
  return s1;
 }

 public String change_password()
 {String s1="change password";
  String password_read;
  while(password_trial<maximum_password_trial)
  {
  password_read=JOptionPane.showInputDialog("password = ");
  if(password_read.equals(password) )
  {
            String newpassword1=JOptionPane.showInputDialog("write your new password  = ");
            String newpassword2=JOptionPane.showInputDialog("re-write your new password  = ");
            if(newpassword1.equals(newpassword2))
            {password=newpassword2;s1="Password is changed sucsessfully";break;}
            else
            {JOptionPane.showMessageDialog(null,"new password1 and new password2 is not the same","wrong new
password",JOptionPane.PLAIN_MESSAGE);s1="Password change failuare";}
     password_trial=0;
  }
  else
  {JOptionPane.showMessageDialog(null,"sorry, wrong password please give it again","Wrong
password",JOptionPane.PLAIN_MESSAGE);
  password_trial++;s1="Password change failure";
  }
 }
 return s1;
 }

 private String write()
 {String s="Account name = "+name+"\n";
  s+="Account number = "+account_number+"\n";
  s+="Account balance = "+account_balance+" T.L.\n";
  return s;
 }

}
```

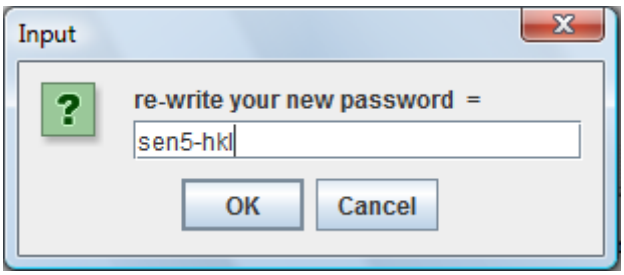
Program 7.1.9 A test program for class bank_account with private class variables

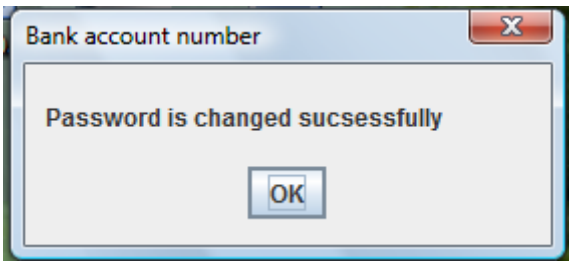
```
import javax.swing.*;

public class bank_account_test
{
public static void main(String arg[])
{
bank_account h1=new bank_account("ali veli",12345,"a1234*",100000.0);
JOptionPane.showMessageDialog(null,h1.change_password(),"Bank account number",JOptionPane.PLAIN_MESSAGE);
JOptionPane.showMessageDialog(null,h1.read_account(),"Bank account information",JOptionPane.PLAIN_MESSAGE);
}
}
```

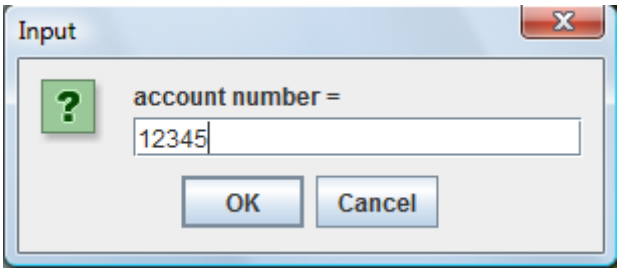
Class bank_account allows the password changes and graphical output of bank acount information. It has a sophisticated control. If wrong password is given some countable time, program stops.

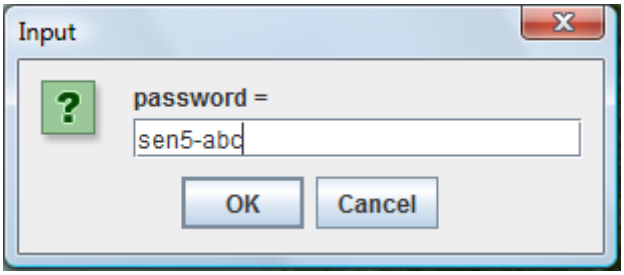


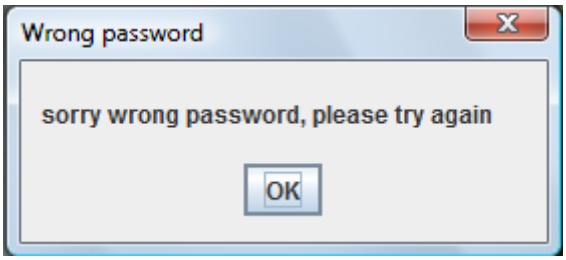


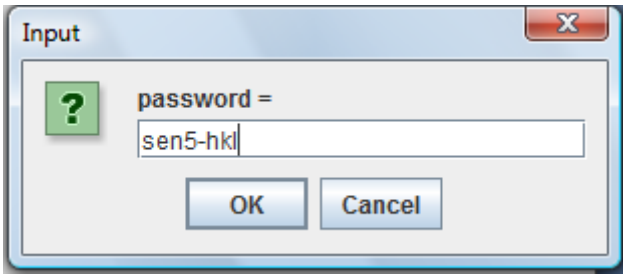


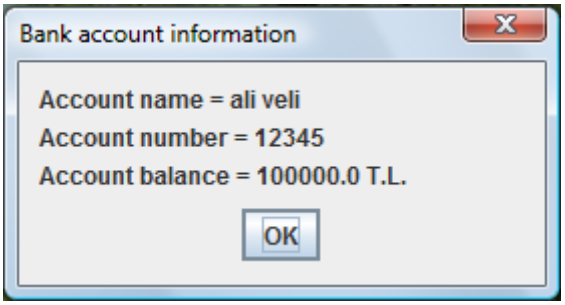












As it is seen from the example program secrecy of the code sometime is required. The case of protected and private is used for this kind of cases.

Now another group of classes are introduced as composition example, this case is presented only with public variables

Program 7.1.10 vehicle.java : a simple class with public class variables

```
import javax.swing.*;

public class vehicle
{
//valid vehicle types bus,truck,minitruck,minivan,auto
public String vehicletype;
public String brandName;
public double power; //KW olarak
public String color;
public double weight;          //yüksüz net ağırlık
```

```
//kurucu metodlar

public vehicle()
{vehicletype="not defined";
brandName="not defined";
power=0;
color="not defined";
weight=0;
}

public vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight)
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
   i_vehicletype.equals("minitruck") ||
   i_vehicletype.equals("bus")||
   i_vehicletype.equals("minibus")||
   i_vehicletype.equals("auto") ||
   i_vehicletype.equals("motorcycle") ||
   i_vehicletype.equals("bicycle") ||
   i_vehicletype.equals("off road vehicle")
   )
{vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
}

public vehicle(vehicle A)
{//gerekli tanımları yap
vehicletype=A.vehicletype;
brandName=A.brandName;
power=A.power;
color=A.color;
weight=A.weight;
}

public void define_vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight)
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
   i_vehicletype.equals("minitruck") ||
   i_vehicletype.equals("bus")||
   i_vehicletype.equals("minibus")||
   i_vehicletype.equals("auto") ||
   i_vehicletype.equals("motorcycle") ||
   i_vehicletype.equals("bicycle") ||
   i_vehicletype.equals("off road vehicle"))
   {vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
```

```
}

public String toString()
{String s1="";
s1+="Vehicle Type : "+vehicletype+"\n";
s1+="Brand Name : "+brandName+"\n";
s1+="Motor Power : "+power+" KW\n";
s1+="Color : "+color+"\n";
s1+="Weight : "+weight+" kg\n";
return s1;//gerekli tanımları yap
}

public void outputwindow()
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,toString(),s1, JOptionPane.PLAIN_MESSAGE);
}

}
```

Program 7.1.11 truck.java : a simple class with public class variables, use class vehicle as class variable

```
public class truck
{
vehicle v;

public truck(String i_marka,double i_motor,String i_renk,double i_agirlik)
{v=new vehicle("truck",i_marka,i_motor,i_renk,i_agirlik);}

public truck()
{ v=new vehicle();
  v.vehicletype="truck";
}
public truck(truck A)
{v=new vehicle(A.v);
 v.vehicletype="truck";
}

}
```

Program 7.1.12 bus.java : a simple class with public class variables, use class vehicle as class variable

```
public class bus extends vehicle
{
vehicle v;

public bus(String i_marka,double i_motor,String i_renk,double i_agirlik)
{v=new vehicle("bus",i_marka,i_motor,i_renk,i_agirlik);}

public bus()
{ v=new vehicle();
  v.vehicletype="bus";
}
public bus(bus A)
{ v=new vehicle(A.v);
 v.vehicletype="bus";

}
```

```
}
```

Program 7.1.13 minibus.java : a simple class with public class variables, use class vehicle as class variable

```
public class minibus
{
vehicle v;

public minibus(String i_marka,double i_motor,String i_renk,double i_agirlik)
{v=new vehicle("minibus",i_marka,i_motor,i_renk,i_agirlik);}

public minibus()
{ v=new vehicle();
  v.vehicletype="minibus";
}
public minibus(minibus A)
{v=new vehicle(A.v);
 v.vehicletype="minibus";
}

}
```

Program 7.1.14 A test program for classes vehicle,bus,truck and minibus

```
import javax.swing.*;
class vehicletesti1
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{
vehicle truck=new vehicle("truck","MAN",200.0,"red",15000);
vehicle bus=new vehicle("bus","Mersedes-Chrysler",300.0,"white",10000);
vehicle a=new vehicle();
String s=truck.toString()+"\n"+bus.toString()+"\n"+a.toString();
outputwindow(s);
}

}
```

Program 7.1.15 A test program for classes vehicle,bus,truck and minibus

```
import javax.swing.*;
class vehicletest2
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{
```

```
truck k1=new truck("MAN",200.0,"red",15000);
bus b1 =new bus("Mersedes-Chrysler",300.0,"white",10000);
minibus mb1=new minibus("BMC",70,"blue",1070);
String s=k1.toString()+"\n"+b1.toString()+"\n"+mb1.toString();
outputwindow(s);
}

}
```

Program 7.1.16 A test program for classes vehicle,bus,truck and minibus

```
import javax.swing.*;
class vehicletest3
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{

truck t1=new truck("MAN",200.0,"red",15000);
bus b1 =new bus("Mersedes-Chrysler",300.0,"white",10000);
minibus mb1=new minibus("BMC",70,"blue",1070);
vehicle v1=t1;
vehicle v2=b1;
vehicle v3=mb1;
String s=v1.toString()+"\n"+v2.toString()+"\n"+v3.toString();
outputwindow(s);
}

}
```

7.2 INHERITANCE

Instead of deriving a class by composition from another class through variable definition, new class can be **extended** from the previous class. To do this the word extend is used. If a class is extended from another class, it is similar to copying the whole codes (variables and methods) from the first class to the extended class. In order to check the basic mechanism, point-circle-cyclinder clas trio is re-defined by using inheritance.

Program 7.2.1 point.java : a simple class with protected class variables

```
public class point
{   protected double x, y;

    public point(double a, double b)
    {enterpoint(a,b);}

    public point(point A)
    {enterpoint(A.readX(),A.readY());}

    public void enterpoint(double a, double b)
    { x=a;y=b;}
```

```
   public double readX()
   {return x;}

   public double readY()
   {return y;}

   public String toString()
   {return "["+x+","+y+"]";}

   public String name(){return "point";}
}
```

Program 7.2.2 circle.java : a simple class with protected class variables, extended from class point

```
public class circle extends point
{
   protected double radius;
   public circle()
   {   super(0.0,0.0);
       enterradius(0.0);}

   public circle(double r, double a, double b)
   {  super(a,b);
      enterradius(r);
   }

   public circle(double r, point n)
   {  super(n.readX(),n.readY());
      enterradius(r);
   }

   public circle(circle c)
   {  super(c.x,c.y);
      enterradius(c.radius);
   }

   public void enterradius(double r)
   {   if(r >= 0.0)
           radius=r;
       else
           radius=0.0;
   }

   public double readradius()
   {return radius;}

   public double area()
   {return Math.PI*radius*radius;}

   public String toString()
   { return "Center = "+"["+x+","+y+"]"+"; radius="+radius;}

   public String name(){return "circle";}
}
```

Program 7.2.3 cylinder.java : a simple class with protected class variables, extended from class circle

```
public class cylinder extends circle
```

```
 {
   protected double height;

 public cylinder()
 { super(0.0,0.0,0.0);
   height=0.0;
 }

 public cylinder( double h,circle d)
 { super(d);
   height=h;
 }

 public cylinder( double h, double r, double a, double b)
 { super(r,a,b);
   height=h;
 }

 public void enterheight (double h )
 { if(h>=0) height=h;
   else height=0;
 }

 public double readheight(){return height;}

 public double area()
 {return 2*super.area()+2*Math.PI*radius*height;}

 public double volume(){return super.area()*height;}

 public String toString()
 {return super.toString()+";height = "+height;}

 public String name(){return "cylinder";}
}
```

Program 7.2.4 A test program for classes point,circle and cylinder (inheritance example)

```
import javax.swing.*;

class inheritancetest1
{
   public static void main(String args[])
   {
   point n=new point(1.0,2.0);
   circle d1=new circle(3.0,n);
   circle d2=new circle(4.0,2.0,3.0);
   cylinder s1=new cylinder(5.0,d1);
   cylinder s2=new cylinder(5.0,3.0,1.0,2.0);
   String s=n.toString()+"\n "+
   d1.name()+" "+d1.toString()+"\n "+
   d2.name()+" "+d2.toString()+"\n "+
   s1.name()+" "+s1.toString()+"\n "+
   s2.name()+" "+s2.toString()+"\n ";
   JOptionPane.showMessageDialog(null,s,
     "classes: point circle cylinder protected inheritance test",JOptionPane.PLAIN_MESSAGE);
     System.exit(0);}
}
```

One thing is to note at in an extended class, if a contructor method of parent class is to be used, it can be used by using word **super.**

Another inheritance test program group is vehicle-truck-bus-minibus group

Program 7.2.5 vehicle.java : a simple class with public class variables

```
import javax.swing.*;

public class vehicle
{
//valid vehicle types bus,truck,minitruck,minivan,auto
public String vehicletype;
public String brandName;
public double power; //KW olarak
public String color;
public double weight;           //yüksüz net ağırlık
//kurucu metodlar

public vehicle()
{vehicletype="not defined";
brandName="not defined";
power=0;
color="not defined";
weight=0;
}

public vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight)
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
   i_vehicletype.equals("minitruck") ||
   i_vehicletype.equals("bus")||
   i_vehicletype.equals("minibus")||
   i_vehicletype.equals("auto") ||
   i_vehicletype.equals("motorcycle") ||
   i_vehicletype.equals("bicycle") ||
   i_vehicletype.equals("off road vehicle")
   )
{vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
}

public vehicle(vehicle A)
{//gerekli tanımları yap
vehicletype=A.vehicletype;
brandName=A.brandName;
power=A.power;
color=A.color;
weight=A.weight;
}

public void define_vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight)
```

```
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
   i_vehicletype.equals("minitruck") ||
   i_vehicletype.equals("bus")||
   i_vehicletype.equals("minibus")||
   i_vehicletype.equals("auto") ||
   i_vehicletype.equals("motorcycle") ||
   i_vehicletype.equals("bicycle") ||
   i_vehicletype.equals("off road vehicle"))
   {vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
}

public String toString()
{String s1="";
s1+="Vehicle Type : "+vehicletype+"\n";
s1+="Brand Name : "+brandName+"\n";
s1+="Motor Power : "+power+" KW\n";
s1+="Color : "+color+"\n";
s1+="Weight : "+weight+" kg\n";
return s1;//gerekli tanımları yap
}

public void outputwindow()
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,toString(),s1, JOptionPane.PLAIN_MESSAGE);
}

}
```

Program 7.2.6 bus.java : a simple class with public class variables, extended from class vehicle

```
public class bus extends vehicle
{
public bus(String i_marka,double i_motor,String i_renk,double i_agirlik)
{super("bus",i_marka,i_motor,i_renk,i_agirlik);}
public bus()
{ super();
  vehicletype="bus";
}
public bus(bus A)
{super(A);
 vehicletype="bus";
}

}
```

Program 7.2.7 truck.java : a simple class with public class variables, extended from class vehicle

```
public class truck extends vehicle
{
public truck(String i_marka,double i_motor,String i_renk,double i_agirlik)
```

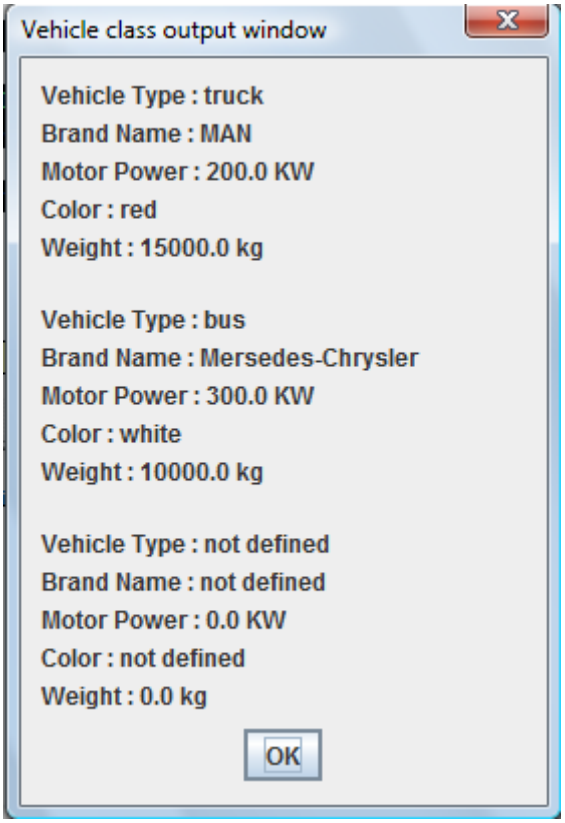
```
{super("truck",i_marka,i_motor,i_renk,i_agirlik);}
public truck()
{ super();
  vehicletype="truck";
}
public truck(truck A)
{super(A);
 vehicletype="truck";
}

}
```

Program 7.2.8 minibus.java : a simple class with public class variables, extended from class vehicle

```
public class minibus extends vehicle
{
public minibus(String i_marka,double i_motor,String i_renk,double i_agirlik)
{super("minibus",i_marka,i_motor,i_renk,i_agirlik);}
public minibus()
{ super();
  vehicletype="minibus";
}
public minibus(minibus A)
{super(A);
 vehicletype="minibus";
}

}
```

Program 7.2.9 A test program for classes vehicle,bus,truck and minibus

```
import javax.swing.*;
class vehicletesti1
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{
vehicle truck=new vehicle("truck","MAN",200.0,"red",15000);
vehicle bus=new vehicle("bus","Mersedes-Chrysler",300.0,"white",10000);
vehicle a=new vehicle();
String s=truck.toString()+"\n"+bus.toString()+"\n"+a.toString();
outputwindow(s);
}

}
```



In the next example problem **indirect referencing** is shown. If classes derived from the same parent class, they can be referenced, indirectly, through the parent (root) class. In c and c++ languages indirect referencing applied by using reference concept. What we are establishing in here is the same thing, but references can not be adressed directly due to security reasıns. Instead they can be reached by indirect class referencing mechanism.

Program 7.2.10 A test program for classes vehicle,bus,truck and minibus

```
import javax.swing.*;
class vehicletest3
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{

truck t1=new truck("MAN",200.0,"red",15000);
bus b1 =new bus("Mersedes-Chrysler",300.0,"white",10000);
minibus mb1=new minibus("BMC",70,"blue",1070);
vehicle v1=t1;
vehicle v2=b1;
vehicle v3=mb1;
String s=v1.toString()+"\n"+v2.toString()+"\n"+v3.toString();
outputwindow(s);
}
```

```
}
```

Output of this program is the same as the previous output.

As a third example of inherited class chain book-fiction-poetry-encyclopedia group is defined

Program 7.2.11 book.java : a simple class with public class variables

```
public class book
{
String name;
String author;
String clasification;
String language;

public book(String kname,String kauthor,String kclasification,String klanguage)
{name=kname;
 author=kauthor;
 clasification=kclasification;
 language=klanguage;
}

public book(book k)
{name=k.name;
 author=k.author;
 clasification=k.clasification;
 language=k.language;
}

public String toString()
{
String s="name = "+name+"\n";
s+="author = "+author+"\n";
s+="book clasification = "+clasification+"\n";
s+="language = "+language+"\n";
return s;
}

}
```

Program 7.2.12 fiction.java : a simple class with public class variables, extended from class book

```
public class fiction extends book
{
public fiction(String kisim,String kyazar,String klanguage)
{ super(kisim,kyazar,"fiction",klanguage);}

}
```

Program 7.2.13 poetry.java : a simple class with public class variables, extended from class book

```
public class poetry extends book
{
public poetry(String kname,String kauthor,String klanguage)
{ super(kname,kauthor,"poetry",klanguage);}
}
```

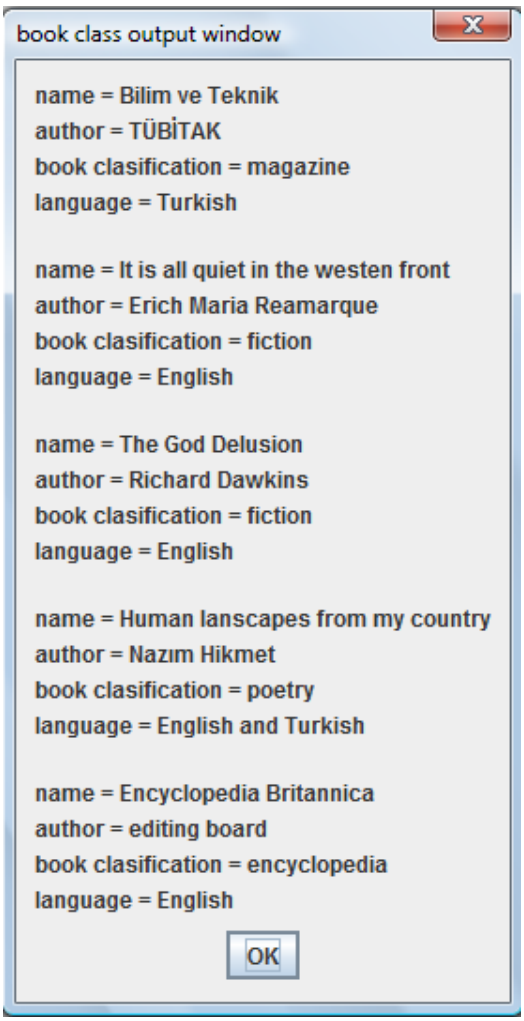
Program 7.2.14 encyclopedia.java : a simple class with public class variables, extended from class book

```
public class encyclopedia extends book
```

```
{
public encyclopedia(String kname,String klanguage)
{ super(kname,"editing board","encyclopedia",klanguage);}
}
```

Program 7.2.15 A test program for classes book,fiction,poetry,encyclopedia

```
import javax.swing.*;
class booktest1
{
public static void outputwindow(String s)
{
String s1="book class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}
public static void main(String args[])
{
book magazine=new book("Bilim ve Teknik","TÜBİTAK","magazine","Turkish");
fiction r1=new fiction("It is all quiet in the westen front ","Erich Maria Reamarque","English");
book r2=new book("The God Delusion","Richard Dawkins","fiction","English");
poetry  s1=new poetry("Human lanscapes from my country","Nazım Hikmet","English and Turkish");
encyclopedia a1=new encyclopedia("Encyclopedia Britannica","English");
String s=magazine.toString()+"\n"+r1.toString()+"\n"+r2.toString()+"\n"+s1.toString()+"\n"+a1.toString();
outputwindow(s);
}
```



7.3 SOME REAL CLASS EXAMPLES

Some classes can be utilised very frequently in engineering applications. One example could be a complex class. The class presented here is considered any kind of possible use and mathematical proceses of complex class, therefore code became a bit excessive, But the it can be used in anytime a complex number calculation is needed

Program 7.3.1 complex.java : a class with public class variables to be used for actual applications

```
import java.io.*;
import java.text.*;
import java.util.Locale;
import java.util.*;

import javax.swing.*; // giriş çıkış

//  class complex
//  complex number abstraction
//  data areas

class complex {
            public double   nreal;
   public double   nimag;
   // constructors
public complex()
{
nreal=0;
nimag=0;
}

public complex(double nre,double nim)
{
nreal=nre;
nimag=nim;
}

public complex(int nre,int nim)
{
nreal=(double)nre;
nimag=(double)nim;
}

public complex(double numer)
{
nreal=numer;
nimag=0;
}

public complex(complex value )
{
nreal=value.real();
nimag=value.imaginary();
}

// accessor functions
```

```
public double real()
{
//return real part of the complex number
return nreal;
}

public double imaginary()
{
//return imaginary part of the complex number
return nimag;
}

public double R()
{
// Return radius of polar coordinate equivalent of complex number
return Math.sqrt(nreal*nreal+nimag*nimag);
}

public double theta()
{
// Return angle of polar coordinate equivalent of complex number in radian
return Math.atan2(nimag,nreal);
}

public double dtheta()
{
// Return angle of polar coordinate equivalent of complex number in radian
return Math.atan2(nimag,nreal)*180.0/Math.PI;
}

// assignments
public void assign(complex right)
{
nreal=right.real();
nimag=right.imaginary();
}

public void add(complex  right)
{
nimag = nimag + right.imaginary();
nreal = nreal + right.real();
}

public void substract(complex right)
{
nimag = nimag - right.imaginary();
nreal = nreal - right.real();
}

public void multiply(complex right )
{
nreal = nreal*right.real() - nimag*right.imaginary();
nimag = nreal*right.imaginary() +  nimag*right.real();
}

public void divide(complex right )
{
double a=nreal*nreal+nimag*nimag;
```

```
nreal = ( nreal*right.real() +  nimag*right.imaginary())/a;
nimag = (-nreal*right.imaginary() +  nimag*right.real())/a;
}

public static complex add(complex left, complex right)
{ // return sum of two complex numbers
 double r1=(left.real() + right.real());
 double i1=(left.imaginary() + right.imaginary());
 complex result;
 result=new complex(r1,i1);
 return result;
}

public static complex substract(complex left, complex right)
{ // return substraction of two complex numbers
 complex result;
 result=new complex((left.real() - right.real()),
              (left.imaginary() - right.imaginary()));
 return result;
}

public static complex multiply(complex left, complex right)
{ // return multiplication of two complex numbers
 complex result;
 result=new complex
 ((left.real()*right.real() - left.imaginary()*right.imaginary()),
  (left.real()*right.imaginary() + left.imaginary()*right.real()));
return result;
}

public static complex divide(complex left, complex  right)
{ // return division of two complex numbers
 double a=right.real()*right.real()+right.imaginary()*right.imaginary();
 complex result;
 result=new complex
 ((left.real()*right.real()  + left.imaginary()*right.imaginary())/a,
  (-left.real()*right.imaginary() + left.imaginary()*right.real())/a);
 return result;
}

public static complex pow(complex left, double right)
{ // return sum of two complex numbers
 double Rad,th;
 Rad=Math.pow(left.R(),right);
 th=right*left.theta();
 complex result;
 result =new complex((Rad*Math.cos(th) ),
              (Rad*Math.sin(th) )  );
 return result;
}
public static complex sqrt(complex left)
{
return pow(left,0.5);
}

public static double abs(complex left)
{
return left.R();
```

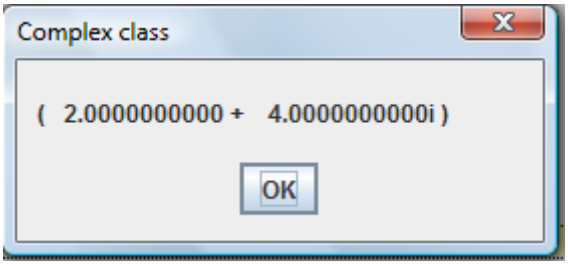
```
}

public boolean smaller(complex left,complex right)
{
// less then comparison of two complex numbers
return (left.R() < right.R());
}

public boolean smaller_equal(complex left,complex right)
{
// less then and equal comparison of two complex numbers
return (left.R() <= right.R());
}

public boolean greater(complex left,complex right)
{
// greater then comparison of two complex numbers
return left.R() > right.R();
}

public boolean greater_equal(complex left,complex  right)
{
// greater then and equal comparison of two complex numbers
return left.R() >= right.R();
}


public boolean equal(complex left,complex right)
{
// equal comparison of two complex numbers
return left.R() == right.R();
}

public boolean not_equal(complex left,complex  right)
{
// not equal comparison of two complex numbers
return left.R() != right.R();
}

public static String toString(double left, int w, int d)
// converts a double to a string with given width and decimals.
{
            NumberFormat df=NumberFormat.getInstance(Locale.US);
            df.setMaximumFractionDigits(d);
   df.setMinimumFractionDigits(d);
   df.setGroupingUsed(false);
   String s = df.format(left);
   while (s.length() < w)
     s = " " + s;
   if (s.length() > w)
   {
      s = "";
      for (int i=0; i<w; i++)
        s = s + "-";
   }
   return s;
}
```

```
public static String toString(double left)
{// converts a double to a string with a constant width and constant decimals.
  return toString(left,15,10);}

public static String toString(complex  value)
{
String b="";
if(Math.abs(value.imaginary())!=1)
  {
  if(value.imaginary() >= 0)
    b=b+"("+toString(value.real())+" + "+toString(value.imaginary())+"i )";
  else
    b=b+"("+toString(value.real())+" - "+toString(-value.imaginary())+"i )";
  }
else
  {
  if(value.imaginary() >= 0)
    b=b+"("+toString(value.real())+" + i )";
  else
    b=b+"("+toString(value.real())+" - i )";
}
return b;
}

public String toString()
{
String b="";
if(Math.abs(imaginary())!=1)
  {
  if(imaginary() >= 0)
    b=b+"("+toString(real())+" + "+toString(imaginary())+"i )";
  else
    b=b+"("+toString(real())+" - "+toString(-imaginary())+"i )";
  }
else
  {
  if(imaginary() >= 0)
    b=b+"("+toString(real())+" + i )";
  else
    b=b+"("+toString(real())+" - i )";
}
return b;
}
    public static complex toComplex(String s)
     {
                //bu metod compleks sayıyı ekrandan okur.
                //StringTokanizer kütüphane sınıfı bir stringicümlelere ayırır
     String s1=JOptionPane.showInputDialog(s);
     StringTokenizer token=new StringTokenizer(s1);
     int n=token.countTokens()-1;
     int m=n+1;
     double a[]=new double[m];
     int j=0;
     while(token.hasMoreTokens())
     {
     Double ax=new Double(token.nextToken());
     a[j++]=ax.doubleValue();
     }
```

```
    complex b=new complex(a[0],a[1]);
    return b;
    }
};
```

Program 7.3.2 An example program for class complex

```
import javax.swing.JOptionPane;
public class Complextest
{
public static void main(String arg[])
{
complex c1=new complex(1.0,2.0);
complex c2=new complex(c1);
c2.add(c1);
 JOptionPane.showMessageDialog(null,c2.toString(),
    "Complex class",JOptionPane.PLAIN_MESSAGE);
}
}
```



Another example from math is the rational numbers, a rational number class is given here.

Program 7.3.1 rational.java : a class with public class variables to be used for actual applications

```
//=========================================
// rational  number description
// Dr. Turhan Coban
//=========================================
import javax.swing.JOptionPane; // giriş çıkış

// class rational numbers


public class rational {
   // sınıf degişkenleri
   protected int   numerator;
   protected int   denominator;

   // kurucu metodlar

public rational()
{
numerator=0;
denominator=1;
}

public rational(int nnumerator,int ndenominator)
{
numerator=nnumerator;
denominator=ndenominator;
}
```

```
public rational(int numer)
{
numerator=numer;
denominator=1;
}

public rational(rational c )
{
numerator=c.numerator();
denominator=c.denominator();
}

//hesap metodları

public int leastcommondivisor()
{
// iki tam sayinin en buyuk ortak divideenini hesaplar
int n=numerator;
int m=denominator;
if(n==0)
   return m;
if(m==0)
   return n;
while(m != n)
{
  if(n>m)
     n=n-m;
  else
     m=m-n;
}
return n;
}

public void simplify()
{
//bayağı kesiri sadeleştirir
int isaret=1;
if(numerator<0)
{
 isaret=-isaret;
 numerator=-numerator;
}
if(denominator<0)
{
 isaret=-isaret;
 denominator=-denominator;
}
if(denominator==0)
   { JOptionPane.showMessageDialog(null,toString(),
     "Bayağı kesir sıfıra bölme hatası",
     JOptionPane.ERROR_MESSAGE);System.exit(0);
   }
int ebob=leastcommondivisor();
ebob=Math.abs(ebob);
numerator=isaret*numerator/ebob;
denominator=denominator/ebob;
}
```

```
// giriş - çıkış metotları

public int numerator()
{
return numerator;
}

public int denominator()
{
return denominator;
}

public void numeratorGir(int r)
{
numerator=r;
}

public void denominatorGir(int i)
{
denominator=i;
}

public void input_rational(rational sag)
{
numerator=sag.numerator();
denominator=sag.denominator();
}

public void input_rational(int nr,int ni)
{
numerator=nr;
denominator=ni;
}

public void input_rational(int nr)
{
numerator=nr;
denominator=1;
}

public void input_rational(double d)
{
// tam sayinin rational esitini yükle
// eh birazcık yaklaşım var tabi
input_rational((int)d*10000,10000);
simplify();
}

public double toDouble()
{
//rational sayinin gercek sayi esidi
return ((double)numerator/(double)denominator);
}

public static rational convert_to_rational(double d)
{
```

```
rational b=new rational();
b.input_rational((int)d*10000,10000);
b.simplify();
return b;
}


public void add(rational  sag)
{
numerator = numerator*sag.denominator() + sag.numerator()*denominator;
denominator = denominator*sag.denominator();
simplify();
}

public void add(int sag)
{
numerator = numerator + sag*denominator;
simplify();
}

public void substract(rational  sag)
{
numerator = numerator*sag.denominator() - sag.numerator()*denominator;
denominator = denominator*sag.denominator();
simplify();
}

public void substract(int sag)
{
numerator = numerator - sag*denominator;
simplify();
}

public void multiply(rational sag )
{
numerator = numerator*sag.numerator();
denominator = denominator*sag.denominator();
simplify();
}

public void multiply(int sag )
{
numerator = numerator*sag;
simplify();
}

public void divide(rational sag )
{
numerator = numerator*sag.denominator();
denominator = denominator*sag.numerator();
simplify();
}

public void divide(int sag )
{
denominator = denominator*sag;
simplify();
}
```

```
public static rational add(rational sol, rational sag)
{
 int r1= sol.numerator() * sag.denominator() + sag.numerator() * sol.denominator();
 int i1= sol.denominator()* sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational add(rational sol, int sag)
{
 int r1=sol.numerator() + sag*sol.denominator();
 int i1=sol.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational add(int sol, rational sag)
{
 int r1=sag.numerator() + sol*sag.denominator();
 int i1=sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

//===================================
public static rational substract(rational sol, rational sag)
{
 // iki rational sayının addmını return deyimiyle rational olarak
 // aktarır
 int r1= sol.numerator() * sag.denominator() - sag.numerator() * sol.denominator();
 int i1= sol.denominator()* sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational substract(rational sol, int sag)
{
 // bir rational ve bir numerator sayının addmını return deyimiyle rational olarak
 // aktarır
 int r1=sol.numerator() - sag*sol.denominator();
 int i1=sol.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational substract(int sol, rational sag)
{
```

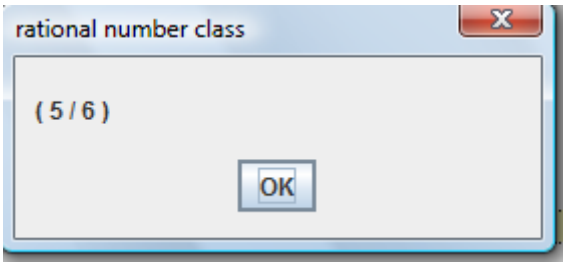
```
 int r1=sag.numerator() - sol*sag.denominator();
 int i1=sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

//===================================

public static rational multiply(rational sol, rational sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator()*sag.numerator(),sol.denominator()*sag.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational multiply(rational sol, int sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator()*sag,sol.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational multiply(int sol, rational sag)
{
 rational sonuc;
 sonuc=new rational(sag.numerator()*sol,sag.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational divide(rational sol, rational  sag)
{
 double a=sag.numerator()*sag.numerator()+sag.denominator()*sag.denominator();
 rational sonuc;
 sonuc=new rational((int)(sol.numerator()*sag.denominator()),(int)(sol.denominator()*sag.numerator()));
 sonuc.simplify();
 return sonuc;
}

public static rational divide(rational sol, int  sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator(),(sol.denominator()*sag));
 sonuc.simplify();
 return sonuc;
}

public boolean smaller_than(rational sol,rational sag)
{
// less then comparison of two rational numbers
return (sol.toDouble() < sag.toDouble());
}

public boolean smaller_than_equals(rational sol,rational sag)
```

```
{
// less then and equals comparison of two rational numbers
return (sol.toDouble() <= sag.toDouble());
}

public boolean greater_than(rational sol,rational sag)
{
// greater_than then comparison of two rational numbers
return sol.toDouble() > sag.toDouble();
}

public boolean greater_than_equals(rational sol,rational  sag)
{
// greater_than then and equals comparison of two rational numbers
return sol.toDouble() >= sag.toDouble();
}


public boolean equals(rational sol,rational sag)
{
// equals comparison of two rational numbers
return sol.toDouble() == sag.toDouble();
}

public boolean not_equals(rational sol,rational  sag)
{
// not equals comparison of two rational numbers
return sol.toDouble() != sag.toDouble();
}

public static String toString(rational  value)
{
String b="";
if(Math.abs(value.denominator())!=1)
  {
    b=b+"( "+value.numerator()+" / "+value.denominator()+" )";
  }
else
  {
    b=b+value.numerator()+" ";
  }
return b;
}

public String toString()
{
// yazima hazır rational formda String deşiskenini iletir.
String b="";
if(Math.abs(denominator())!=1)
  {
    b=b+"( "+numerator+" / "+denominator()+" )";
  }
else
  {
    b=b+numerator()+" ";
  }
return b;
}
```

```
            };

//rational sınıfının sonu
```

Program 7.3.4 An example program for class rational

```
import javax.swing.JOptionPane;
public class rationaltest
{
public static void main(String arg[])
{
rational r1=new rational(1,2);
rational r2=new rational(1,3);
r2.add(r1);
 JOptionPane.showMessageDialog(null,r2.toString(),
    "rational number class",JOptionPane.PLAIN_MESSAGE);
}
}
```



EXAMPLE 1

WEEK 7 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) write a small test program to calculate cube (third power) z=1+2i by using class complex

EX2) By using 3 dimensional cartesian coordinate system, write class point3D P(x,y,z) (similar to class point)

EX3) By using 3 dimensional cylindirical coordinate system, write class point3DCYL P(r,θ,z)

WEEK 7 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) Write a class cone1 by using composition (use class circle). Class should include area and volume of the cone. Write an example test program and display cone area and volume.

HW2) Write a class cone1 by using inheritance-extend statement(use class circle). Class should include area and volume of the cone. Write an example test program and display cone area and volume.

HW3) By using point3D class and composition create class cylinder in 3D space.

HW4) In exercise 3, in a three dimensional cylindirical coordinate system, class point3DCYL is created. P(r,θ,z) . write a 3 dimensional representation of a cylinder by using inheritance and point3DCYL class

8 ARRAYS

8.1 GENERAL DEFINITION OF ONE DIMENSIONAL ARRAY VARIABLES

Arrays defines more than one variable or object with the same group name. Each individual variable can be reached with the assigned index. In General Array type variables are defined as :

Variable_type variable_name[];

Or

Variable_type[] variable_name;

An example definition :

int days_of_the_month[]; or

int[] days_of_the_month;

This definition defines that variable has more than one sub-variable in it, but it does not define how many sub-variables it containes. In order to define array size

variable_name=new Variable_type[variable_size];

structure used. example definition :

days_of_the_month=new int[12];

After array variable is defined, any subvariable can be reached by using its index number. For example:

days_of_the_month[0]=31;

days_of_the_month[1]=28;

Note that index variable is always starts from 0

If array concept is given in a program form :

Program 8.1.1 The first array example array1a.java

```
import java.io.*;
class array1a
{
public static void main(String args[] )
{
int days_of_the_month[];
days_of_the_month=new int[12];
days_of_the_month[0]=31;
days_of_the_month[1]=28;
days_of_the_month[2]=31;
days_of_the_month[3]=30;
days_of_the_month[4]=31;
days_of_the_month[5]=30;
days_of_the_month[6]=31;
days_of_the_month[7]=31;
days_of_the_month[8]=30;
days_of_the_month[9]=31;
days_of_the_month[10]=30;
days_of_the_month[11]=31;
System.out.println("Month April is "+days_of_the_month[3]+" days long");
}}
```

The output will be:

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" array1a
Month April is 30 days long

> Terminated with exit code 0.
```

In the first creation of arrays arrays assignments can also be made by using big paranthesis,{}.
Variable_type[] variable_name={value_0,value_1,..,value_n};
Not that in this type of definition the size of the variable is not defined seperately, program counts the number of variables inside of the paranthesis and assign the size as well as actual valu for each sub-variable. An example can be given as:
int days_of_the_month[[]={31,28,31,30,31,30,31,31,30,31,30,31};
If array concept is given in a program form :

Program 8.1.2 The second array example array1b.java

```
import java.io.*;
class array1b
{
public static void main(String args[] )
{
int days_of_the_month[]={31,28,31,30,31,30,31,31,30,31,30,31};
System.out.println("Month April is "+days_of_the_month[3]+" days long");
}
}
```

The result will be same as the previous program. Since array type of variables are reached by using indices, they can be used easily by using loop structures. The following program finds avarage of given set of numbers by using for loop structures an arrays

Program 8.1.3 math1.java an example of acessing array indices through for loop structure

```
class math1
{public static void main(String args[])
{int number[]={10,5,7,9,11,13,14,18};
double total=0;
int i;
for(i=0;i<number.length;i++)
{total+=number[i];}
System.out.println("Average = "+total/number.length);
}}
```

The output will be :

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" math1
Average = 10.875
> Terminated with exit code 0.
```

Of course the same result can be achived by using while structure

Program 8.1.4 math2.java an example of acessing array indices through while loop structure

```
class math2
```

```
{public static void main(String args[])
{int number[]={10,5,7,9,11,13,14,18};
double total=0;
int i=0;
while(i<number.length)
{total+=number[i];i++;}
System.out.println("Average = "+total/number.length);
}}
```

The result will be the same as the previous program. In another example the values in two arrays are added up together. In this example formatted output is used to get a nice output for arrays. Some of the format varible types are given in table 8.1.1

Table 8.1.1 Format structures

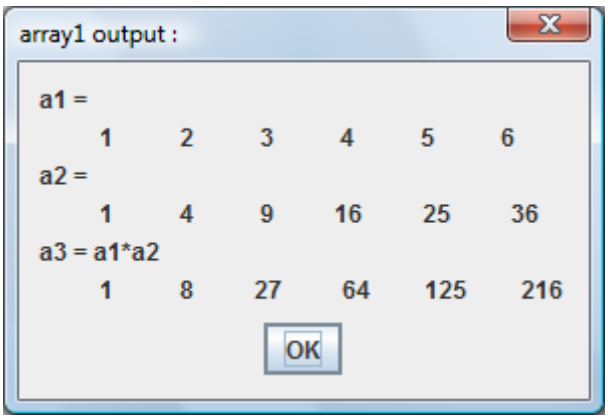
Conversion	Argument Category	Description
'b', 'B'	general	If the argument *arg* is null, then the result is "false". If *arg* is a boolean or Boolean, then the result is the string returned by String.valueOf(). Otherwise, the result is "true".
'h', 'H'	general	If the argument *arg* is null, then the result is "null". Otherwise, the result is obtained by invoking Integer.toHexString(arg.hashCode()).
's', 'S'	general	If the argument *arg* is null, then the result is "null". If *arg* implements Formattable, then arg.formatTo is invoked. Otherwise, the result is obtained by invoking arg.toString().
'c', 'C'	character	The result is a Unicode character
'd'	integer	The result is formatted as a decimal integer
'o'	integer	The result is formatted as an **octal** (base 8) integer
'x', 'X'	integer	The result is formatted as a **hexadecimal** (base 16) integer
'e', 'E'	floating point	The result is formatted as a decimal number in computerized scientific notation
'f'	floating point	The result is formatted as a decimal number
'g', 'G'	floating point	The result is formatted using computerized scientific notation or decimal format, depending on the precision and the value after rounding.
'a', 'A'	floating point	The result is formatted as a hexadecimal floating-point number with a significand and an exponent
't', 'T'	date/time	Prefix for date and time conversion characters. See Date/Time Conversions.
'%'	percent	The result is a literal '%' ('\u0025')
'n'	line separator	The result is the platform-specific line separator

Program 8.1.5 array1.java an example of acessing array indices through for loop structure

```
import javax.swing.*;
import java.util.Locale;

 public class array1
 {
 public static void main(String[] args)
 {
    int a1[]={1,2,3,4,5,6};
    int a2[]={1,4,9,16,25,36};
    int a3[]=new int[a2.length];
    String s="array1 output : ";
    String s1="a1 =\n"+output(a1);
```

```
    s1+="a2 = \n"+output(a2);
    for(int i=0;i<a1.length;i++) {a3[i]=a1[i]*a2[i];}
    s1+="a3 = a1*a2 \n"+output(a3);
    JOptionPane.showMessageDialog(null,s1,s,JOptionPane.PLAIN_MESSAGE);
}
 public static String output(int a[])
 {
 Locale tr=new Locale("TR");
 int x=25;
 String s1="";
   for(int i=0;i<a.length;i++)
   { s1+=String.format(tr," %10d ",a[i]);}
  s1+="\n";
  return s1;
 }

 }
```

The result will be :

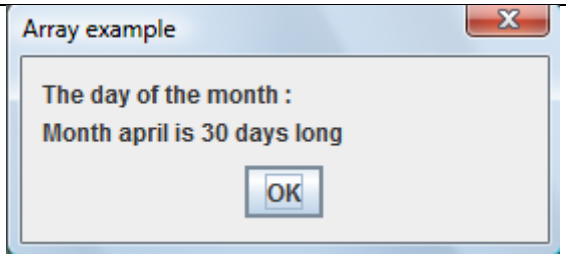


In the next example String type of array variable is introduced.

Program 8.1.6 H8Ex2B.java an example of acessing array indices through for loop structure

```
import java.io.*;
import javax.swing.*;
class H8Ex2B {
public static int month(String s)
{

String month[]={"january","february","march","april","may","june","july",
          "august","september","october","november","december"};
int i;
int j=-1;
for(i=0;i<12;i++)
{if(s.equals(month[i])) {j=i;break;}}
return j;
}


public static void main(String args[] )
{ int the_day_of_the_month[]={31,28,31,30,31,30,31,31,30,31,30,31};
String s="The day of the month : \n";
String s2="april";
int i=month(s2);
```

```
s+="Month "+s2+" s "+ayin_gunleri[i]+" days long \n";
String s1="Array example";
JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```



8.2 ENHANCED FOR STATEMENT

In latest java versions a simpler for statement is introduced to use with for arrays. As an old programmer I have not used it much but you have right to know the new structure. In the normal for structure, the for loop was given as :

Program 8.2.1 ForTest.java an example of acessing array indices through for loop structure

```
public class ForTest
 {
 public static void main( String args[] )
 {
 int array[] = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
 int total = 0;
   for ( int i=0;i<array.length;i++)
     {total += array[i];}
   System.out.println( "Total = "+total );
 } // end main
 }
```

In the enhanced for loop the same can be achived as :

Program 8.2.2 ForTest.java an example of acessing array indices through enhanced for loop structure

```
public class ForTest1
 {
 public static void main( String args[] )
 {
 int array[] = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
 int total = 0;
     for ( int i:array)
     {total += number;}
   System.out.println( "total = "+total );
 } // end of main }
```

As it is seen for structure in the second program is given as:

```
for ( int number:array)
{total += number;}
```

In the original for structure this procedure was

```
for ( int i=0;i<array.length;i++)
  {total += array[i];}
```

Note that in the second structure number is not counter but represent array[i] value in the previous program. Using this new form certainly has advantages and disadvantages. Original form is best known and understood by everyone. Understanding by everybody is an important issue in languages computer as well as human languages. Therefore mostly original form is used in this book, but when it is used with the object array the new form has advantages in referencing.

8.3 ARRAYS OF OBJECTS

Array structures can be used for objects as well as simple variables.

Program 8.3.1 box.java

```
class box
{
private double length;
private double width;
private double height;

box(double l,double w, double h)
{length=l;
width=w;
height=h;
}
// This methods are written to access private length,
// width and height variables
public double read_length() {return length;}
public double read_width() {return width;}
public double read_height() {return height;}
//This methods are written to change private length,
// width and height variables
public void write_length(double l) {length=l;}
public void write_width(double w) {width=w;}
public void write_height(double h) {height=h;}

box(box bi)
{length=bi.length;
width=bi.width;
height=bi.height;
}

double volume()
{return length*width*height;}

public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n\n";
  return s;
}

}
```

Program 8.3.2 An example program using box.java in array form

```
import java.io.*;
import javax.swing.*;
class H8Ex3
{
public static void main(String args[] )
{   box x[]=new box[4];
    x[0]=new box(1.1,2.2,3.3);
    x[1]=new box(1.23,2.11,3.12);
    x[2]=new box(2.5,2.11,3.0);
    x[3]=new box(2.5,2.11,3.1);
  String s1="Array example ";
  String s="";
  for(int i=0;i<x.length;i++)
    {s+="Box number = "+i+"\n"+x[i].toString();}
  JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

If this program written with enhanced for statement:

Program 8.3.2A An example program using box.java in array form written with enhanced for statement

```
import java.io.*;
import javax.swing.*;
class H8Ex3A
{
public static void main(String args[] )
{   box x[]=new box[4];
    x[0]=new box(1.1,2.2,3.3);
    x[1]=new box(1.23,2.11,3.12);
    x[2]=new box(2.5,2.11,3.0);
    x[3]=new box(2.5,2.11,3.1);
  String s1="Array example ";
  String s="";
  for(box number:x)
    {s+=number.toString();}
  JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

The advantages of using an enhanced for with arrays of objects are clear in here, it is a shorter form. But as it is seen it is not necassary. The basic difference of the object type arays is that we should initilize them by using constructor classes. Each individual array element should be constructed individually. In our second example a vector class will be defined and vectors will be added directly in the output example. It should also be note that dimension variables can be used when methods area called

Program 8.3.3 vector.java class

```
import java.util.Locale;
```

```
public class vector
 {
 public double V[];
 //kurucu metodlar
 public vector(double Vi[])
 { V=new double[Vi.length];
   input_vector(Vi);
 }

 public vector(vector Vi)
 { V=new double[Vi.V.length];
   input_vector(Vi.V);
 }

 public vector(int n)
 { V=new double[n];
   for(int i=0;i<n;i++)
                { V[i]=0.0;}
 }

 public void input_vector(double Vi[])
 {   for(int i=0;i<V.length;i++)
                  {V[i]=Vi[i];}
 }

 public static vector add(vector V1,vector V2)
 {
 //
 int n=V1.V.length;
 vector V3=new vector(n);
  for(int i=0;i<n;i++)
                  {V3.V[i]=V1.V[i]+V2.V[i];}
 return V3;
 }

 public static vector add(double[] V1,double[] V2)
 {
 //
 int n=V1.length;
 vector M=new vector(n);
  for(int i=0;i<n;i++)
                  {M.V[i]=V1[i]+V2[i];}
 return M;
 }

 public void add(vector V1)
 {
 //
 int n=V.length;
  for(int i=0;i<n;i++)
                  {V[i]+=V1.V[i];}
 }

 public void add(double [] B)
 {
 //
 int n=V.length;
  for(int i=0;i<n;i++)
```

```
                {V[i]+=B[i];}
 }

 public static String toString(vector M)
 { int n=M.V.length;
   String s="";
  for(int i=0;i<n;i++)
                {s+=M.V[i]+" ";}
 s+="\n";
 return s;
 }

 public String toString()
 {  Locale tr=new Locale("TR");
              int n=V.length;
    String s1="";
    for(int i=0;i<n;i++)
    { s1+=String.format(tr," %10.5f ",V[i]);}
   s1+="\n";
 return s1;
 } //

 }//
```

Program 8.3.4 An example using vector class

```
import javax.swing.*;
class H8Ex7 {
public static void main(String args[] )
{ double A[]={1.0,2.0,3.0,4.0};
  double B[]={1.0,1.0,1.0,1.0};
  vector M1=new vector(A);
  vector M2=new vector(B);
  M1.add(M2);
  String s="summation of two vector : \n"+M1.toString();
  String s1="class vector";
  JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
} }
```

As a third example, book class defined in the previous chapter will be used in array structure

Program 8.3.5 library class that use book class as an array object variable

```
public class library
{
String name;
int number_of_books;
book list[];

public library(String kname,book klist[])
{name=kname;
 number_of_books=klist.length;
 list=new book[number_of_books];
 for(int book_number=0;book_number<number_of_books;book_number++)
 {list[book_number]=klist[book_number];}
}


 public String toString()
```

```
 {          String s="Library name : "+name+"\n";
    s+="Total number of books : "+number_of_books+"\n\n";
    for(int book_number=0;book_number<number_of_books;book_number++)
    {s+=list[book_number].toString()+"\n";}
    return s;
 }
}
```

Program 8.3.6 An example program using book and library class that use book class as an array object variable

```
import javax.swing.*;
class booktest1
{
public static void outputwindow(String s)
{
String s1="library class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{

book b[]=new book[5];
b[0]=new book("Bilim ve Teknik","TÜBİTAK","magazine","Turkish");
b[1]=new fiction("It is all quiet in the westen front ","Erich Maria Reamarque","English");
b[2]=new book("The God Delusion","Richard Dawkins","fiction","English");
b[3]=new poetry("Human lanscapes from my country","Nazım Hikmet","English and Turkish");
b[4]=new encyclopedia("Encyclopedia Britannica","English");
library l=new library("Library of congress",b);
String s=l.toString();
outputwindow(s);
}
```

It should be noted that, in the last examples fiction poetry and encyclopedia classes are used as well in the book definition, because, tese classes are extended from class book.

A similar example will be given for vehicle class of the previous chapter

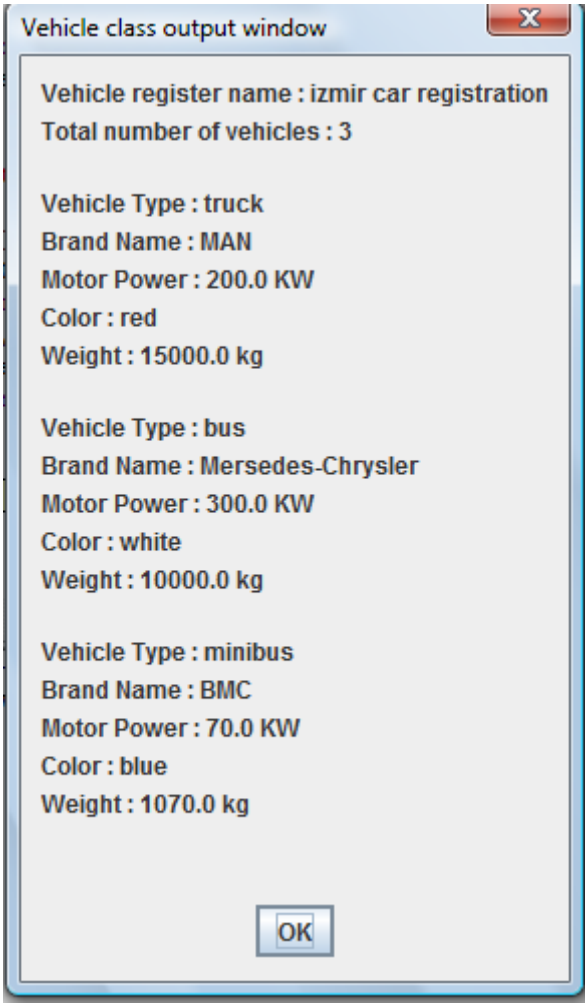
Program 8.3.7 vehicle_register class that use vehicle class as an array object variable

```
public class vehicle_register
{
String name;
int number_of_vehicles;
vehicle list[];

public vehicle_register(String kname,vehicle klist[])
{name=kname;
 number_of_vehicles=klist.length;
 list=new vehicle[number_of_vehicles];
 for(int vehicle_number=0;vehicle_number<number_of_vehicles;vehicle_number++)
 {list[vehicle_number]=klist[vehicle_number];}
}

 public String toString()
 {          String s="Vehicle register name : "+name+"\n";
    s+="Total number of vehicles : "+number_of_vehicles+"\n\n";
```

```
    for(int vehicle_number=0;vehicle_number<number_of_vehicles;vehicle_number++)
    {s+=list[vehicle_number].toString()+"\n";}
    return s;
 }

}
```

Program 8.3.8 An example program utilises vehicle class and vehicle_registration class

```
import javax.swing.*;
class vehicletest1
{

public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}

public static void main(String args[])
{
vehicle v[]=new vehicle[3];
v[0]=new truck("MAN",200.0,"red",15000);
v[1]=new bus("Mersedes-Chrysler",300.0,"white",10000);
v[2]=new minibus("BMC",70,"blue",1070);
vehicle_register vr=new vehicle_register("izmir car registration",v);
String s=vr.toString()+"\n";
outputwindow(s);
}

}
```



Program 8.3.9 Another example program Student and course classes are given, this two classes specify students in the class and avarage grade from the class for each student and letter grades of the students according to class average and standard deviation

```
public class Student
{
private String firstName;
private String lastName;
private String StudentNumber;
private String school;
private double grade[];
private String lettergrade;
private double weight[];
private double totalweight;
public Student( String first, String last, String sn,String school_name )
{
firstName = first;
lastName = last;
StudentNumber = sn;
if(school_name.equals("mechanical") || school_name.equals("chemical") ||
school_name.equals("electrical") || school_name.equals("industrial") ||
school_name.equals("civil") || school_name.equals("mining"))
{school=school_name;}
```

```
else
{school="not defined";}
lettergrade=" ";
}

public void setFirstName( String first )
{
firstName = first;
}
public String getFirstName()
{
return firstName;
}

public void setLastName( String last )
{
lastName = last;
}

public String getLastName()
{
return lastName;
}

public void setStudentNumber( String ssn )
{
StudentNumber = ssn;
}
public void setLetterGrade( String ssn )
{lettergrade = ssn; }

public String getStudentNumber()
{
return StudentNumber;
}
public String getSchool()
{ return school; }

public void setGrade(double gradi[])
{int n=gradi.length;
grade=new double[n];
for(int i=0;i<n;i++)
{grade[i]=gradi[i];}
}
public void setWeight(double weighti[])
{int n=weighti.length;
weight=new double[n];
totalweight=0;
for(int i=0;i<n;i++)
{weight[i]=weighti[i];totalweight+=weight[i];}
}
public double[] getGrade()
```

```
{ return grade;
}
public double[] getWeight()
{ return weight;
}
public double getAverage()
{ double avg=0;
  int n=weight.length;
  for(int i=0;i<n;i++)
  {avg+=weight[i]*grade[i];}
  avg/=totalweight;
  return avg;
}
public String toString()
{ String s=getFirstName()+" "+getLastName()+" "+getStudentNumber()+" "+getSchool()+"
"+getAverage()+"Letter grade = "+lettergrade+"\n";
 return s;
}
}
```

Program 8.3.10 class course

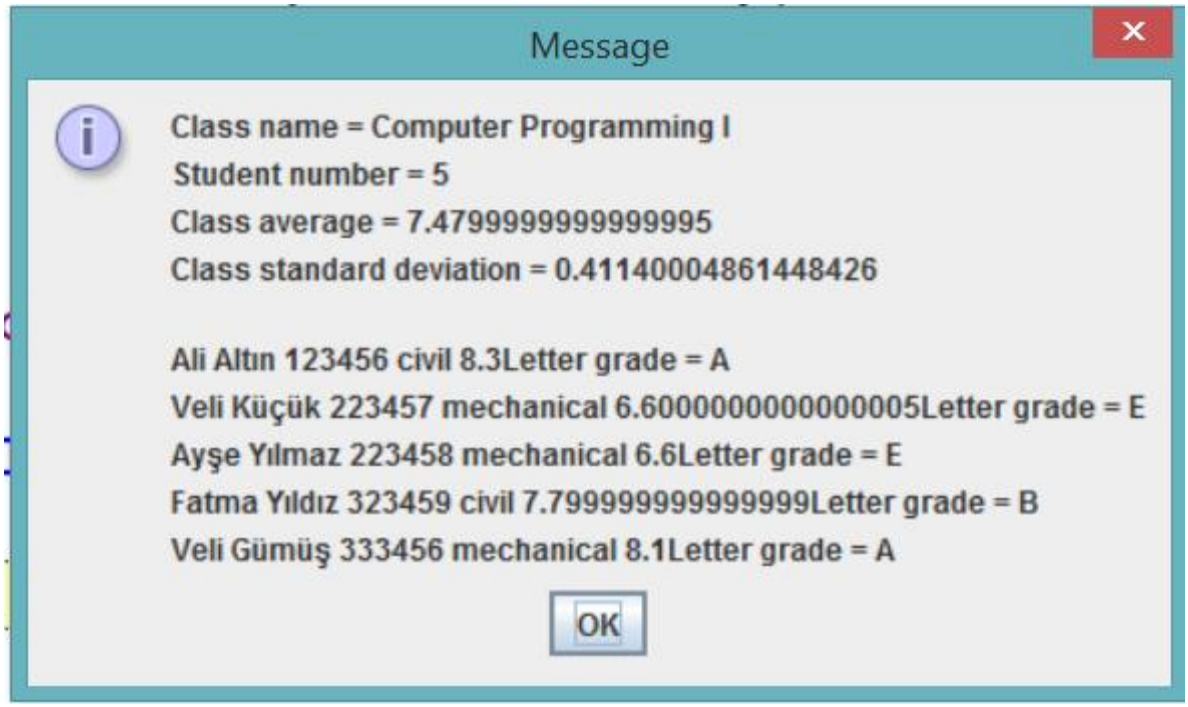
```
public class course
{
public String name;
public Student[] student;
private double weight[];
private double totalweight;

public course(String namei,Student[] studenti,double weighti[])
{
name=namei;
int n=studenti.length;
student=studenti;
setWeight(weighti);
for(int i=0;i<n;i++)
{student[i].setWeight(weighti);}
}

public void setWeight(double weighti[])
{int n=weighti.length;
weight=new double[n];
totalweight=0;
for(int i=0;i<n;i++)
{weight[i]=weighti[i];totalweight+=weight[i];}
}
public void setGrade(double grade[][])
{int n=weight.length;
 int m=student.length;
 for(int i=0;i<m;i++)
 {student[i].setGrade(grade[i]);}
}
```

```
public double getAverage()
{ double avg=0;
  int n=student.length;
  for(int i=0;i<n;i++)
  {
   avg+=student[i].getAverage();
  }
  avg/=n;
  return avg;
}

public double getStandardDeviation()
{ double avg=getAverage();
  double studentavg=0;
  int n=student.length;
  double total=0;
  for(int i=0;i<n;i++)
  { studentavg=student[i].getAverage();
   total+=(studentavg-avg)*(studentavg-avg);
  }
  total=Math.sqrt(total)/(n-1);
  return total;
}

public void setLettergrades()
{  double avg=getAverage();
   double std=getStandardDeviation();
   double studentavg=0;
   int n=student.length;
   for(int i=0;i<n;i++)
   { studentavg=student[i].getAverage();
     if(studentavg>(avg+3.0*std/2.0)) student[i].setLetterGrade("A");
     else if(studentavg<=(avg+3.0*std/2.0) && studentavg>=(avg+std/2.0))
student[i].setLetterGrade("B");
     else if(studentavg<=(avg+std/2.0) && studentavg>=(avg-std/2.0))
student[i].setLetterGrade("C");
     else if(studentavg<=(avg-std/2.0) && studentavg>=(avg-3.0*std/2.0))
student[i].setLetterGrade("D");
     else if(studentavg<=(avg-3.0*std/2.0) && studentavg>=(avg-5.0*std/2.0))
student[i].setLetterGrade("E");
     else if(studentavg<=(avg-5.0*std/2.0)) student[i].setLetterGrade("F");
   }
}

public String toString()
{int n=student.length;
 setLettergrades();
 String s="Class name = "+name+"\n";
 s+="Student number = "+n+"\n";
 s+="Class average = "+getAverage()+"\n";
 s+="Class standard deviation = "+getStandardDeviation()+"\n\n";
 for(int i=0;i<n;i++)
```

```
 {s+=student[i].toString();}
return s;
}
}
```

Program 8.3.11 class coursetest

```
import javax.swing.*;
public class coursetest
{
public static void main(String arg[])
{
Student s1[]=new Student[5];
s1[0]=new Student("Ali","Altın","123456","civil");
double a1[]={10.0,7.0,8.0};
s1[0].setGrade(a1);

s1[1]=new Student("Veli","Küçük","223457","mechanical");
double a2[]={7.0,7.0,6.0};
s1[1].setGrade(a2);

s1[2]=new Student("Ayşe","Yılmaz","223458","mechanical");
double a3[]={9.0,5.0,6.0};
s1[2].setGrade(a3);

s1[3]=new Student("Fatma","Yıldız","323459","civil");
double a4[]={9.0,5.0,9.0};
s1[3].setGrade(a4);

s1[4]=new Student("Veli","Gümüş","333456","mechanical");
double a5[]={10.0,5.0,9.0};
s1[4].setGrade(a5);

double weight[]={0.3,0.3,0.4};
course c1=new course("Computer Programming I",s1,weight);
String s=""+c1;
JOptionPane.showMessageDialog(null,s);
}
}
```

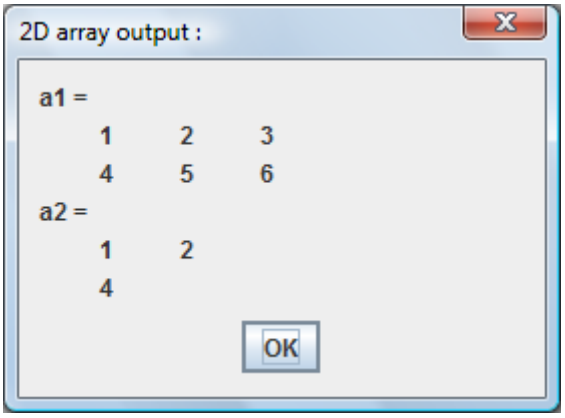


8.4 MULTI-DIMENSIONAL ARRAY VARIABLES OF OBJECTS

Arrays can be given in more than one dimensions. When arrays are more than one dimension, they can be considered as arrays of arrays.

Program 8.4.1 array2D.java 2 dimensional arry example

```
import javax.swing.*;
import java.util.Locale;

 public class array2D
 {
 public static void main(String[] args)
 {
    int a1[][]={{1,2,3},{4,5,6}};
    int a2[][]={{1,2},{4}};
    String s="2D array output : ";
    String s1="a1 =\n"+output2D(a1);
    s1+="a2 = \n"+output2D(a2);
    JOptionPane.showMessageDialog(null,s1,s,JOptionPane.PLAIN_MESSAGE);
}
 public static String output2D(int a[][])
 {
 Locale tr=new Locale("TR");
 int x=25;
 String s1="";
   for(int i=0;i<a.length;i++)
   {  for(int j=0;j<a[i].length;j++)
    { s1+=String.format(tr," %10d ",a[i][j]);}
   s1+="\n";
   }
   return s1;
 }}
```



The same concept can be applied to class objects as well

Program 8.4.2 matrix.java class with two dimensional double variables

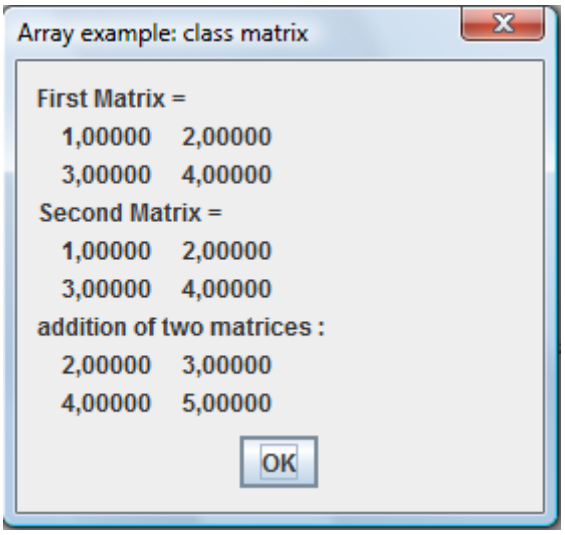
```
import java.util.Locale;
public class matrix
 {

 public double A[][];
 //constructor methods
 public matrix(double Ai[][])
 { A=new double[Ai.length][Ai[0].length];
   input_matrix(Ai);
 }

 public matrix(matrix M)
 { A=new double[M.A.length][M.A[0].length];
   input_matrix(M.A);
 }

 public matrix(int n,int m)
 { A=new double[n][m];
   for(int i=0;i<n;i++)
              { for(int j=0;j<m;j++)
                {A[i][j]=0.0;}
     }
 }

 public void input_matrix(double Ai[][])
 {   for(int i=0;i<A.length;i++)
              { for(int j=0;j<A[0].length;j++)
                {A[i][j]=Ai[i][j];}
     }
 }

 public static matrix add(matrix M1,matrix M2)
 {
 //
 int n=M1.A.length;
 int m=M1.A[0].length;
 matrix M3=new matrix(n,m);
  for(int i=0;i<n;i++)
              { for(int j=0;j<m;j++)
                {M3.A[i][j]=M1.A[i][j]+M2.A[i][j];}
```

```
    }
return M3;
}

public static matrix add(double[][] A1,double[][] A2)
{
//
int n=A1.length;
int m=A1[0].length;
matrix M=new matrix(n,m);
 for(int i=0;i<n;i++)
            { for(int j=0;j<m;j++)
              {M.A[i][j]=A1[i][j]+A2[i][j];}
    }
return M;
}

public void add(matrix M1)
{
//
int n=A.length;
int m=A[0].length;
 for(int i=0;i<n;i++)
            { for(int j=0;j<m;j++)
              {A[i][j]+=M1.A[i][j];}
    }
}

public void add(double [][] B)
{
//
int n=A.length;
int m=A[0].length;
 for(int i=0;i<n;i++)
            { for(int j=0;j<m;j++)
              {A[i][j]+=B[i][j];}
    }
}

public static String toString(matrix M)
{ int n=M.A.length;
  int m=M.A[0].length;
  String s="";
 for(int i=0;i<n;i++)
            { for(int j=0;j<m;j++)
              {s+=M.A[i][j]+" ";}
            s+="\n";
            }
return s;
}

public String toString()
{  Locale tr=new Locale("TR");
           int n=A.length;
  int m=A[0].length;
  String s1="";
  for(int i=0;i<n;i++)
  {  for(int j=0;j<m;j++)
```

```
   { s1+=String.format(tr," %10.5f ",A[i][j]);}
  s1+="\n";
  }
 return s1;
 } //end of method toString

 }//matrix sınıfının sonu
```

Program 8.4.3 An example program using class matrix to add two matrices

```
import javax.swing.*;
class H8Ex4 {
public static void main(String args[] )
{ double A[][]={{1.0,2.0},{3.0,4.0}};
double B[][]={{1.0,1.0},{1.0,1.0}};
matrix M1=new matrix(A);
String s="First Matrix = \n"+M1.toString();
matrix M2=new matrix(B);
s+="Second Matrix = \n"+M1.toString();
M1.add(M2);
s+="addition of two matrices : \n"+M1.toString();
String s1="Array example: class matrix";
JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
} }
```



In a different for of the program, matrix is also used as an array structure. In this version static add method is used to get output.

Program 8.4.4 An example program using class matrix to add two matrices

```
import javax.swing.*;
class H8Ex5 {
public static void main(String args[] )
{ double A[][]={{1.0,2.0},{3.0,4.0}};
  double B[][]={{1.0,1.0},{1.0,1.0}};
  matrix M[]=new matrix[3];
  M[0]=new matrix(A);
  M[1]=new matrix(B);
  M[2]=matrix.add(M[0],M[1]);
  String s="First Matrix = \n"+M[0].toString();
```

```
  s+="Second Matrix = \n"+M[1].toString();
  s+="addition of two matrices : \n"+M[2].toString();
  String s1="Array example: class matrix";
  JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
} }
```

The result will be same as the previous example.

8.5 CHANGING ARRAY DIMENSIONS

Dimensions of the arrays can not be changed due to the fact that memory is assigned in the definition of the array. But by using indirect methods it can be changed, namely by creating a different array with a different dimension and reloaded array values into this new array.

```
double a[]={3.0,5.0,7.0}
double b[]=new double[4];
for(int i=0;i<a.length;i++)
   {b[i]=a[i];}
a=b;
```

In this small program part dimension of the array a is changed from 3 to 4. Now this basic process is carried out for a double array through a class, doubleArray

Program 8.5.1 : dimension changing example doubleArray

```
public class doubleArray
{
  //sınıf değişkenleri
  public double a[];
  public int length;

  public doubleArray(double x[] )
  {
  length=x.length;
  a=new double[length];
     for(int i=0;i<length;i++)
       a[i]=x[i];
  }

  public doubleArray(int n)
  {
  a=new double[n];
  length=a.length;
  }

  public void addDimension(int n)
  {
  if(n>0)
     {
     int z=a.length+n;
     double[] b=new double[z];
     for(int i=0;i<a.length;i++)
       {b[i]=a[i];}
     a=b;
     length=a.length;
```

```
    }
  }

  public void addDimension()
  {
  addDimension(1);
  }

  public void removeDimension(int n)
  {
  int z=a.length-n;
   if(z>1)
   {
   double[] b=new double[a.length-n];
   for(int i=0;i<z;i++)
      b[i]=a[i];
   a=b;
   }
   else
   {
   double[] b=new double[1];
   b[0]=a[0];
   a=b;
   }
   length=a.length;
  }

  public void removeDimension()
  {
removeDimension(1);
  }

  public double getValue(int i)
  {
  return a[i];
  }

  public double[] getValue()
  {
  return a;
  }

  public void setValue(double x,int i)
  {
  a[i]=x;
  }

  public void setValue(double[] x )
  {
  length=x.length;
  a=new double[length];
  for(int i=0;i<length;i++)
```

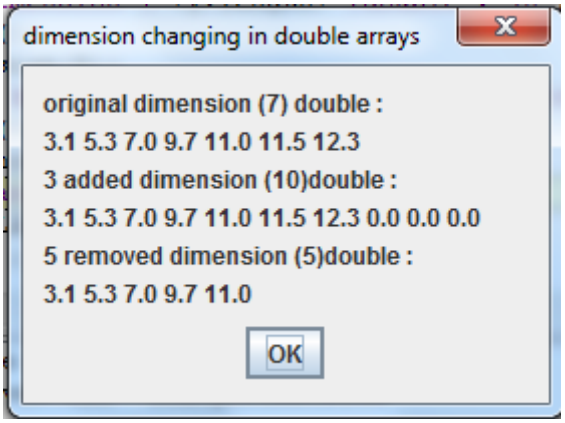
```
    {
    a[i]=x[i];
    }
  }

  public String toString(int i)
  {
   return ""+a[i];
  }

  public String toString()
  {
  String s="";
  for(int i=0;i<length;i++)
   {
   s+=a[i]+" ";
   }
   s+="\n";
   return s;
  }
}
```

DoubleBoyut sınıfında boyutu değiştirilebilen değişken (nesne) tanımladık. Bu değişkeni boyutDegistir sınıfında test edelim :

Program 8.5.2 : dimension changing example : changeDimension

```
import javax.swing.JOptionPane;

public class changeDimension
{
  public static void main(String args[])
  {
  double x[]={3.1,5.3,7.0,9.7,11.0,11.5,12.3};
  doubleArray y=new doubleArray(x);
  String s="original dimension ("+y.length+") double : \n";
  s+=y.toString();
  y.addDimension(3);
  s+="3 added dimension ("+y.length+")double : \n";
  s+=y.toString();
  y.removeDimension(5);
  s+="5 removed dimension ("+y.length+")double : \n";
  s+=y.toString();
  JOptionPane.showMessageDialog(null,s,
  "dimension changing in double arrays",
  JOptionPane.PLAIN_MESSAGE);
  System.exit(0);
  }
}
```

Program output:



ARRAY EXAMPLES (created 2010 fall semester class lecture)

EXAMPLE 1 : Create a double array and calculate sum

```
public class array1
{
public static void main(String arg[])
{
double x[]=new double[3];
x[0]=1.2;
x[1]=2.3;
x[2]=3.1;
double total=0;
for(int i=0;i<x.length;i++)
{total+=x[i];
System.out.println("x["+i+"] = "+x[i]);
}
System.out.println("total="+total);
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array1
x[0] = 1.2
x[1] = 2.3
x[2] = 3.1
total=6.6
> Terminated with exit code 0.
```

EXAMPLE 2 : Create a double array and calculate sum

```
public class array2
{
public static void main(String arg[])
{
double x[]={1.2,2.3,3.3};
double total=0;
for(int i=0;i<x.length;i++)
```

```
{total+=x[i];
System.out.println("x["+i+"] = "+x[i]);
}
System.out.println("total="+total);
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array2
x[0] = 1.2
x[1] = 2.3
x[2] = 3.3
total=6.8

> Terminated with exit code 0.
```

EXAMPLE 3 : Create a string array and add array variables into one string

```
public class array3
{
public static void main(String arg[])
{
String x[]={"Ali "," Veli"," 49 "," elli"};
String s="";
for(int i=0;i<x.length;i++)
{s+=x[i];
System.out.println("x["+i+"] = "+x[i]);
}
System.out.println("s="+s);
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array3
x[0] = Ali
x[1] =  Veli
x[2] =  49
x[3] =  elli
s=Ali  Veli 49  elli

> Terminated with exit code 0.
```

EXAMPLE 4 : Create a box3 class array and calculate volume of the boxes in array elements

```
public class box3
{
public double length;
public double width;
public double height;
```

```
public String color;
public box3(double l,double w,double h,String s)
{
length=l;
width=w;
height=h;
color=s;
}
public double volume()
{ return height*width*length;}

public String output()
{ return "volume = "+volume()+"color = "+color;}
}
```

```
public class array4
{
public static void main(String arg[])
{
box3 x[]=new box3[2];
x[0]=new box3(1.2,2.3,2.5,"yellow");
x[1]=new box3(1.1,1.3,1.5,"blue");
String s="";
for(int i=0;i<x.length;i++)
{
System.out.println("volume of x["+i+"] = "+x[i].volume());
}
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array4
volume of x[0] = 6.8999999999999995
volume of x[1] = 2.1450000000000005
> Terminated with exit code 0.
```

EXAMPLE 5 : Create a complex class array and calculate sum of the two complex array element through the static method definition

```
public class complex
{
public double real;
public double imag;
public complex(double r,double i)
{real=r;imag=i;}
//static addition of two complex variable result returns as complex
public static complex add(complex z1,complex z2)
{complex z3=new complex((z1.real+z2.real),(z1.imag+z2.imag));
return z3;
}
```

```
//dynamic addition to a complex variable into the value of the class
public void add(complex z1)
{real+=z1.real;
imag+=z1.imag;
}

public String output()
{String s=real+" +i*"+imag;
return s;
}
}
```

```
public class array5
{
public static void main(String arg[])
{
complex x[]=new complex[2];
x[0]=new complex(1.2,2.3);
x[1]=new complex(1.1,1.3);
complex z=complex.add(x[0],x[1]);
System.out.println(z.output());
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array4
volume of x[0] = 6.8999999999999995
volume of x[1] = 2.1450000000000005
> Terminated with exit code 0.
```

EXAMPLE 6 : Create a complex class array and calculate sum of the two complex array element through the dynamic method definition

```
public class array6
{
public static void main(String arg[])
{
complex x[]=new complex[2];
x[0]=new complex(1.2,2.3);
x[1]=new complex(1.1,1.3);
x[0].add(x[1]);
System.out.println(x[0].output());
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array4
volume of x[0] = 6.8999999999999995
volume of x[1] = 2.1450000000000005
```

```
> Terminated with exit code 0.
```

EXAMPLE 7 : Create a polynomial function class and calculate polynomial values for x=1..10

```
public class f
{// General polynomial function
 // y=a[0]+a[1]*x+a[2]*x^2+...
 public double a[];
 public f(double ia[])
 {int n=ia.length;
  a=new double[n];
  for(int i=0;i<n;i++) {a[i]=ia[i];}
 }
 public double func(double x)
 {
  int n=a.length;
  double ff=a[n-1];
  for(int i=n-2;i>=0;i--)
  {ff=ff*x+a[i];}
  return ff;
 }
}
```

```
public class array7
{
public static void main(String arg[])
{
double a[]={1.0,2.0,3.0};
f z=new f(a);
for(double x=0;x<=10;x+=1)
{System.out.println("x="+x+"y="+z.func(x));}
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" array7
x=0.0y=1.0
x=1.0y=6.0
x=2.0y=17.0
x=3.0y=34.0
x=4.0y=57.0
x=5.0y=86.0
x=6.0y=121.0
x=7.0y=162.0
x=8.0y=209.0
x=9.0y=262.0
x=10.0y=321.0

> Terminated with exit code 0.
```

WEEK 8 CLASS EXERSIZES (Will be shown at the and of Lab class)

EX1) enter a double array of 10 numbers as
double x[]={1.2,3.3,4.76,….,1.7465};
calculate the avarage value of x values.

EX2) write a small test program to calculate square of z[0]=1+2i , z[1]=1.2+3i, z[2]=0.2+0.1i, z[3]=1+i using class complex. In your example program enter 4 complex number into an array of complex number, such as
complex z[]=new complex[4];
z[0]=new complex(1.0,2.0);
…
z[3]= new complex(1.0,1.0);
then add them together by using a for loop, print out the result

EX3) By using 3 dimensional cartesian coordinate system, write class point3D P(x,y,z) (similar to class point), Then assign P(1,1,1); P(1,2,1), P(1,0,0) and P(2,1,2). Calculate the total distance form P[0] to P[3]
Point3D P[]=new Point3D[4];
P[0]= new Point3D(1.0,1.0,1.0);
…
P[3]= new Point3D(2.0,1.0,2.0);
Note : writing of this class was assigned to you last week

WEEK 8 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) A method to multiply two matrices. Include these methods to class matrix and use it to multiply two matrices in an example program.

```
public static double[][] multiply(double[][] left,double[][] right)
{
//multiplication of two matrices
int ii,jj,i,j,k;
int m1=left[0].length;
int n1=left.length;
int m2=right[0].length;
int n2=right.length;
double[][] b;
b=new double[m1][n2];
 if(n1 != m2)
 {
 System.out.println("inner matrix dimensions must agree");
 for(ii=0;ii<n1;ii++)
   {
   for(jj=0;jj<m2;jj++)
     b[ii][jj]=0;
   }
   return b;
 }
 for(i=0;i<m1;i++)
   {
   for(j=0;j<n2;j++)
     {
     for(k=0;k<n1;k++)
```

```
                b[i][j]+=left[i][k]*right[k][j];
    }
  }
return b;
//end of multiply of two matrices
}
```

HW2 Write a small test program to calculate multiplications of z[0]=1+2i , z[1]=1.2+3i, z[2]=0.2+0.1i, z[3]=1+i using class complex. In your example program enter 4 complex number into an array of complex muber, such as

complex z[]=new complex[4];
z[0]=new complex(1.0,2.0);
…
z[3]= new complex(1.0,1.0);

then multiply them together by using a for loop, print out the result. Which methor are you going to use to multiply this complex numbers?

HW3 Write a method to multiply two vectors
public static double[] multiply(double[] left,double[] right) and and use it to multiply two vectors in an example program.

9 POLYMORPHISM, ABSTRACT CLASSES AND INTERFACE

9.1 ABSTRACT CLASS

It is seen in chapter 7 and 8 that when classes extended from each other, base class can be used as an indirect reference adres. This property is quite useful when a code is desired to generalise for different classes. In order to establishe this an emty class, called abtract class, can be used as base reference and programs and classes can be developed by using this class. When, in later stages, a new class is developed, this classes can be used with the new class due to this indirect reference possibility.

Definition of abstract class

abstact class class_name
{
class variable list
class method list
}

Class methods can also be abstract in an abstract class. In an abstract method no definition is given

As the first example class shape will be given

Program 9.1.1 Abstract class shape

```
public abstract class shape
{ public double area(){return 0.0;}
  public double volume() {return 0.0;}
  public abstract String name();
}
```

In this class no abstract method is defined, all methods have same definition. As a second example class f_x is given

Program 9.1.2 Abstract class f_x

```
abstract class f_x
{ abstract double func(double x);}
```

In this class, method of the class is defined also as abstract so no class definition is given. If a method definition is given as abstract, definition of the method must be given in the extended method with the same input and output structure. To see how extended classes can be established and used, classes point and circle and cylinder is created as extended from class shape.

Program 9.1.3 class point extended from class shape

```
public class point extends shape
{   protected double x, y;
    public point(double a, double b)
    {  x=a;y=b;}
    public void enterpoint(double a, double b)
    {x=a;y=b;}
    public double read_x()
```

```
    {return x;}
    public double read_y()
    {return y;}
    public String toString()
    {return "["+x+","+y+"]";}
    public String name() {return "point";}
}
```

Program 9.1.4 class circle extended from class point

```
public class circle extends point
{   protected double R;
    public circle(double r, double a, double b)
    {  super(a,b);enterR(r);}
    public void enterR(double r)
    {   if(r >= 0.0)  R=r;
        else          R=0.0;}
    public double read_R()
    { return R;}
    public double area()
    { return Math.PI*R*R;}
    public String toString()
    { return "Center =  "+super.toString()+"; Radius ="+R;}
    public String isim() {return "circle";}
}
```

Program 9.1.5 class cylinder extended from class circle

```
public class cylinder extends circle
 {protected double H;
 public cylinder( double h, double r, int a, int b)
 {super(r, a, b );enterH( h);}
 public void enterH(double h )
 {if(h>=0) H=h;
  else H=0; }
 public double read_H(){return H;}
 public double area()
 { return 2* super.area()+2*Math.PI*R*H;}
 public double volume(){return super.area()*H;}
 public String toString()
 {return super.toString()+";Height = "+H;}
 public String isim(){return "Cylinder";}
}
```

When these classes needs to be used the abstract class shape can be used to get Access to any of these classes.

Program 9.1.6 output example class H9Ex1

```
import javax.swing.*;
class H9Ex1 {
public static void main(String args[])
{point     n   = new point(7,11);
 circle    d   = new circle(3.5,22,8);
 cylinder  s   = new cylinder(10,3.3,10,10);
 shape     a[] = new shape[3];
 a[0]=n;
 a[1]=d;
 a[2]=s;
```

```
 String s2="direct call : \n";
       s2+=n.name()+": "+n.toString()+"\n";
       s2+=d.name()+": "+d.toString()+"\n";
       s2+=s.name()+": "+s.toString()+"\n";
       s2+="indirect call : \n";
 for(int i=0;i<a.length;i++)
 {s2+=a[i].name()+": "+a[i].toString()+"\n";}
 String s1="indirect adress use";
 JOptionPane.showMessageDialog(null,s2,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

As it is seen acces to this classes can be achived throyugh the abstract class shape. As another exampel class numerical is given. In this class two mathematical method is given one to get derivative, another to integrate a function. While programsa re written actual functions to be used is not defined, theredore method definitions are utilises abstract class f_x and all programs are written for this abstract class.

Program 9.1.7 class numerical, uses abstract class f_x and abstract method func as a general function definition.

```
public class numerical
{

public static double derivative(f_x f,double x)
{  double h=0.00001;
  return (-f.func(x+2.0*h)+8.0*f.func(x+h)-8.0*f.func(x-h)+f.func(x-2.0*h))/(12.0*h);
}

public static double integral(f_x f_xnt,double a,double b)
{
//integral f(x)dx
 double r[]={-0.973906528517171,-0.865063366688984,-0.679409568299024,-0.433395394129247,-0.148874338981631,
 0.148874338981631,0.433395394129247,0.679409568299024,0.865063366688984,0.973906528517171};
 double c[]={0.066671344308684,0.149451349150580,0.219086362515982,0.269266719309996,0.295524224714752,
 0.295524224714752,0.269266719309996,0.219086362515982,0.149451349150580,0.066671344308684};
double z=0,x,y;
double k1=(b-a)/2.0;
double k2=(b+a)/2.0;
for(int i=0;i<r.length;i++)
{
x=k2+k1*r[i];
y=f_xnt.func(x);
z+=k1*c[i]*y;
}
return z;
}
}
```

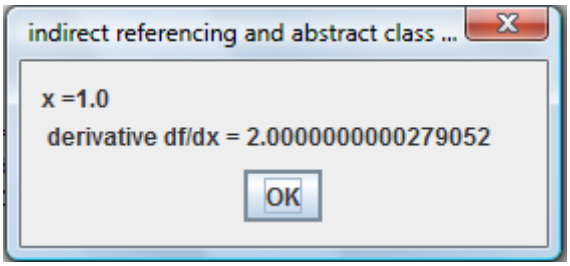
When the methods in the numerical class is needed to be used, the actual function is also known. Therefore it can be defined when user is needed to be use it and used previously defined class numerical.

Program 9.1.8 An example class uses abstract class f_x and numerical and derive the new class for the fuction definition

```
import javax.swing.*;
```

```
class fb extends f_x
{ public double func(double x)
   { return x*x-5.0;}}

public class H9Ex3 {
public static void main(String args[])
{ fb ff=new fb();
  double x=1.0;
 String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(ff,x);
 String s1=" indirect referencing and abstract class example :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

In the example program class fb is defined as extended from class f_x and used with derivative static method of numerical class. The program output :



Program 9.1.9 An example class uses abstract class f_x and numerical and derive the new class fort he fuction definition

```
import javax.swing.*;

class fb extends f_x
{ public double func(double x)
   { return x*x-5.0;}}

public class H9Ex3b {
public static void main(String args[])
{  double x=1.0;
   String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(new fb(),x);
   String s1=" indirect referencing and abstract class example :";
   JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

In the second example instead of defining a new instance object of class fb, and then calling it into the derivative method, the new object directly created in the method call.

Program 9.1.10 An example class uses abstract class f_x and numerical and derive the new class for the fuction definition

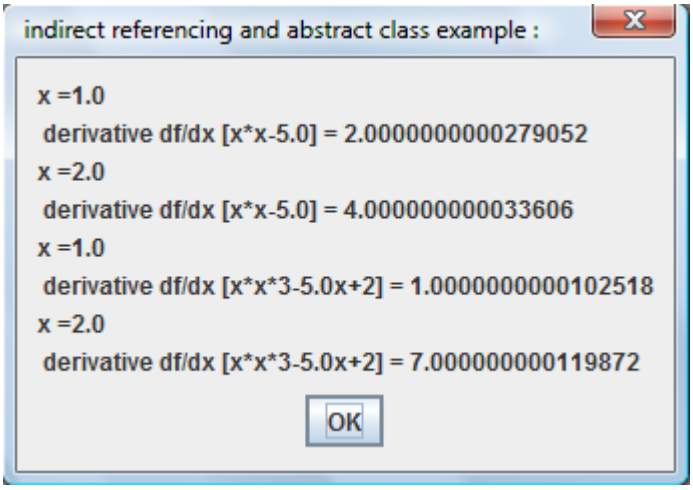
```
import javax.swing.*;

class fb extends f_x
{ public double func(double x)
   { return x*x-5.0;}}

public class H9Ex3b {
```

```
public static void main(String args[])
{  double x=1.0;
   String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(new fb(),x);
   String s1=" indirect referencing and abstract class example :";
   JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

In the third example instead of one class for a fuction definition, two different class with two different function is defined.

Program 9.1.11 An example class uses abstract class f_x and numerical and derive the new class for the fuction definition

```
import javax.swing.*;

class fb extends f_x
{ public double func(double x)
   { return x*x-5.0;}}

class fc extends f_x
{ public double func(double x)
   { return x*x*3-5.0*x+2;}}


public class H9Ex3c {

public static void main(String args[])
{
    double x=1.0;
            String s="x ="+x+"\n derivative df/dx [x*x-5.0] = "+numerical.derivative(new fb(),x)+"\n";
            x=2;
            s+="x ="+x+"\n derivative df/dx [x*x-5.0] = "+numerical.derivative(new fb(),x)+"\n";
            x=1;
            s+="x ="+x+"\n derivative df/dx [x*x*3-5.0x+2] = "+numerical.derivative(new fc(),x)+"\n";
            x=2;
    s+="x ="+x+"\n derivative df/dx [x*x*3-5.0x+2] = "+numerical.derivative(new fc(),x)+"\n";

   String s1=" indirect referencing and abstract class example :";
   JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```



In our final example, in order to avoid the lengthly definition of function call and output preperation each time a method is defined fort his duty.

Program 9.1.12 An example class uses abstract class f_x and numerical and derive the new class for the fuction definition

```
import javax.swing.*;

class fb extends f_x
{ public double func(double x)
  { return x*x-5.0;}}

class fc extends f_x
{ public double func(double x)
  { return x*x*3-5.0*x+2;}}


public class H9Ex3d {

public static String toString(f_x f,String s,double x)
{ String s1="x ="+x+"\n derivative df/dx [s] = "+numerical.derivative(f,x)+"\n";
 return s1;
}

public static void main(String args[])
{
   String s=toString(new fb(),"x*x-5.0",1.0);
   s+=toString(new fb(),"x*x-5.0",2.0);
   s+=toString(new fc(),"x*x*3-5.0x+2",1.0);
   s+=toString(new fb(),"x*x*3-5.0x+2",2.0);
   String s1="indirect adress usage :";
   JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

9.2 INTERFACE

Extend statement can be used only once. One class can be extended only by another one class. From time to time more extensions might be needed. For this purpose, a definition similar to abstract class definition is given in java. It is called interface. The basic difference of interface is not only the

possibility of using for more than one classes, but also having all methods as empty methods(like abstract methods mentioned above but without word abstract). Not that interfaces can nat be considered as class by themselves. They are rather templates of classes to define how and with which method and variables derived classes should be defined.

Definition of interface

```
interface class_name
{
class variable list
class method list
}
```

Class methods can also be abstract in an abstract class. In an abstract method no definition is given. As the first example interface shape will be given

Program 9.2.1 interface shape

```
public interface shape
{public double area();
 public double volume();
 public String name();
}
```

Compare abstract class and interface shape definitions. No method body is existed for area, volume or name methods (all methods of interface sre abstract type automatically) . As a second example interface f_x is given

Program 9.2.2 interface f_x

```
public interface f_x
{ double func(double x);}
```

In this interface like all other interfaces, no method definition is given. To see how a method is written by using interface and used, classes point and circle and cylinder is created as extended from class shape. The word **implements** is used in interface connection istead of word **extend**.

Program 9.2.3 class point implemented from interface shape

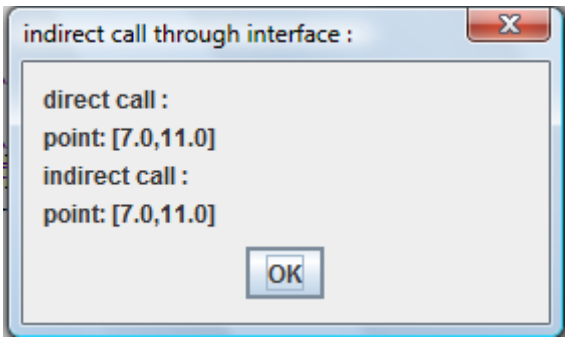
```
public class point implements shape
{ double x, y;
   public point(double a, double b) {  x=a;y=b;}
   public void enter_point(double a, double b) {x=a;y=b;}
   public double read_X() {return x;}
   public double read_Y() {return y;}
   public String toString(){ return "["+x+","+y+"]";}
   public double area() {return 0.0;}
   public double volume(){return 0.0;}
   public String name() {return "point";}
}
```

When these classes needs to be used the interface shape can be used to get Access to class point

Program 9.2.4 output example class H9Ex2

```
import javax.swing.*;
```

```
class H9Ex2 {
public static void main(String args[])
{ point    n = new point(7,11);
  shape    a;
  a=n;
  String s="direct call : \n";
        s+=n.name()+": "+n.toString()+"\n";
        s+="indirect call : \n";
        s+=a.name()+": "+a.toString()+"\n";
  String s1="indirect call through interface : ";
  JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

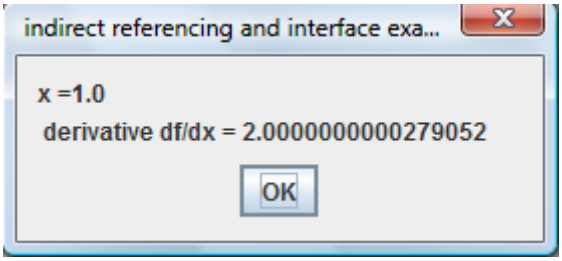


As it is seen acces to class point can be achived through the interface shape. As another example class numerical is given. Code in class numerical will be exatly same as the previous one, therefore class is not re-listed here, but usage of it will be different.

Program 9.2.5 An example class uses interface f_x and numerical and derive the new class for the fuction definition

```
import javax.swing.*;

class fb implements f_x
{ public double func(double x)
   { return x*x-5.0;}}

public class H9Ex3 {
public static void main(String args[])
{ fb ff=new fb();
  double x=1.0;
 String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(ff,x);
 String s1=" indirect referencing and interface example :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

In the example program class fb is defined as **implemented** from class f_x and used with derivative static method of numerical class. The program output :



As it is seen from the example, usage of the interface is very similar to abstract class. Until Java 8 no defined methods or variables are allowed in the interface class and all methods and variable defined in interface must be defined also in the implemented class. But Java 8 on, it is possible to define methods in interfaces by adding word **default**. But in order to use it, interface should be implemented in another class.

Program 9.2.6 An example class uses interface person with a default method sayHello

```
interface Person {
   //adds a java 8 default method
   default String sayHello() {
      return "Hello there!";
   }
}

class Sam implements Person {}

public class inherit1 {
   public static void main(String [] args) {
      Sam sam = new Sam();
      //calling sayHello method calls the method
      //defined in interface
      System.out.println(sam.sayHello());
   }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" inherit1
Hello there!

> Terminated with exit code 0.
```

9.3 INNER CLASSES AND LAMBDA EXPRESSIONS

Consider interface f_x and H9Ex3 defined in previous section. In H9Ex3 class fb is defined as implemented from interface f_x. In java classes can be written as innner classes instead of seperate outside classes.

Program 9.3.1 An example class uses interface f_x and numerical and derive the new inner class for the function definition

```
import javax.swing.*;
public class H9Ex3_1 {
public static void main(String args[])
{     class fb implements f_x
```

```
    { public double func(double x) { return x*x-5.0;}}

 fb ff=new fb();
 double x=1.0;
 String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(ff,x);
 String s1=" indirect referencing and interface example with inner class :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

In java 8 on a new short-cut definition of the previous example can be written as:

Program 9.3.2 An example class uses interface f_x and numerical and derive the new inner class in lambda expression form for the function definition

```
import javax.swing.*;
public class H9Ex3_2 {
public static void main(String args[])
{
 f_x ff = x -> x*x-5.0;
 double x=1.0;
 String s="x ="+x+"\n derivative df/dx = "+numerical.derivative(ff,x);
 String s1=" indirect referencing and interface example with inner class lambda definition:";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}}
```

This particular short form is called **lambda expressions.**
Let us give a second example to further explain the consept

Program 9.3.3 An interface f_boolean

```
public interface f_boolean
{ boolean decision(double x,double y);}
```

Program 9.3.4 An example class uses interface f_boolean and class fc

```
import javax.swing.*;
class fc implements f_boolean
   { public boolean decision(double x,double y)
     { return x>y;}
   }
public class H9Ex4_1 {
public static void main(String args[])
{
 fc ff=new fc();
 double x=1.0;
 double y=2.0;
 String s=x+" > "+y+" = "+ff.decision(x,y);
 String s1=" indirect referencing and interface example :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
```

```
}
}
```

Program 9.3.5 An example class uses interface f_boolean and inner class fc

```
import javax.swing.*;

public class H9Ex4_2 {
public static void main(String args[])
{   class fc implements f_boolean
    { public boolean decision(double x,double y)
      { return x>y;}
    }
 fc ff=new fc();
 double x=1.0;
 double y=2.0;
 String s=x+" > "+y+" = "+ff.decision(x,y);
 String s1=" indirect referencing and interface example :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

Program 9.3.6 An example class uses interface f_boolean and inner class fc as lambda expression

```
import javax.swing.*;

public class H9Ex4_3 {
public static void main(String args[])
{
 f_boolean ff=(x,y)->x>y;
 double x=1.0;
 double y=2.0;
 String s=x+" > "+y+" = "+ff.decision(x,y);
 String s1=" indirect referencing and interface example :";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}
}
```

It should be note that in order to lambda expressions work, there should be only one method definition in interface class new inner class connected. This kind of interfaces called **functional interfaces**. A functional interface is an interface with a single abstract method that is used as the type of a lambda expression. It is not required but function interfaces can be specified in Java 8 by using **@FunctionalInterface** word in front of the interface. In the next example such an interface is specified and used.

Program 9.3.7 An example class uses WorkerInterface as lambda expression and Anonymous inner class

```
 //define a functional interface
@FunctionalInterface
```

```
interface WorkerInterface {
   public void doSomeWork();
}
public class WorkerInterfaceTest {

   public static void execute(WorkerInterface worker) {
      worker.doSomeWork();
   }

   public static void main(String [] args) {
      //invoke doSomeWork using Annonymous inner class
      execute(new WorkerInterface() {
         @Override
         public void doSomeWork() {
            System.out.println("Worker invoked using Anonymous class");
         }
      });
      //invoke doSomeWork using Lambda expression
      execute( () -> System.out.println("Worker invoked using Lambda expression") );
   }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" WorkerInterfaceTest
Worker invoked using Anonymous class
Worker invoked using Lambda expression

> Terminated with exit code 0.
```

Another version of the same program:

Program 9.3.8 An example class uses WorkerInterface as lambda expression and Anonymous inner class

```
 //define a functional interface
@FunctionalInterface
interface WorkerInterface {
   public void doSomeWork();
}

public class WorkerInterfaceTest1 {
   public static void main(String [] args) {
      WorkerInterface worker=()->System.out.println("Worker invoked using Lambda expression ");
      worker.doSomeWork();
   }
}
```

As a last exercises, combining default interface methods usage with lambda expressions will be introduced. Previously, interface used together with numeric class to calculate integral and derivative of the function. This can be done by using default derivative and integral methods inside the interface class.

Program 9.3.9 An example class defines interface inter_f_x

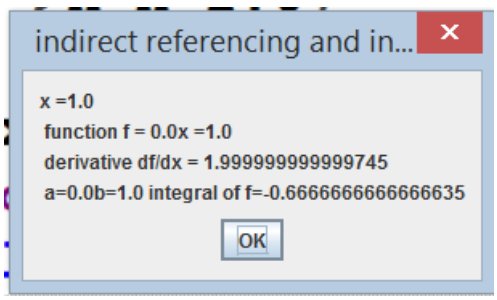
```
 import static java.lang.Math.*;
```

```
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;

@FunctionalInterface
interface inter_f_x
{
//Function
public double func(double x);

default double dfunc(double x)
{double h=1.0e-3;
 int n=1;
 return dfunc_n(x,n,h);
}
default double dfunc(double x,double h)
{int n=1;
 return dfunc_n(x,n,h);
}
default double dfunc(double x,int n)
{double h=1.0e-3;
 return dfunc_n(x,n,h);
}
default double dfunc_n(double x,int n,double h)
{
double hh=1/h;
double df=0;
if(n==0) df=func(x);
else if(n==1)
{df=(3.0*func(x-4.0*h)-32.0*func(x-3.0*h)+168.0*func(x-2.0*h)-672.0*func(x-h)+672.0*func(x+h)-
168.0*func(x+2.0*h)+32.0*func(x+3.0*h)-3.0*func(x+4.0*h))/840.0*hh;}
else if(n==2)
{df=(-14350.0*func(x)-9.0*func(x-4*h)+128*func(x-3*h)-1008*func(x-2*h)+8064*func(x-h)+8064.0*func(x+h)-
1008.0*func(x+2.0*h)+128.0*func(x+3.0*h)-9.0*func(x+4.0*h))/5040.0*hh*hh;}
else if(n==3)
{df=(-7.0*func(x-4.0*h)+72.0*func(x-3.0*h)-338.0*func(x-2.0*h)+488.0*func(x-h)-
488.0*func(x+h)+338.0*func(x+2.0*h)-72.0*func(x+3.0*h)+7.0*func(x+4.0*h))/240.0*hh*hh*hh;}
else if(n==4)
{df=(2730.0*func(x)+7.0*func(x-4.0*h)-96.0*func(x-3.0*h)+676.0*func(x-2*h)-1952.0*func(x-h)-
1952.0*func(x+h)+676.0*func(x+2.0*h)-96.0*func(x+3.0*h)+7.0*func(x+4.0*h))/240.0*hh*hh*hh*hh;}
else if(n==5)
{df=(func(x-4.0*h)-9.0*func(x-3.0*h)+26.0*func(x-2.0*h)-29.0*func(x-h)+29.0*func(x+h)-
26.0*func(x+2.0*h)+9.0*func(x+3.0*h)-func(x+4.0*h))/6.0*hh*hh*hh*hh*hh;}
else if(n==6)
{df=(-150.0*func(x)-func(x-4.0*h)+12.0*func(x-3.0*h)-52.0*func(x-2.0*h)+116.0*func(x-h)+116.0*func(x+h)-
52.0*func(x+2.0*h)+12.0*func(x+3.0*h)-func(x+4.0*h))/4.0*hh*hh*hh*hh*hh*hh;}
else if(n==7)
{df=(-func(x-4.0*h)+6.0*func(x-3.0*h)-14.0*func(x-2.0*h)+14.0*func(x-h)-14.0*func(x+h)+14.0*func(x+2.0*h)-
6.0*func(x+3.0*h)+func(x+4.0*h))/2.0*hh*hh*hh*hh*hh*hh*hh;}
else if(n==8)
```

```
{df=(70.0*func(x)+func(x-4.0*h)-8.0*func(x-3.0*h)+28.0*func(x-2.0*h)-56.0*func(x-h)-
56.0*func(x+h)+28.0*func(x+2.0*h)-8.0*func(x+3.0*h)+func(x+4.0*h))*hh*hh*hh*hh*hh*hh*hh*hh;}
else df=0;
return df;
}
//INTEGRAL
default double integral(double a,double b)
{ //10 point Gauss-Legendre formula
  //integral f(x)dx
 double r[]={-0.973906528517171,-0.865063366688984,-0.679409568299024,-0.433395394129247,-
0.148874338981631,
 0.148874338981631,0.433395394129247,0.679409568299024,0.865063366688984,0.973906528517171};
 double c[]={0.066671344308684,0.149451349150580,0.219086362515982,0.269266719309996,0.295524224714752,
 0.295524224714752,0.269266719309996,0.219086362515982,0.149451349150580,0.066671344308684};
double z=0,x,y;
double k1=(b-a)/2.0;
double k2=(b+a)/2.0;
for(int i=0;i<r.length;i++)
{
x=k2+k1*r[i];
y=func(x);
z+=k1*c[i]*y;
}
return z;
}
}
```

Program 9.3.10 An example class uses interface inter_f_x as lambda expression

```
import javax.swing.*;
public class inter_fx_Ex4{
public static void main(String args[])
{ inter_f_x ff=x->x*x-1.0;
 double x=1.0;
 String s="x ="+x+"\n function f = "+ff.func(x);
 s+="x ="+x+"\n derivative df/dx = "+ff.dfunc(x);
 double a=0;double b=1.0;
 s+="\n a="+a+"b="+b+" integral of f="+ff.integral(a,b);
 String s1=" indirect referencing and interface example with inner class lambda definition:";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}}
```



Program 9.3.11 An example class uses interface inter_f_x as outside class definition

```
 import javax.swing.*;
  class ff1 implements inter_f_x
  {public double func(double x)
  {return x*x-1.0;}
  }
public class inter_fx_Ex1{
public static void main(String args[])
{ ff1 ff=new ff1();
 double x=1.0;
 String s="x ="+x+"\n function f = "+ff.func(x);
 s+="x ="+x+"\n derivative df/dx = "+ff.dfunc(x);
 double a=0;double b=1.0;
 s+="\n a="+a+"b="+b+" integral of f="+ff.integral(a,b);
 String s1=" indirect referencing and interface example with inner class lambda definition:";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}}
```

This concept can be used similarly if the input variable is an array instead of a single variable.

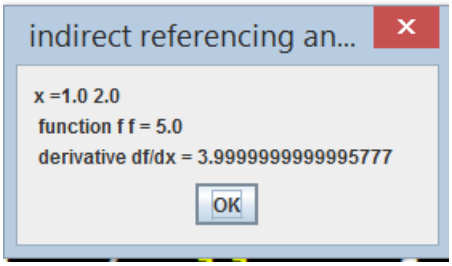
Program 9.3.12 An example class defines interface if_xj

```
@FunctionalInterface
interface if_xj
{
// single function  multi independent variable
// a single value is returned indiced to equation_ref
public double func(double x[]);
default double dfunc(double x[],int x_ref)
{ // derivative of the function with respect to x_ref
double h0=0.0256;
int i,m;
int n=7;
double f1,f2;
double x1[];
x1=new double[x.length];
double x2[];
x2=new double[x.length];
for(i=0;i<x.length;i++)
{
x1[i]=x[i];
x2[i]=x[i];
}
//derivative of a simple function
double T[][];
T=new double[n][n];
double h[];
h=new double[n];
//vector<double> h(n,0);
for(i=0;i<n;i++)
{
  h[i]=0;
  for(int j=0;j<n;j++)
    T[i][j]=0;
```

```
}
h[0]=h0;
double r=0.5;
for( i=1;i<n;i++)
{
h[i]=h0*Math.pow(r,i);
}

for(i=0;i<n;i++)
{
x1[x_ref]+=h[i];
x2[x_ref]-=h[i];
f1=func(x1);
f2=func(x2);
T[i][0]=( f1 - f2)/(2.0*h[i]);
x1[x_ref]=x[x_ref];
x2[x_ref]=x[x_ref];
}
for(m=1;m<n;m++)
{
  for(i=0;i<n-m;i++)
  {
  T[i][m]=(h[i]*h[i]*T[i+1][m-1] - h[i+m]*h[i+m]*T[i][m-1])/(h[i]*h[i]
  - h[i+m]*h[i+m]);
  }
}
double xx=T[0][n-1];
return xx;
}

}
```

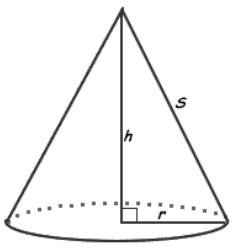
Program 9.3.13 An example class uses interface if_xj as lambda expression

```
import javax.swing.*;
public class ifxj_Ex4 {
public static String array(double x[])
{   String s="";
            for(int i=0;i<x.length;i++) {s+=x[i]+" ";}
            return s;
}
public static void main(String args[])
{
 if_xj ff=x->x[0]*x[0]+x[1]*x[1];
 double x[]={1.0,2.0};
  String s="x ="+array(x)+"\n function f f = "+ff.func(x);
  s+="\n derivative df/dx = "+ff.dfunc(x,1);
 String s1=" indirect referencing and interface example with inner class lambda definition:";
 JOptionPane.showMessageDialog(null,s,s1,JOptionPane.PLAIN_MESSAGE);
}}
```



WEEK 9 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1 investigate abstract class shape and classes point, circle, and cylinder written as extended from abstract class shape. Create a cone.java clas, extended from class cylinder that will be calculated area and volume of a cone **(cone.java) .** Note that this class will be similar to cyclinder class except the way of calculating area and volume

	Area of the conic surface = πrs Area of the base= πr^2 Area of the cone = $\pi rs + \pi r^2$ Volume of the cone = $1/3\ \pi r^2 h$ $s = (r^2 + h^2)^{0.5}$

EX2 Calculate the derivative of function f(x)=x*x-3*x+2 by using class numerical and **abstract class f_x** at point x=1.0

EX3 Calculate the derivative of function f(x)=sin(x) at the point x=π/4 by using class numerical and **interface f_x** fonksiyonunu ve türevini graph1.java programınında verilen fb sınıfını değiştirerek çizdiriniz

EX4 Calculate the integral of function f(x)=sin(x) for the integration limits a=0 and b=π/4 by using class numerical and **interface f_x**

EX5 Calculate the integral of function f(x)=sin(x) for the integration limits a=0 and b=π/4 by using class numerical and **interface f_x** and an inner class **lambda** expression

EX6 Calculate the integral of function f(x)=cos(x) for the integration limits a=0 and b=π/2 by using class **inter_ f_x** and an inner class **lambda** expression

WEEK 9 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) From **abstract class shape** derive class **sphere**. This class should calculate its area and volume , Test it by using indirect reference shape

HW2) From **interface shape** derive class **sphere**. This class should calculate its area and volume , Test it by using indirect reference shape

HW3) Write a test program to read value of x from input then calculate derivative of function

$y=e^{-x}/(x^2-1)$ in a test program

a) By using an outside class **fb implements f_x**
b) By using an inner class **fb implements f_x**
c) By using a lambda expression inner class

HW4 Write a test program to read value of a and b (lower and upper limit of the integral) from input then calculate integral of function

$y=e^{-x}/(x^2-1)$ in a test program

a) By using an outside class **fb implements f_x**
b) By using an inner class **fb implements f_x**
c) By using a lambda expression inner class

10 GRAPHICS

10.1 JAVA SWING GRAPHICS ENVIRONMENT AND JFRAME AND JPANEL CLASSES

In Java, two graphics environment (Graphics application development environment) is defined. The first one is awt. The second one is swing library which came out with more sophisticated drawing facilities. In order to understand java drawing capabilities, the first concept is to understand java coordinate system. In java graphics screen, if anything is drawn a graphic screen coordinates is used. Coordinate unit is pixel(Picture elements) in your graphic window. Every computer graphic window consist of pixels.

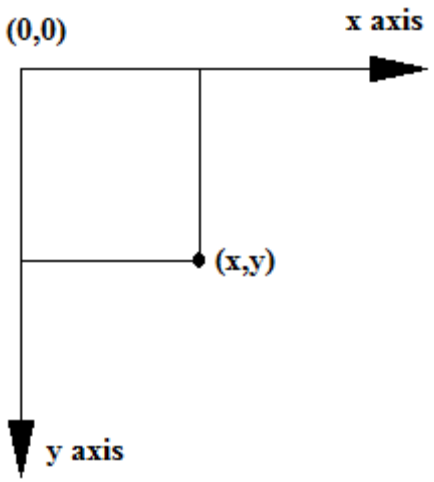


Figure 10.1 Java Graphic coordinate system

In order to graph someting, class JFrame is used. Actual drawings will be prepared in another class JPanel and will be added up into JFrame to show the graphics. In order to show graphics in JFrame, a small interface program FrameGraph is prepared for you.

10.1.1 FrameGraphics interpahese class to show a JPanel into the graphic window

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
public class FrameGraphic extends JFrame
{
 JPanel d;
  public FrameGraphic(String a,JPanel di)
  {   super(a);
      d=di;
      add(d);
  }
  public static void plot(String a,JPanel di)
  {
      FrameGraphic f = new FrameGraphic(a,di);
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
```

```
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

In this class JPanel object d is added up into the JFrame graphic window. Drawing of the graphic is carried out inside of plot method. So using plot method in your main program with the definition of JPanel class is enough to draw someting inside of the window. The first application we will be looking is to write a welcome message in Java graphic window.

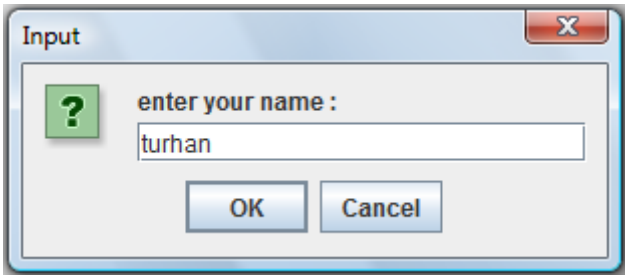
10.1.2 WelcomeP JPanel class to draw a welcome message in the graphic window

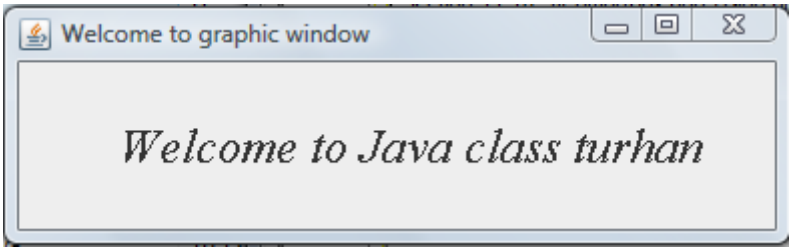
```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class WelcomeP extends JPanel
{
 String isim;
 public WelcomeP()
 {isim=JOptionPane.showInputDialog("enter your name : ");}
 public void paint(Graphics g)
 {
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.ITALIC,24));
 g2.drawString("Welcome to Java class " + isim,50,50);
 }
}
```

In class WelcomeP a name is entered in constructor method. For actual drawing paint or paintComponent method is used. Actual graphic screen is defined by Graphics class(awt) library and Graphics2D class(swing library). setFont method defines which fonts to use to draw characters, and drawString method draws String variable into x,y coordinates. If we use this program in a main program:

10.1.3 An example program to get graphic output WelcomeP JPanel class

```
class H10Ex1 {
public static void main(String args[])
{FrameGraphic.plot("Welcome to graphic window",new WelcomeP());}}
```





Now this structure can be used to draw different graphic environment elements. If we would like to import a paint into the graphic window:

10.1.4 pictureP JPanel class to draw a picture in the graphic window

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;
import java.net.URL;
import java.awt.image.*;
import java.util.*;

public class pictureP extends JPanel
{
 private Image picture;

 public pictureP(String s)
 {super();
  URL url = getClass().getResource(s);
  picture = getToolkit().getImage(url);
 }


 public void paint(Graphics g)
 {
 Graphics2D g2=(Graphics2D)g;
 Dimension d=getSize();
 int dx = d.width;
 int dy = d.height;
 g2.drawImage( picture, 0, 0,dx,dy, this);
 }
}
```

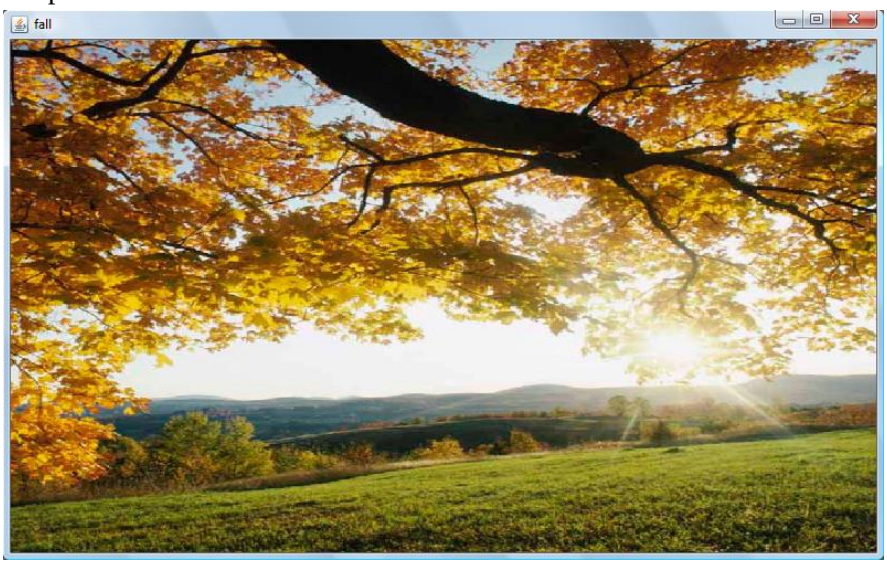
In this class in order to read the Picture data Image class is used together with URL class. Class drawn by using drawImage method of Graphics2D class. The main program calling the picture

10.1.5 pictureP JPanel class to draw a picture in the graphic window

```
class H10Ex6 {
public static void main(String args[])
{
```

```
pictureP pp=new pictureP("fall.jpg");
FrameGraphic.plot("fall",pp);
}
}
```

Output will be:



A different geometric shapes can also be drawn in JPanel environment. In order to draw a line Line2D and draw methods are used. In the following lineP JPanel program setColor method to set the color of the line is used. We also set the background color as yellow. A SetStroke method is also used to define how thick the line is. In this example line is defined as 5 pixel thick by 5.0f.

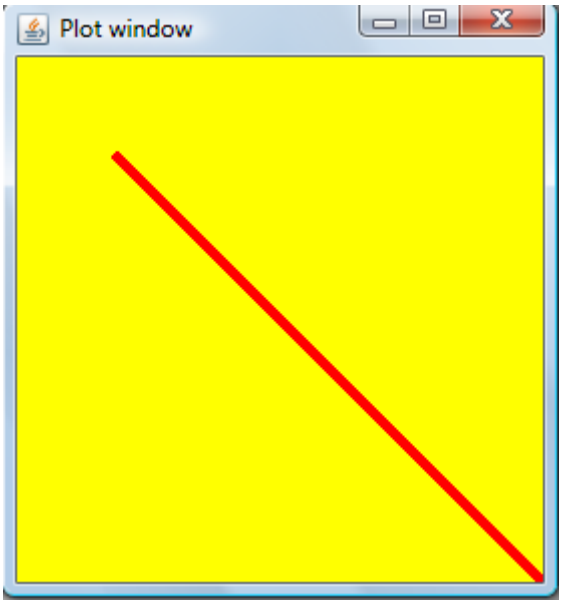
10.1.6 lineP JPanel class to draw a line in the graphic window

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class lineP extends JPanel
{

 public void paintComponent(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 this.setBackground(Color.YELLOW);
 g2.setColor(Color.red);
 g2.setStroke(new BasicStroke(5.0f));
 Line2D x=new Line2D.Double(50,50,800,800);
 g2.draw(x);
 }}
```

10.1.7 example program to draw line by using lineP JPanel class

```
class H10Ex5 {
public static void main(String args[])
{
```

```
lineP pp=new lineP();
FrameGraphic.plot("Plot window",pp);
}
}
```



In this stage we can look at class color more closely, to understand how it can be called. Class Color has the following constructor and Acces classes:

Method **Description**

Color constructors and methods

public Color(int r, int g, int b)

Creates a color based on red, green and blue components expressed as integers from 0 to 255.

public Color(float r, float g, float b)

Creates a color based on red, green and blue components expressed as floating-point values from 0.0 to 1.0.

public int getred()

Returns a value between 0 and 255 representing the red content.

public int getGreen()

Returns a value between 0 and 255 representing the green content.

public int getBlue()

Returns a value between 0 and 255 representing the blue content.

Graphics methods for manipulating Colors

public Color getColor()

Returns Color object representing current color for the graphics context.

public void setColor(Color c)

Sets the current color for drawing with the graphics context.

Some colors are also defined as constant values which can be called directly as **Color.YELLOW**

Color constant	**Color**	**RGB value**
public final static Color RED	Red	255, 0, 0
public final static Color GREEN	Green	0, 255, 0

Color constant	Color	RGB value
public final static Color BLUE	Blue	0, 0, 255
public final static Color ORANGE	Orange	255, 200, 0
public final static Color PINK	Pink	255, 175, 175
public final static Color CYAN	Cyan	0, 255, 255
public final static Color MAGENTA	Magenta	255, 0, 255
public final static Color YELLOW	Yellow	255, 255, 0
public final static Color BLACK	Black	0, 0, 0
public final static Color WHITE	White	255, 255, 255
public final static Color GRAY	Gray	128, 128, 128
public final static Color LIGHT_GRAY	light gray	192, 192, 192
public final static Color DARK_GRAY	dark gray	64, 64, 64

Every color is created from a red, a green and a blue component. Together these components are called **RGB values**. All three RGB components can be integers in the range from 0 to 255, or they can be floating-point values in the range 0.0 to 1.0. The first RGB component specifies the amount of red, the second the amount of green and the third the amount of blue. The larger the RGB value, the greater the amount of that particular color. Java enables the programmer to choose from 256 x 256 x 256 (or approximately 16.7 million) colors. Java has ready to use color choser window, which we can be looked in next chapters.

We can look at to draw a rectangle next. In order to draw rectangle

Rectangle2D R=new Rectangle2D.Double(x,y,width,height);

g2.draw(R);

methods are used. İn here to draw rectangle starting coordinates x and y and width and height is used.

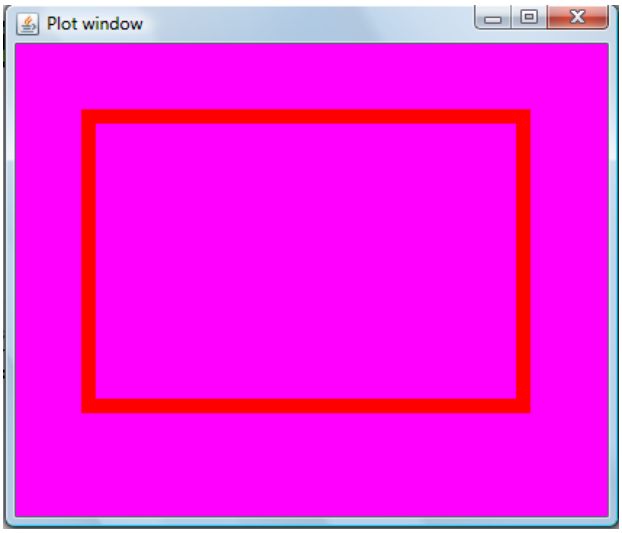
10.1.8 rectangleP JPanel class to draw a rectangle in the graphic window

```
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class rectangleP extends JPanel
{
 public void paintComponent(Graphics g)
 {
 Graphics2D g2=(Graphics2D)g;
 super.paintComponent(g);
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.red);
 this.setBackground(Color.MAGENTA);
 g2.setStroke(new BasicStroke(10.0f));
```

```
 Rectangle2D x=new Rectangle2D.Double(50,50,300,200);
 g2.draw(x);
 }
}
```

10.1.9 example program to draw rectangle by using rectangleP JPanel class

```
class H10Ex7 {
public static void main(String args[])
{
rectangleP pp=new rectangleP();
FrameGraphic.plot("Plot window",pp);
}
}
```



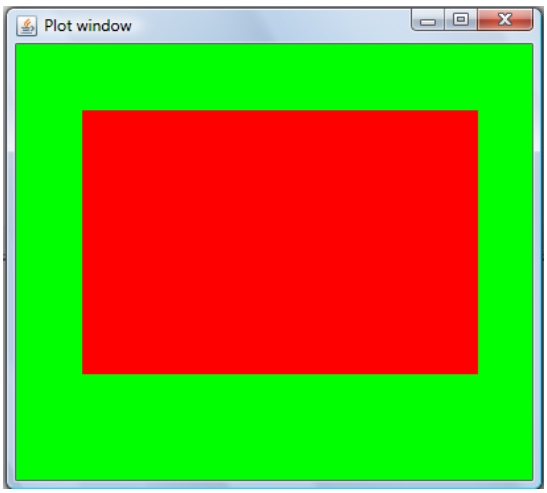
When a closed shape is given instead of **g2.draw(x);** method **g2.fill(x)** can be used to fill the interior of the shape

10.1.10 rectangleP1 JPanel class to draw a filled rectangle in the graphic window

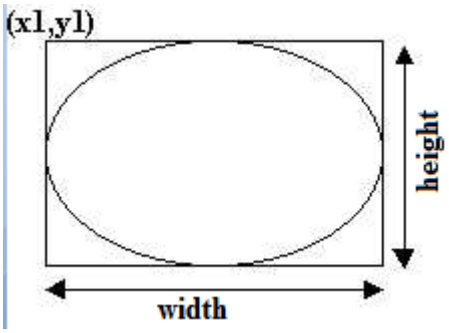
```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class rectangleP1 extends JPanel
{
 public void paintComponent(Graphics g)
 {
 Graphics2D g2=(Graphics2D)g;
 super.paintComponent(g);
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.RED);
 this.setBackground(Color.GREEN);
 g2.setStroke(new BasicStroke(10.0f));
 Rectangle2D x=new Rectangle2D.Double(50,50,300,200);
 g2.fill(x);
 }
}
```

10.1.11 example program to draw rectangle by using rectangleP JPanel class

```
class H10Ex8 {
public static void main(String args[])
{
rectangleP1 pp=new rectangleP1();
FrameGraphic.plot("Plot window",pp);
}
}
```



Now we can look at how to draw elipse. Drawing elipse is basically drawing one into a reactangle



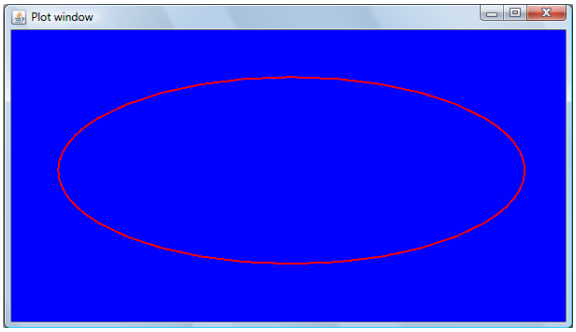
10.1.12 ellipseP1 JPanel class to draw an ellipse in the graphic window

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class ellipseP extends JPanel
{
    public void paintComponent(Graphics g) {
            super.paintComponent(g);
    Graphics2D g2=(Graphics2D)g;
    g2.setFont(new Font("Serif",Font.BOLD,24));
    g2.setColor(Color.RED);
    this.setBackground(new Color(0,0,255));
```

```
    g2.setStroke(new BasicStroke(2.0f));
    Ellipse2D x=new Ellipse2D.Double(50,50,500,200);
    g2.draw(x);
   }
}
```

10.1.13 example program to draw ellipse by using ellipseP JPanel class

```
class H10Ex9 {
public static void main(String args[])
{
ellipseP pp=new ellipseP();
FrameGraphic.plot("Plot window",pp);
}
}
```

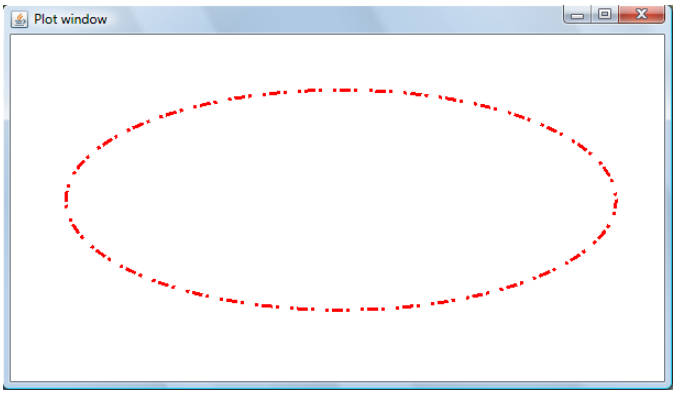


Line characteristics can be changed by changing parameters of BasicStroke

10.1.14 ellipseP JPanel class to draw an ellipse in the graphic window

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class ellipseP extends JPanel
{
    public void paintComponent(Graphics g) {
            super.paintComponent(g);
    Graphics2D g2=(Graphics2D)g;
    g2.setFont(new Font("Serif",Font.BOLD,24));
    g2.setColor(Color.RED);
    this.setBackground(new Color(255,255,255));
    float dash3[] = {10.0f,3.0f,3.0f};
    BasicStroke d3 = new BasicStroke(3.0f,BasicStroke.CAP_BUTT,
                                     BasicStroke.JOIN_MITER,
                                     3.0f, dash3, 2.0f);
    g2.setStroke(d3);
    Ellipse2D x=new Ellipse2D.Double(50,50,500,200);
    g2.draw(x);
   }
}
```

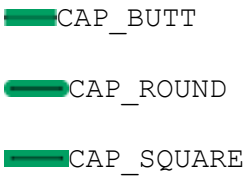
The result will be



In the last program a new concept of dotted line drawing is defined. İn order to draw such a line method **setStroke** is used. An object in class BasicStroke is created to call inside setStroke method. This class contains information about line thickness, line connection style, line end style and line empty space information. Line thickness is measured perpendicular tol ine directions. Line thickness 1f (f here indicated that it is a float type real variable) is equal to 0.35277 milimeter. But if shape is transformed, this can be changed.

Line connection style shows how the lines are connected to each other. There are 3 type line connection styles in BasicStroke class

sitilleridir. Line end styles are makes the end shape of a dotted line. There are also three line end shapes in BasicStroke



Line empty space information showing the width of the shown and invisible parts of the line. As an example given

```
g2.setStroke(new BasicStroke(1f,BasicStroke.CAP_ROUND,

    BasicStroke.JOIN_ROUND,0,new float[] {10,5,2,5},0));
```

Shows a line thickness of 1f, line 10 pixel long follows 5 pixel of an empty space, 2 pixel long line and 5 pixel long empty space

```
g2.setStroke(new BasicStroke(1f,BasicStroke.CAP_ROUND,
```

BasicStroke.JOIN_ROUND,5,new float[] {5},0));

Shows a line thickness of 1f, line 5 pixel long follows 5 pixel of an empty space

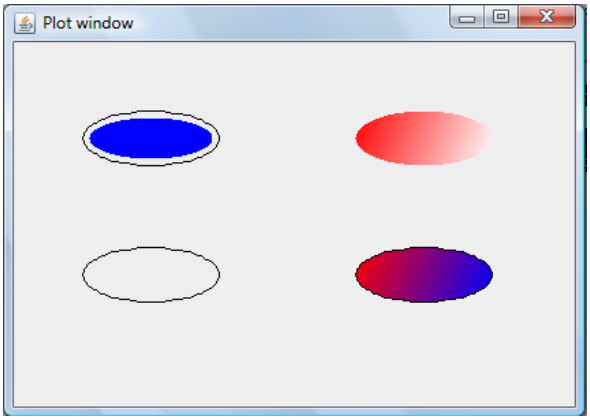
Another interesting feature of the graphics are the gradient paint. Color of a shape can be defined as a changing gradients of colors.

10.1.15 graph2P JPanel class to draw an ellipses with gradient paint in the graphic window

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
public class graph2P extends JPanel
{

    public void paintComponent(Graphics g)
    {  super.paintComponent(g);
       Graphics2D g2 = (Graphics2D)g;
       g2.setPaint(Color.blue);
       Ellipse2D elips1=new Ellipse2D.Double(55,55,90,30);
       g2.fill(elips1);
       g2.setPaint(Color.black);
       Ellipse2D elips2=new Ellipse2D.Double(50,50,100,40);
       g2.draw(elips2);
       g2.setPaint(Color.black);
       Ellipse2D elips3=new Ellipse2D.Double(50,150,100,40);
       g2.draw(elips3);
       GradientPaint kirmizidanbeyaza=new GradientPaint(250,50,Color.red,350,90,Color.white);
       g2.setPaint(kirmizidanbeyaza);
       Ellipse2D elips4=new Ellipse2D.Double(250,50,100,40);
       g2.fill(elips4);
       GradientPaint kirmizidanmaviye=new GradientPaint(250,150,Color.red,350,190,Color.blue);
       g2.setPaint(kirmizidanmaviye);
       Ellipse2D elips5=new Ellipse2D.Double(250,150,100,40);
       g2.fill(elips5);
       g2.setPaint(Color.black);
       g2.draw(elips5);
    }
}
```

10.1.16 example program to draw ellipses with gradint paint by using graph2P JPanel class

```
class H10Ex10 {
public static void main(String args[])
{
graph2P pp=new graph2P();
FrameGraphic.plot("Plot window",pp);
}
}
```



Another interesting paint feature is the texture paint. We can use a Picture as a painting background. For example, fall picture will be used as a TexturePaint in the next elipse drawing

10.1.17 ellipseP2 JPanel class to draw an ellipse in the graphic window with TexturePaind Picture fall.jpg

```
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;
import java.net.URL;
import java.awt.image.*;
import java.util.*;

public class ellipseP2 extends JPanel
{
   TexturePaint tp = getImageTexture("fall.jpg");

   public TexturePaint getImageTexture(String imageFile)
   {
   URL url = getClass().getResource(imageFile);
   Image img = getToolkit().getImage(url);
   try {
     MediaTracker tracker = new MediaTracker(this);
     tracker.addImage(img, 0);
     tracker.waitForID(0);
   } catch (Exception e) {}
   int width = img.getWidth(this);
   int height = img.getHeight(this);
   BufferedImage buffImg = new BufferedImage(width, height, BufferedImage.TYPE_INT_ARGB);
   Graphics g = buffImg.getGraphics();
   g.drawImage(img, 0, 0, this);
   Rectangle2D rect = new Rectangle(0, 0, width, height);
   return new TexturePaint(buffImg, rect);
   }

            public void paintComponent(Graphics g) {
            super.paintComponent(g);
    Graphics2D g2=(Graphics2D)g;
    g2.setPaint(tp);
    g2.setFont(new Font("Serif",Font.BOLD,24));
```

```
    //g2.setColor(Color.RED);
    //this.setBackground(new Color(255,255,255));
    Ellipse2D x=new Ellipse2D.Double(50,50,500,200);
    g2.fill(x);
   }
}
```

10.1.18 example program to draw ellipse with Texture paint by using ellipseP2 JPanel class

```
class H10Ex11 {
public static void main(String args[])
{
ellipseP2 pp=new ellipseP2();
FrameGraphic.plot("Plot window",pp);
}
}
```

When a series of lines existed, instead of drawing one line, the series of lines can be drawn by using General Path.

10.1.19 starP JPanel class to draw a star in the graphic window with GeneralPath type serial line

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class starP extends JPanel
{

public void paint(Graphics g)
{
int x[]={55,67,109,73,83,55,27,37,1,43};
int y[]={0,36,36,54,96,72,96,54,36,36};
Graphics2D g2=(Graphics2D)g;
GeneralPath star=new GeneralPath();
star.moveTo(x[0],y[0]);
for(int i=1;i<x.length;i++)
  {star.lineTo(x[i],y[i]);}
star.closePath();
```

```
g2.setColor(Color.blue);
g2.draw(star);

  }
}
```

10.1.20 example program to draw star with GeneralPath

```
class H10Ex12 {
public static void main(String args[])
{
starP pp=new starP();
FrameGraphic.plot("Plot window",pp);
}
}
```

Instead of lines, a quadratic or coubic connection is also possible. QuadCurve2D will give us a quadratic line connections

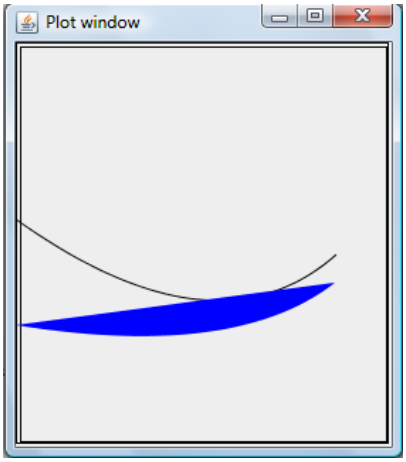
10.1.21 quadcurveP JPanel class to draw a star in the graphic window with QuadCurve2D type serial line

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class quadcurveP extends JPanel
{
   public void paintComponent(Graphics g) {
      super.paintComponent(g);
      Graphics2D g2 = (Graphics2D)g;
      g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
      RenderingHints.VALUE_ANTIALIAS_ON);
      Dimension boyut=getSize();
      int dx=boyut.width;
      int dy=boyut.height;
      g2.setPaint(Color.BLACK);
      g2.draw3DRect(0,0,dx-3,dy-3,true);
      g2.draw3DRect(3,3,dx-7,dy-7,false);
      g2.setPaint(Color.BLACK);
      QuadCurve2D qc2=new QuadCurve2D.Double(0,125,140,225,225,150);
      g2.draw(qc2);
      QuadCurve2D qc2_1=new QuadCurve2D.Double(0,200,155,225,225,170);
      g2.setPaint(Color.blue);
```

```
        g2.fill(qc2_1);
    }

}
```

10.1.22 example program to draw quadratic line with QuadCurve2D

```
class H10Ex13 {
public static void main(String args[])
{
quadcurveP pp=new quadcurveP();
FrameGraphic.plot("Plot window",pp);
}
}
```



A General Curve with cubic and quadratic pieces connected tohgether is given in the next program

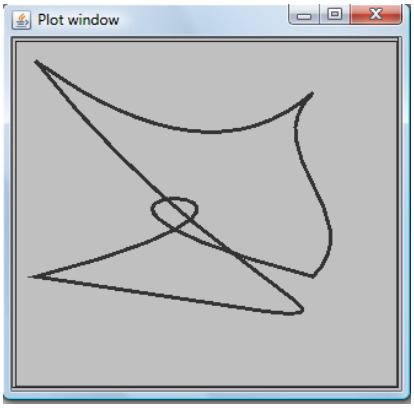
10.1.23 qeneralcurveP JPanel class to draw a star in the graphic window with QuadCurve2D type serial line

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class qeneralcurveP extends JPanel
{
        public void paintComponent(Graphics g) {
        super.paintComponent(g);
                Graphics2D g2 = (Graphics2D)g;
        setBackground(Color.lightGray);
        Dimension boyut=getSize();
        int dx=boyut.width;
        int dy=boyut.height;
        g2.setStroke(new BasicStroke(3));
        g2.draw3DRect(0,0,dx-3,dy-3,true);
        g2.draw3DRect(3,3,dx-7,dy-7,false);
```

```
        GeneralPath shape=new GeneralPath(GeneralPath.WIND_EVEN_ODD);
        shape.moveTo(20,20);
        //quadratik ekleme
        shape.quadTo(160,120,245,45);
        //kübik ekleme
        shape.curveTo(195,95,295,145,245,195);
        shape.curveTo(-80,110,345,110,20,195);
        shape.curveTo(400,250,200,250,20,20);
        g2.draw(shape);
    }
}
```

10.1.24 example program to draw general lines

```
class H10Ex14 {
public static void main(String args[])
{
qeneralcurveP pp=new qeneralcurveP();
FrameGraphic.plot("Plot window",pp);
}
}
```



In the next example, a multi-line text wil be output in the Graphic window.

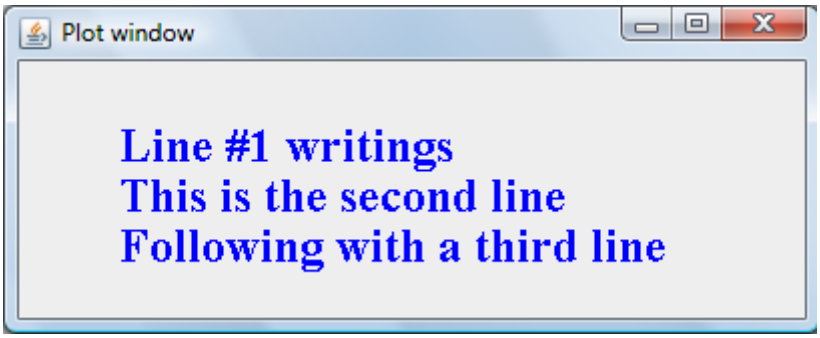
10.1.25 writeP JPanel class to draw a series of lines writing in the graphic window

```
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
public class writeP extends JPanel
{ //grafik penceresine yazı yazar
 String s[];
 public writeP(String si[])
 {super();
 s=new String[si.length];
 for(int i=0;i<si.length;i++){s[i]=si[i];}
 }
```

```
public void paintComponent(Graphics g)
   {   super.paintComponent(g);
            Graphics2D g2 = (Graphics2D)g;
      g2.setFont(new Font("Serif",Font.BOLD,24));
      g2.setColor(Color.blue);
      g2.setBackground(Color.yellow);
      for(int i=0;i<s.length;i++)
            g2.drawString(s[i],50,50+25*i);
   }
}
```

10.1.26 example program to draw several lines of writings

```
class H10Ex15 {
public static void main(String args[])
{
String s[]={"Line #1 writings","This is the second line","Following with a third line"};
writeP pp=new writeP(s);
FrameGraphic.plot("Plot window",pp);
}
}
```



As the last examples, We can combine String drawing method to draw a plot.

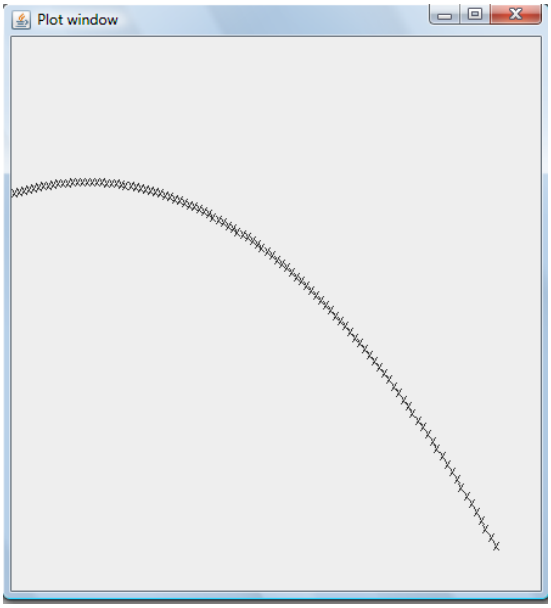
10.1.27 plotP JPanel class to draw a series of x writing following a plot function

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

abstract class f_x
{ abstract double func(double x);}

public class plotP extends JPanel
{
 int x[];
 int y[];
 double xmin;
 double xmax;
 int n;
```

```
 fb f;

 public plotP(f_x f,double xmini,double xmaxi,int width,int height)
 {
  // height : height of the plot window;
  // width  : width of the plot window
  // xmin minimum x value
  // x maximum x value
  xmin=xmini;
  xmax=xmaxi;
  n=100;
  x=new int[n];
  y=new int[n];
  double xd[]=new double[n];
  double yd[]=new double[n];
  double ymin=1.0e60;
  double ymax=-1.0e60;
  for(int i=0;i<n;i++)
  {xd[i] = (xmax-xmin)*(double)i/(double)n;
  yd[i] =  f.func(xd[i]);
  if(yd[i]<ymin) ymin=yd[i];
  if(yd[i]>ymax) ymax=yd[i];
  }
 for(int i=0;i<n;i++)
 {x[i]=(int)(0.8*width*xd[i]/(xmax-xmin));
  y[i]=height/3+(int)(0.6*height*yd[i]/(ymax-ymin));
 }
 }
 public void paintComponent(Graphics g)
 {
 Graphics2D g2=(Graphics2D)g;
 for(int i=0;i<x.length;i++)
  {g2.drawString("x",x[i],y[i]);}
 }

}
```

10.1.28 An example program for plotP

```
import javax.swing.*;

class fb extends f_x
{  public double func(double x)
   { return x*x-3*x-8;}
}

class H10Ex2 {
public static void main(String args[])
{
fb ff=new fb();
plotP pp=new plotP(ff,0.0,10.0,500,500);
//cizgiP pp=new cizgiP();
FrameGraphic.plot("Plot window",pp);
}
}
```



Now lets change the program in small detail and draw line instead of x sign also a rectabngle will be drawn around the plot area

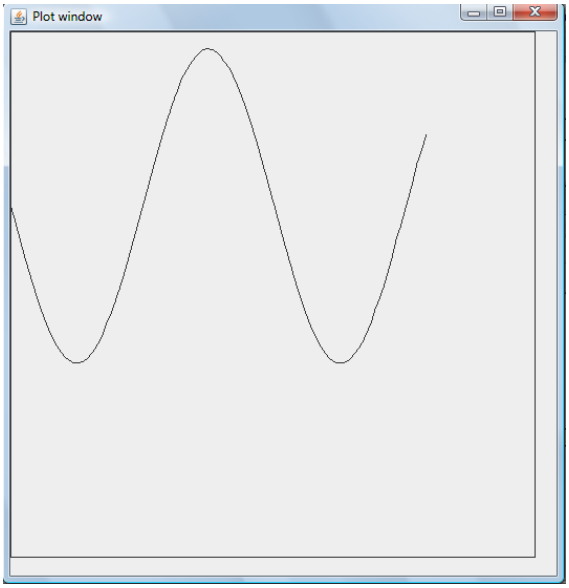
10.1.29 plot2P JPanel class to draw a series of lines writing following a plot function

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

abstract class f_x
{ abstract double func(double x);}

public class plot2P extends JPanel
{
 int x[];
 int y[];
 double xmin;
 double xmax;
 int n;
 fb f;
 int width;
 int height;
 public plot2P(f_x f,double xmini,double xmaxi,int widthi,int heighti)
 {
   // height : height of the plot window;
   // width  : width of the plot window
   // xmin minimum x value
   // x maximum x value
   xmin=xmini;
   xmax=xmaxi;
   width=widthi;
   height=heighti;
   n=100;
   x=new int[n];
   y=new int[n];
   double xd[]=new double[n];
```

```
  double yd[]=new double[n];
  double ymin=1.0e60;
  double ymax=-1.0e60;
  for(int i=0;i<n;i++)
  {xd[i] = (xmax-xmin)*(double)i/(double)n;
  yd[i] =  f.func(xd[i]);
  if(yd[i]<ymin) ymin=yd[i];
  if(yd[i]>ymax) ymax=yd[i];
  }
 for(int i=0;i<n;i++)
 {x[i]=(int)(0.8*width*xd[i]/(xmax-xmin));
  y[i]=height/3+(int)(0.6*height*yd[i]/(ymax-ymin));
 }
}

public void paintComponent(Graphics g)
{
Graphics2D g2=(Graphics2D)g;
for(int i=1;i<x.length;i++)
 { Line2D l=new Line2D.Double(x[i-1],y[i-1],x[i],y[i]);
   g2.draw(l);
 }
Rectangle2D r=new Rectangle2D.Double(0,0,height,width);
g2.draw(r);
}

}
```

10.1.30 An example program for plot2P

```
import javax.swing.*;

class fb extends f_x
{  public double func(double x)
   { return Math.sin(x);}
}

class H10Ex3 {
public static void main(String args[])
{
fb ff=new fb();
plot1P pp=new plot1P(ff,0.0,10.0,500,500);
//cizgiP pp=new cizgiP();
FrameGraphic.plot("Plot window",pp);
}
}
```



10.2 JAVA SWING GRAPHICS ENVIRONMENT AND JAPPLET AND JPANEL CLASSES

Another way of using graphic environment is using internet browser programs as an output channel. In this way java programs can run in internet environment. JApplet programs are similar to JFrame programs, but it has an init method instead of constructor method to start with. As a first example to Applet, plot programs previously defined in the last chapter is redefined

10.2.1 Applet_plot2P JApplet graphic program to show JPanel plot2P

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

class fb extends f_x
{  public double func(double x)
   { return Math.sin(x);}
}

public class  Applet_plot2P extends JApplet
{
  JPanel d;
  public void init()
  {
  fb ff=new fb();
  d=new plot2P(ff,0.0,10.0,500,500);
  add(d);
  }
}
```

In order to use this Aplet program, it should be called internet Hyper Text Mark-up language (HTML) code. A very simle HTML code will look like :

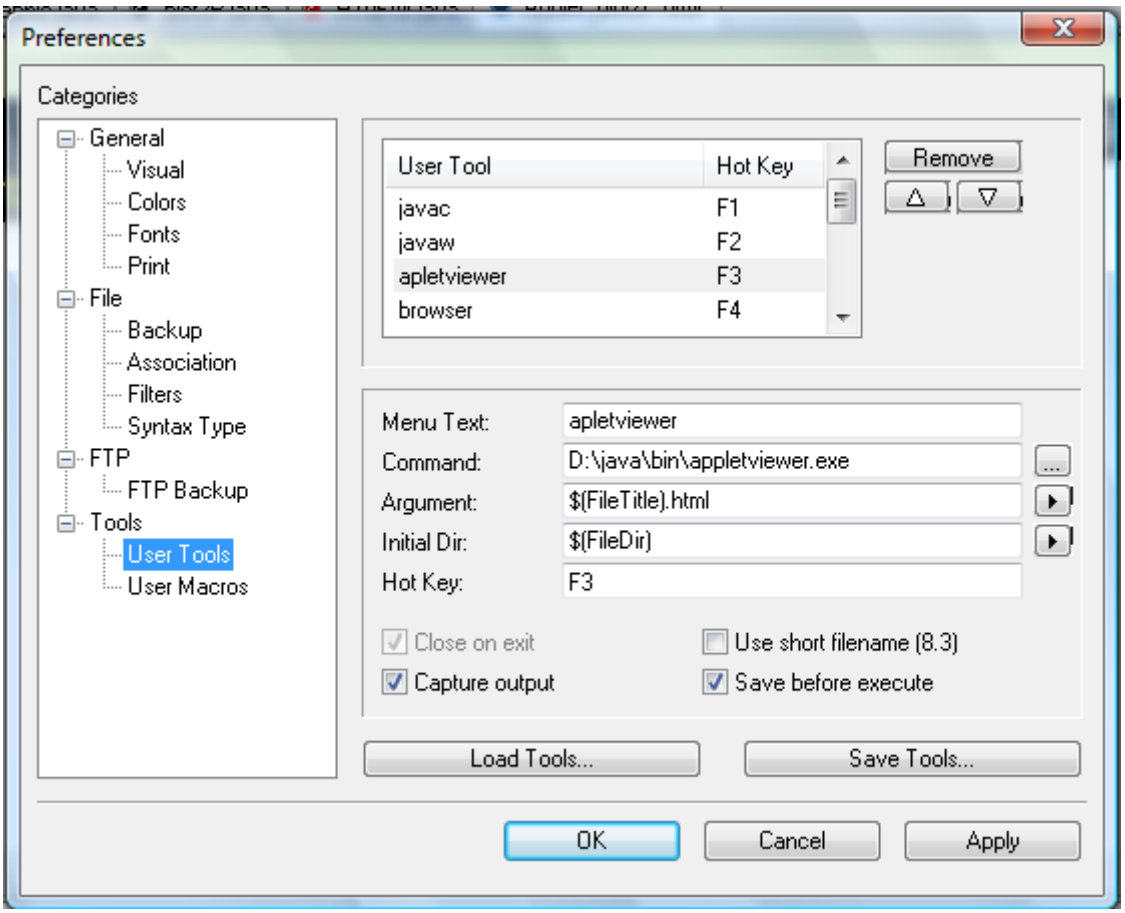
10.2.2 Applet_plot2P HTML program :

```
<html>
<applet code="Applet_plot2P.class" width=500 height=500>
</applet>
</html>
```

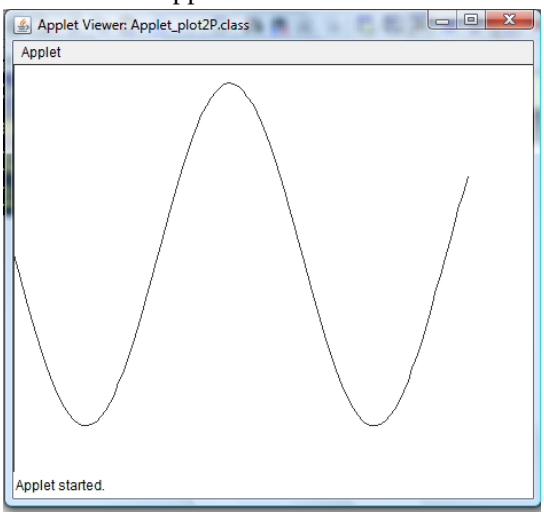
This code can be open in any browser program. Java also has a browser Simulator called appletviewer. Appletviewer ca be called from console environmet(like DOS of window system) as

- **appletviewer** Applet_plot2P.html

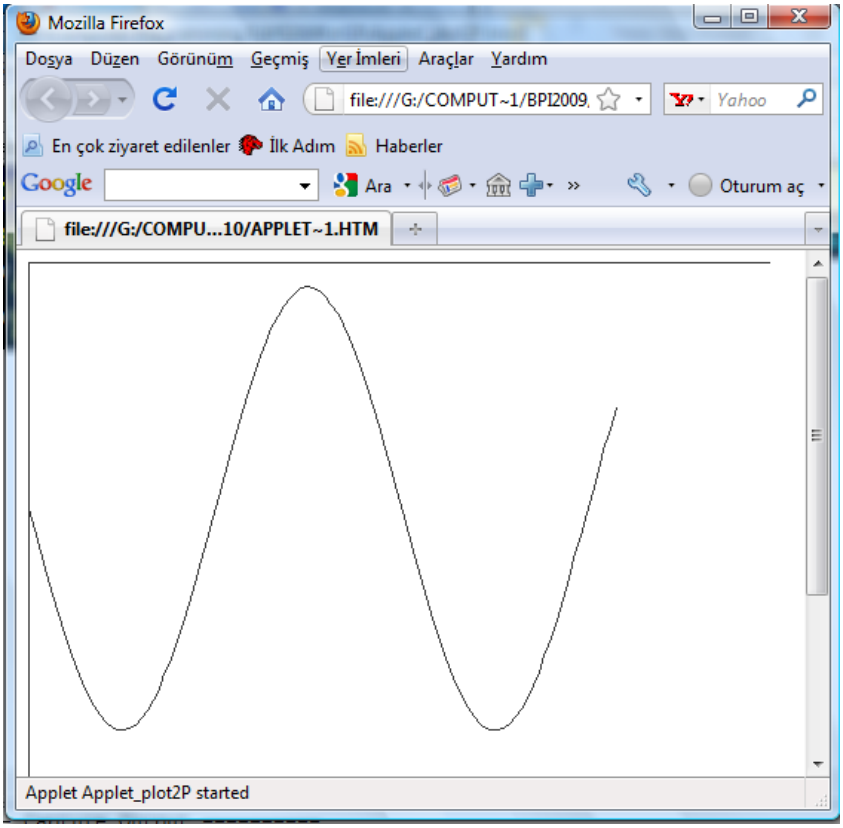
Or the same can be achived from programmable editor like crimson editor



The result in appletviewer:



The result in Mozzilla Firefox browser :



We can look at a second example:

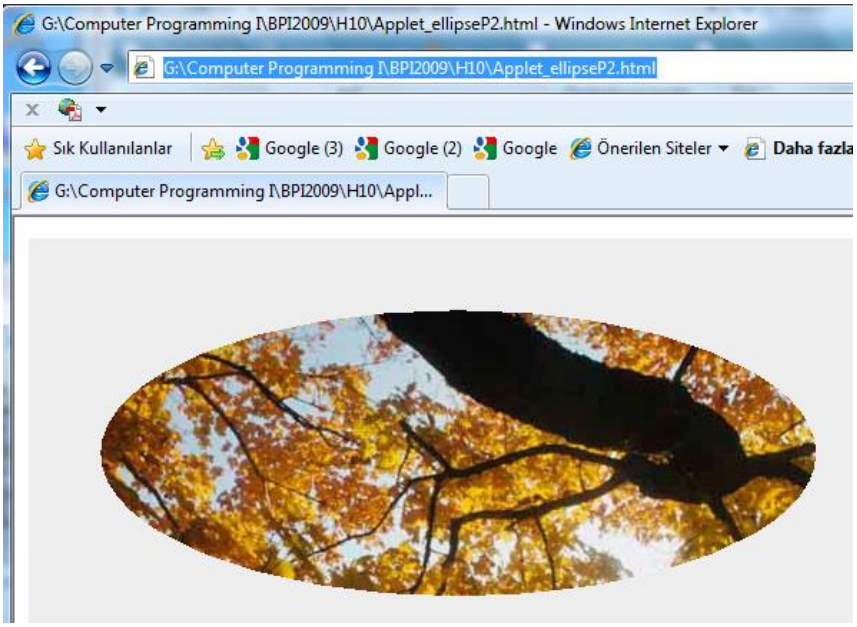
10.2.3 Applet_ellipseP2 JApplet graphic program to show JPanel plot2P

```
import java.awt.event.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class  Applet_ellipseP2 extends JApplet
{
 JPanel d;
 public void init()
 {
 d=new ellipseP2();
 add(d);
 }
}
```

10.2.4 Applet_ellipseP2 HTML program :

```
<html>
<applet code="Applet_ellipseP2.class" width=500 height=500>
</applet>
</html>
```

Output at internet Explorer browser:



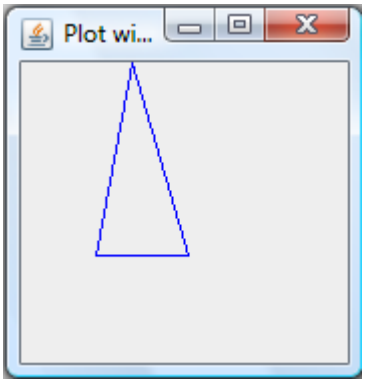
As a last thing, it should be note that an Applet program can be run from a main program through JFrame class.

10.2.5 a main program (java application program) to run class Applet_ellipseP2 program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  AppletGraphics extends JApplet
{
   public static void main(String s[])
   {
      JFrame f = new JFrame("Applet Graphics");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      JApplet applet = new Applet_ellipseP2();
      f.getContentPane().add("Center", applet);
      applet.init();
      f.pack();
      f.setSize(new Dimension(600,600));
      f.setVisible(true);
   }
}
```

WEEK10 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) study class starP and draw a triangle by using a similar programming . You give the traingle coordinates.

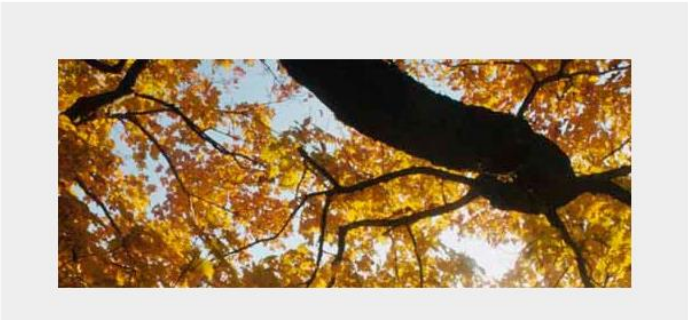


EX2) study class plot2P. Plot fonction y=x*x-2/x by using class plot2P between x=1 and x=5 . Use FrameGraphic.

WEEK 10 HOMEWORK EXERSIZES (Will bring to next class -Tuesday)

HW1) Study class plot2P. Plot fonction y=Math.sin(x)*(0.1*Math.random()) by using class plot2P between x=0 and x=π . Use Applet graphics

HW2) Write an applet to draw following graphics



HW3) Write a FrameGraphic program to draw a pentagon(five sided shape)

11 GRAPHIC CONTROL

In the last week we have seen how to draw the graphics. In this week we will investigate some of the graphic controllers. In graphic programming, programs first start and waits for user interface to enter some commands by using devices such as Mouse, keyboard etc. When a new command is given system carry out the given command and continue to wait fort the new command. To carry out graphic commands we have a group of methods called listeners. In the first group of graphic controllers buton(JButton), textfield(JTextField), textarea(JTextArea) and label(JLabel) will be investigated. The listener class fort his group of commands are called ActionListener.

11.1 JBUTTON CONTROL

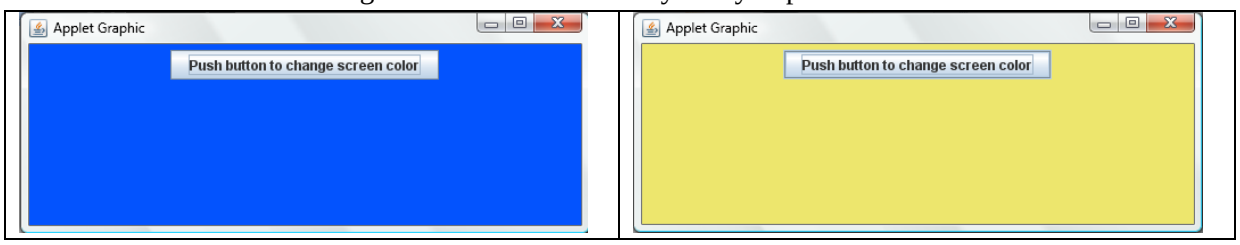
In the JButton examples , graphic windows of the previous chapter will be combined with JButton example. If the button in the graphic window is pushed by using the Mouse, actions given in the
void actionPerformed(ActionEvent e)
method will be carried out. At the end of **actionPerformed** method, the method **repaint()** should be called to be certain that graphic window is repainted in this new conditions. As the first example a button used to change screen color

11.1.1 JButton test program to change screen color randomly

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonA0 extends JApplet implements ActionListener
{
 JPanel inputPanel;
 JButton b;

 public void init()
 {
 inputPanel=new JPanel();
 b=new JButton("Push button to change screen color");
 b.addActionListener(this);
 inputPanel.add(b);
 add(inputPanel);
 }
 public void actionPerformed( ActionEvent e)
 {
 int red    =(int)(255.0*Math.random());
 int green  =(int)(255.0*Math.random());
 int blue   =(int)(255.0*Math.random());
 Color c=new Color(red,green, blue);
 inputPanel.setBackground(c);
 repaint();
 }

  public static void main(String s[])
  {
     JFrame f = new JFrame("Applet Graphic");
     f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     JApplet applet = new buttonA0();
     f.getContentPane().add("Center", applet);
     applet.init();
     f.setSize(800,500);
     f.setVisible(true);
  }
```

```
}
```

Color of the screen will change to a random color everytime you push the button



If the same program is written as **Lamda expressions (internal definition):**

11.1.1L JButton test program to change screen color randomly written as Lambda expression inner class form

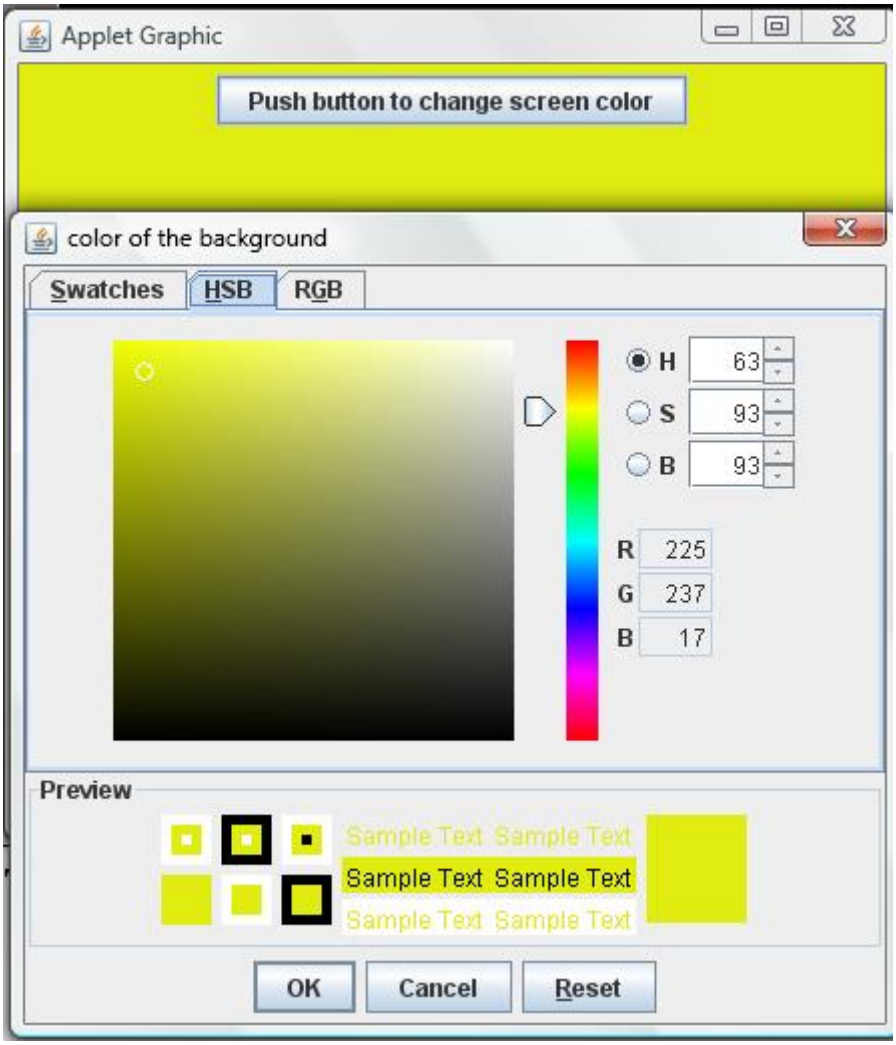
```
import java.awt.*;
import java.awt.event.*;

public class  buttonA0L extends JApplet
{
 JPanel inputPanel;
 JButton b;

 public void init()
 {
 inputPanel=new JPanel();
 b=new JButton("Push button to change screen color");
 b.addActionListener(e->{
 int red    =(int)(255.0*Math.random());
 int green  =(int)(255.0*Math.random());
 int blue   =(int)(255.0*Math.random());
 Color c=new Color(red,green, blue);
 inputPanel.setBackground(c);
 repaint();
 });
 inputPanel.add(b);
 add(inputPanel);
 }
  public static void main(String s[])
  {
     JFrame f = new JFrame("Applet Graphic");
     f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     JApplet applet = new buttonA0();
     f.getContentPane().add("Center", applet);
     applet.init();
     f.setSize(800,500);
     f.setVisible(true);
  }
}
```

As the second example the color change will be made through class JColorChooser instead of the random color

11.1.2 JButton test program to change screen color by using JColorChooser class

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
```

```
import java.awt.event.*;

public class  buttonA01 extends JApplet implements ActionListener
{
  JPanel inputPanel;
  JButton b;
  Color c;
  public void init()
  {
  c=Color.blue;
  inputPanel=new JPanel();
  b=new JButton("Push button to change screen color");
  b.addActionListener(this);
  inputPanel.add(b);
  inputPanel.setBackground(c);
  add(inputPanel);
  }
  public void actionPerformed( ActionEvent e)
  {
  c=JColorChooser.showDialog(null,"color of the background ",c );
  inputPanel.setBackground(c);
  repaint();
  }

    public static void main(String s[])
    {
        JFrame f = new JFrame("Applet Graphic");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new buttonA01();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```



The same program in Lambda expression form:

11.1.2L JButton test program to change screen color by using JColorChooser class Lambda expressions inner class form

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonA01L extends JApplet
{
  JPanel inputPanel;
  JButton b;
  Color c;
  public void init()
  {
  c=Color.blue;
  inputPanel=new JPanel();
  b=new JButton("Push button to change screen color");
  b.addActionListener(e->
  {
  c=JColorChooser.showDialog(null,"color of the background ",c );
  inputPanel.setBackground(c);
  repaint();
  });
  inputPanel.add(b);
```

```
  inputPanel.setBackground(c);
  add(inputPanel);
  }

   public static void main(String s[])
   {
      JFrame f = new JFrame("Applet Graphic");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      JApplet applet = new buttonA01();
      f.getContentPane().add("Center", applet);
      applet.init();
      f.setSize(800,500);
      f.setVisible(true);
   }
}
```

As the third example previously examined starP class is changed and new methods are added up to make it more usable together with JButton command and ActionListener class.

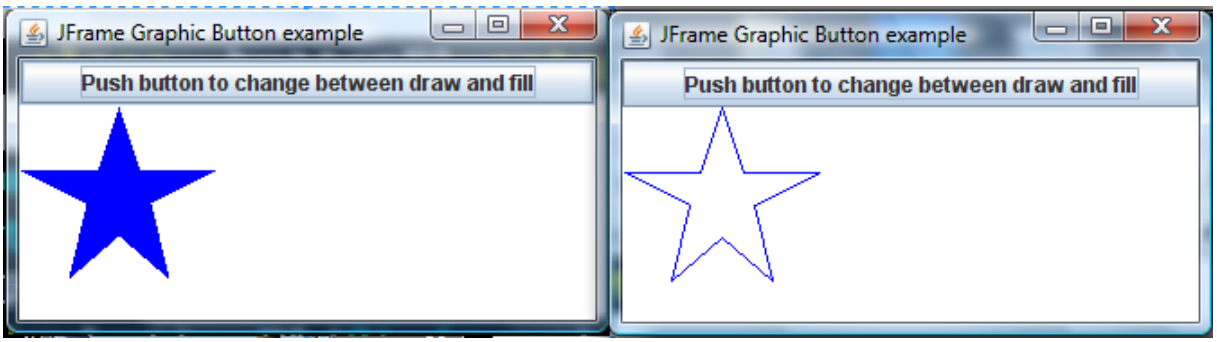
11.1.3 the new changed version of starP JPanel program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class starP extends JPanel
{
Color c1=Color.blue;//for shape
Color c2=Color.white;//for background
public boolean fill=false;

public void setFill() {fill=true;}
public void setDraw() {fill=false;}
public void setColor(Color c) {c1=c;}
public void setColor(int red,int green, int blue) {c1=new Color(red,green,blue);}
public void setBColor(Color c) {c2=c;}
public void setBColor(int red,int green, int blue) {c2=new Color(red,green,blue);}


public void paintComponent(Graphics g)
{
super.paintComponent(g);
Graphics2D g2=(Graphics2D)g;
GeneralPath star=new GeneralPath();
int x[]={55,67,109,73,83,55,27,37,1,43};
int y[]={0,36,36,54,96,72,96,54,36,36};
star.moveTo(x[0],y[0]);
for(int i=1;i<x.length;i++)
  {star.lineTo(x[i],y[i]);}
star.closePath();
g2.setColor(c1);
this.setBackground(c2);
if(fill) g2.fill(star);
else g2.draw(star);
  }
}
```

The first example program draws star but by controlling with buton, it used draw or fill option in the starP drawing method **public void paintComponent(Graphics g).** The actual control is carried out in the buton control within actionPerformed method.

11.1.4 buttonF1 JFrame graphic program program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF1 extends JFrame implements ActionListener
{
 JPanel inputPanel;
 JButton b;
 starP d;
 public buttonF1(String a)
 {    super(a);
      inputPanel=new JPanel();
      inputPanel.setLayout(new BorderLayout());
      d=new starP();
      d.setFill();
      b=new JButton("Push button to change between draw and fill");
      b.addActionListener(this);
      inputPanel.add(b,BorderLayout.NORTH);
      inputPanel.add(d,BorderLayout.CENTER);
      add(inputPanel);
 }

 public void actionPerformed( ActionEvent e)
 {
 if(d.fill) d.setDraw();
 else      d.setFill();
 repaint();
 }

  public static void main(String s[])
  {
      JFrame f = new buttonF1("JFrame Graphic Button example");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      f.setSize(800,500);
      f.setVisible(true);
  }
}
```



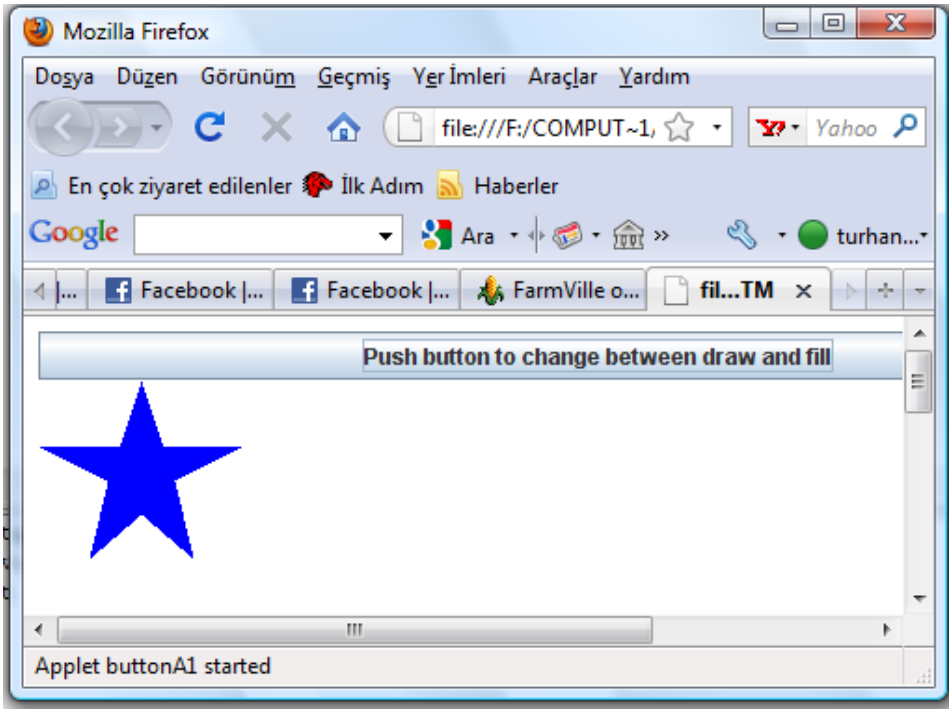
In the second example the same program is given as JApplet, instead of JFrame. Note that a main program to run this program from the console environment is also given as a part of the code. Html code is also listed. Output is shown in a browser program.

11.1.5 buttonA1 JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonA1 extends JApplet implements ActionListener
{
  JPanel inputPanel;
  JButton b;
  starP d;

  public void init()
  {
  inputPanel=new JPanel();
  inputPanel.setLayout(new BorderLayout());
  d=new starP();
  d.setFill();
  b=new JButton("Push button to change between draw and fill");
  b.addActionListener(this);
  inputPanel.add(b,BorderLayout.NORTH);
  inputPanel.add(d,BorderLayout.CENTER);
  add(inputPanel);
  }
  public void actionPerformed( ActionEvent e)
  {
  if(d.fill) d.setDraw();
  else       d.setFill();
  repaint();
  }

    public static void main(String s[])
    {
        JFrame f = new JFrame("Applet Graphic");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new buttonA1();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

11.1.6 buttonA1.html html program

```
<html>
<applet code="buttonA1.class" width=600 height=500>
</applet>
</html>
```

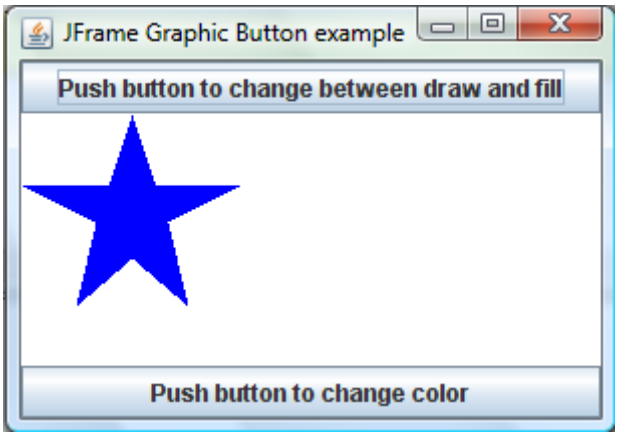


In the next example, a second buton is added up to program to paint the drawn shape with a random number. When more than one control process is existed, different processes is seperated by using and if procedure such as **if(e.getSource()==b1) in public void actionPerformed(ActionEvent e)** method.

11.1.7 buttonF2 JFrame graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF2 extends JFrame implements ActionListener
{
 JPanel inputPanel;
 JButton b1,b2;
 starP d;
 int red,green,blue;
 Color c;

 public buttonF2(String a)
 {   super(a);
     inputPanel=new JPanel();
     inputPanel.setLayout(new BorderLayout());
     d=new starP();
     d.setFill();
     b1=new JButton("Push button to change between draw and fill");
     b1.addActionListener(this);
     b2=new JButton("Push button to change color");
     b2.addActionListener(this);
     inputPanel.add(b1,BorderLayout.NORTH);
     inputPanel.add(b2,BorderLayout.SOUTH);
     inputPanel.add(d,BorderLayout.CENTER);
     add(inputPanel);
```

```
  }

  public void actionPerformed( ActionEvent e)
  {
  if(e.getSource()==b1)
  {
  if(d.fill) d.setDraw();
  else      d.setFill();
  }
  else if(e.getSource()==b2)
  {
  red    =(int)(255.0*Math.random());
  green  =(int)(255.0*Math.random());
  blue   =(int)(255.0*Math.random());
  d.setColor(red,green,blue);
  }
  repaint();
  }

   public static void main(String s[])
   {
       JFrame f = new buttonF2("JFrame Graphic Button example");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       f.setSize(800,500);
       f.setVisible(true);
   }
}
```



If Lambda expression inner class version of the same program is written:

11.1.7L buttonF2 JFrame graphic program , lambda version

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF2L extends JFrame
{
  JPanel inputPanel;
  JButton b1,b2;
  starP d;
  int red,green,blue;
  Color c;
```

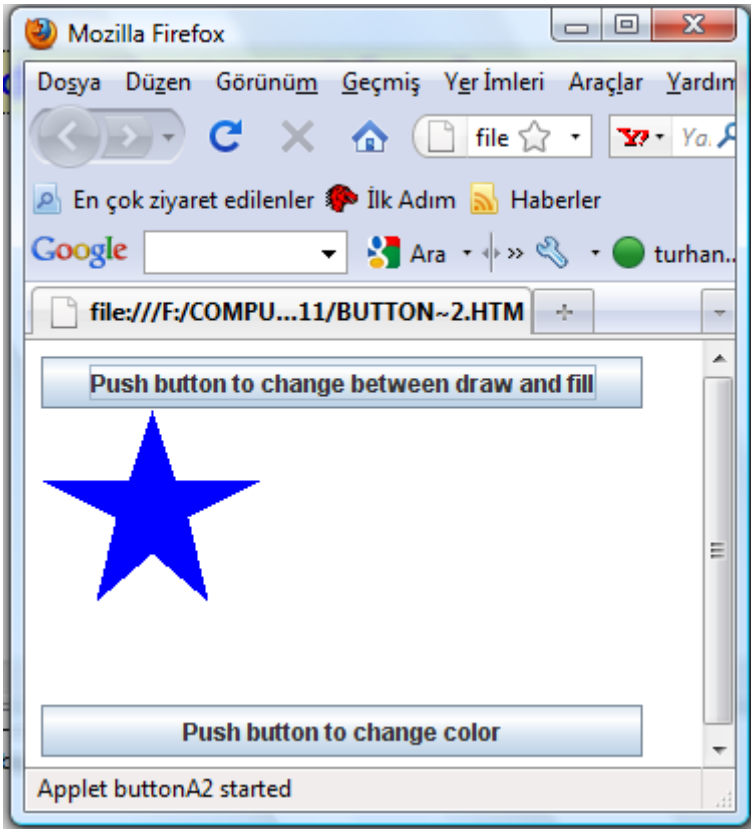
```
  public buttonF2L(String a)
  {    super(a);
       inputPanel=new JPanel();
       inputPanel.setLayout(new BorderLayout());
       d=new starP();
       d.setFill();
       b1=new JButton("Push button to change between draw and fill");
       b1.addActionListener(e1->{ if(d.fill) d.setDraw();
                                  else      d.setFill();
                                  repaint();
                                 });
       b2=new JButton("Push button to change color");
       b2.addActionListener(e2->  {
       red    =(int)(255.0*Math.random());
       green  =(int)(255.0*Math.random());
       blue   =(int)(255.0*Math.random());
       d.setColor(red,green,blue);
       repaint();
                                  });
       inputPanel.add(b1,BorderLayout.NORTH);
       inputPanel.add(b2,BorderLayout.SOUTH);
       inputPanel.add(d,BorderLayout.CENTER);
       add(inputPanel);
  }

   public static void main(String s[])
   {
        JFrame f = new buttonF2L("JFrame Graphic Button example");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(800,500);
        f.setVisible(true);
   }
}
```

Same program is rewritten as JApplet :

11.1.8 buttonA2 JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonA2 extends JApplet implements ActionListener
{
  JPanel inputPanel;
  JButton b1,b2;
  starP d;
  int red,green,blue;
  Color c;

    public void init()
    {
       inputPanel=new JPanel();
       inputPanel.setLayout(new BorderLayout());
       d=new starP();
       d.setFill();
       b1=new JButton("Push button to change between draw and fill");
```

```
      b1.addActionListener(this);
      b2=new JButton("Push button to change color");
      b2.addActionListener(this);
      inputPanel.add(b1,BorderLayout.NORTH);
      inputPanel.add(b2,BorderLayout.SOUTH);
      inputPanel.add(d,BorderLayout.CENTER);
      add(inputPanel);
    }

  public void actionPerformed( ActionEvent e)
  {
  if(e.getSource()==b1)
  {
  if(d.fill) d.setDraw();
  else       d.setFill();
  }
  else if(e.getSource()==b2)
  {
  red    =(int)(255.0*Math.random());
  green  =(int)(255.0*Math.random());
  blue   =(int)(255.0*Math.random());
  d.setColor(red,green,blue);
  }
  repaint();
  }

    public static void main(String s[])
    {
       JFrame f = new JFrame("Applet Graphic");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       JApplet applet = new buttonA2();
       f.getContentPane().add("Center", applet);
       applet.init();
       f.setSize(800,500);
       f.setVisible(true);
    }
}
```

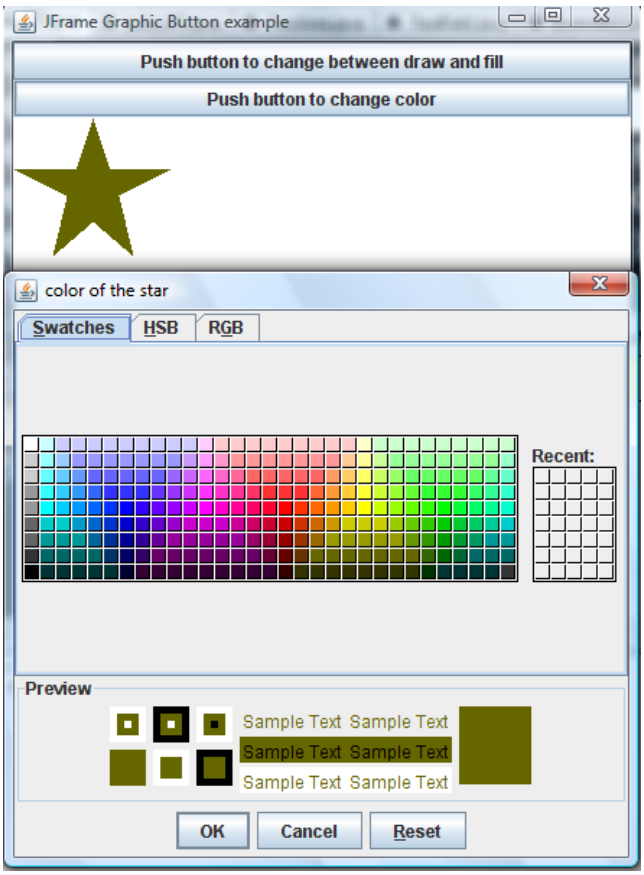


Previous program is changed a little bit. In this new version, instead of random color selection color selected through a library color change graphic window JColorChooser class **JColorChooser.showDialog** method, This method returns a variable in class Color according to the color chosen in the graphic window.

11.1.9 buttonF3 JFrame graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF3 extends JFrame implements ActionListener
{
  JPanel inputPanel;
  JButton b1,b2;
  starP d;
  Color c;

  public buttonF3(String a)
  {    super(a);
       c=Color.blue;
       inputPanel=new JPanel();
       inputPanel.setLayout(new BorderLayout());
       d=new starP();
       d.setFill();
       b1=new JButton("Push button to change between draw and fill");
       b1.addActionListener(this);
       b2=new JButton("Push button to change color");
       b2.addActionListener(this);
       inputPanel.add(b1,BorderLayout.NORTH);
```

```
        inputPanel.add(b2,BorderLayout.CENTER);
        add(inputPanel,BorderLayout.NORTH);
        add(d,BorderLayout.CENTER);
    }

    public void actionPerformed( ActionEvent e)
    {
    if(e.getSource()==b1)
    {
    if(d.fill) d.setDraw();
    else       d.setFill();
    }
    else if(e.getSource()==b2)
    {
    c=JColorChooser.showDialog(null,"color of the star ",c );
    d.setColor(c);
    }
    repaint();
    }

     public static void main(String s[])
     {
        JFrame f = new buttonF3("JFrame Graphic Button example");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(800,300);
        f.setVisible(true);
     }
}
```

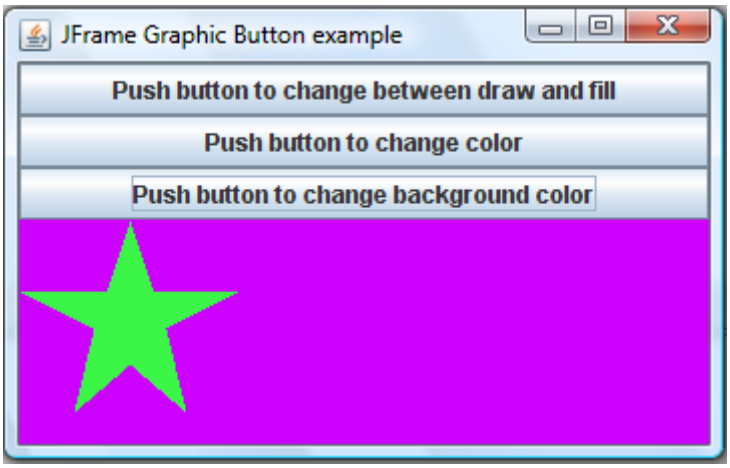


In the next program three control butons are used. The first one controls drawing or filling mode. The second controls the color of the shape by random color chosing. And the third buton control the background color of the shape through **JColorChooser.showDialog** method

11.1.10 buttonF4 JFrame graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF4 extends JFrame implements ActionListener
{
  JPanel inputPanel;
  JButton b1,b2,b3;
  starP d;
  int red,green,blue;
  Color c1,c2;

  public buttonF4(String a)
  {    super(a);
       inputPanel=new JPanel();
       inputPanel.setLayout(new BorderLayout());
       d=new starP();
       d.setFill();
       b1=new JButton("Push button to change between draw and fill");
       b1.addActionListener(this);
       b2=new JButton("Push button to change color");
       b2.addActionListener(this);
       b3=new JButton("Push button to change background color");
       b3.addActionListener(this);
       inputPanel.add(b1,BorderLayout.NORTH);
       inputPanel.add(b2,BorderLayout.CENTER);
       inputPanel.add(b3,BorderLayout.SOUTH);
       add(inputPanel,BorderLayout.NORTH);
       add(d,BorderLayout.CENTER);
  }

  public void actionPerformed( ActionEvent e)
  {
  if(e.getSource()==b1)
  {//choose draw or select mode
  if(d.fill) d.setDraw();
  else       d.setFill();
  }
  else if(e.getSource()==b2)
  {//Random shape color
  red    =(int)(255.0*Math.random());
  green  =(int)(255.0*Math.random());
  blue   =(int)(255.0*Math.random());
  d.setColor(red,green,blue);
  }
  else if(e.getSource()==b3)
  {//selected background color
  c2=JColorChooser.showDialog(null,"color of the star ",c2 );
  d.setBColor(c2);
  }
  repaint();
  }

   public static void main(String s[])
   {
       JFrame f = new buttonF4("JFrame Graphic Button example");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       f.setSize(800,500);
```

```
        f.setVisible(true);
    }
}
```

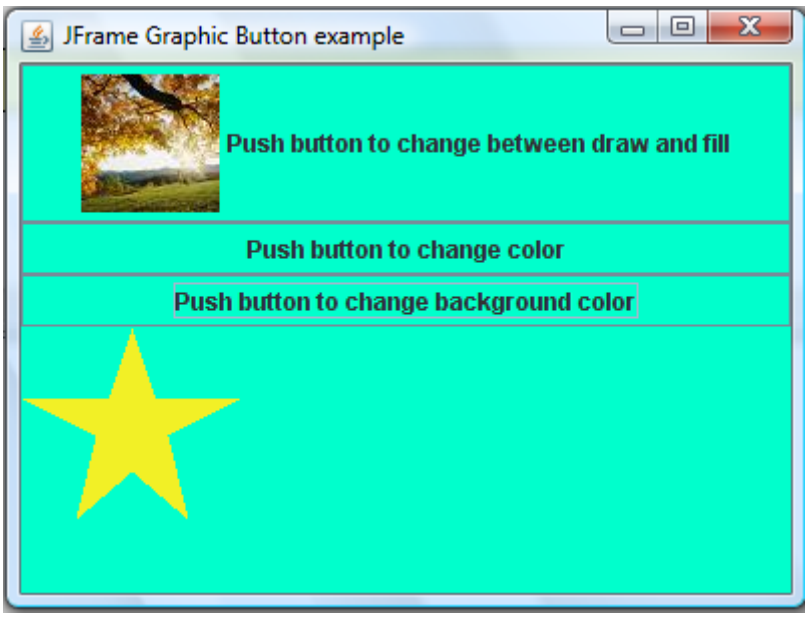


In the last example of the series buton is drawn in graphic window with a Picture on the button face. This program also paints the whole when background color change buton pushed.

11.1.11 buttonF5 JFrame graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonF5 extends JFrame implements ActionListener
{
  JPanel inputPanel;
  JButton b1,b2,b3;
  starP d;
  int red,green,blue;
  Color c1,c2;
  Icon shape;

  public buttonF5(String a)
  {    super(a);
       inputPanel=new JPanel();
       inputPanel.setLayout(new BorderLayout());
       d=new starP();
       d.setFill();
       shape=new ImageIcon("fall.jpg");
       b1=new JButton("Push button to change between draw and fill",shape);
       b1.addActionListener(this);
       b2=new JButton("Push button to change color");
       b2.addActionListener(this);
       b3=new JButton("Push button to change background color");
       b3.addActionListener(this);
       inputPanel.add(b1,BorderLayout.NORTH);
       inputPanel.add(b2,BorderLayout.CENTER);
       inputPanel.add(b3,BorderLayout.SOUTH);
       add(inputPanel,BorderLayout.NORTH);
       add(d,BorderLayout.CENTER);
  }

  public void actionPerformed( ActionEvent e)
  {
  if(e.getSource()==b1)
  {//choose draw or select mode
```

```
  if(d.fill) d.setDraw();
  else       d.setFill();
  }
  else if(e.getSource()==b2)
  {//Random shape color
  red    =(int)(255.0*Math.random());
  green  =(int)(255.0*Math.random());
  blue   =(int)(255.0*Math.random());
  d.setColor(red,green,blue);
  }
  else if(e.getSource()==b3)
  {//selected background color
  c2=JColorChooser.showDialog(null,"color of the star ",c2 );
  d.setBColor(c2);
  b1.setBackground(c2);
  b2.setBackground(c2);
  b3.setBackground(c2);
  }
  repaint();
  }
   public static void main(String s[])
   {
       JFrame f = new buttonF5("JFrame Graphic Button example");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       f.setSize(400,300);
       f.setVisible(true);
   }
}
```



We should note that all the graphic programming can be prepare as a JPanel, and inserted into a ready to use template program similar to in the previous chapter. For example let us re-write buton example buttonF1.java as a panel buttonP.java

11.1.12 buttonP JPanel graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  buttonP extends JPanel implements ActionListener
```

```
{
 JButton b;
 starP d;
 public buttonP()
 {
     setLayout(new BorderLayout());
     d=new starP();
     d.setFill();
     b=new JButton("Push button to change between draw and fill");
     b.addActionListener(this);
     add(b,BorderLayout.NORTH);
     add(d,BorderLayout.CENTER);
 }

 public void actionPerformed( ActionEvent e)
 {
 if(d.fill) d.setDraw();
 else      d.setFill();
 repaint();
 }

}
```

Now, this Panel can be inserted into a Standard JFrame graphic template as we used in the previous chapter

11.1.13 FrameGraphic JPanel graphic program

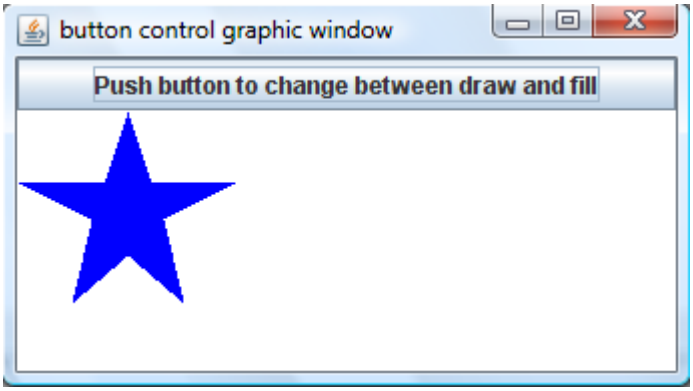
```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
public class FrameGraphic extends JFrame
{
                JPanel d;

  public FrameGraphic(String a,JPanel di)
  {    super(a);
       d=di;
       add(d);
  }

  public static void plot(String a,JPanel di)
  {
       FrameGraphic f = new FrameGraphic(a,di);
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       f.setSize(800,500);
       f.setVisible(true);
  }

}
```

And by using a small test program we can run the program

11.1.14 buttonP test program

```
class H11Ex1 {
public static void main(String args[])
{
FrameGraphic.plot("button control graphic window",new buttonP());
}
}
```

The result will be the same as the previous example



11.2 JLABEL, JTEXTFIELD AND JTEXTAREA GRAPHIC CONTROL CLASSES

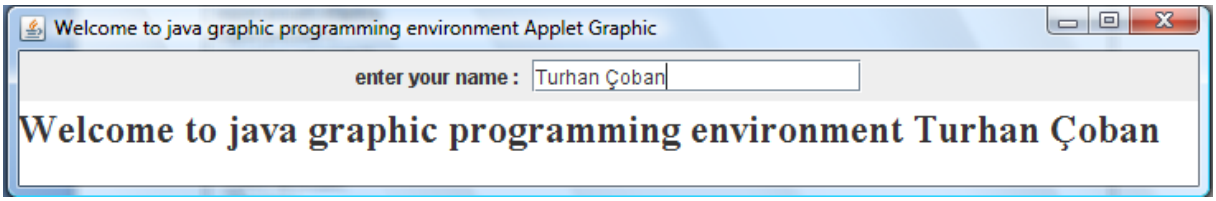
Three control groups will be introduced here: JLabel, JTextField and JTextArea. This classes used to write messages. JLabel is for only writing. In JTextField and JTextArea we can write, read and change the writings. Method .**getText()** read the text, and method **setText(String s)** change the text. JTextField and JTextArea is similar classes, but usually JTextField used for on eline inputs, while TextArea is used for editing/changing of a big text.

In our first example we will use this three classes together a name wil be read from the screen by using JTextField and a message will be written into JTextArea field. Usage of the JTextField boz is defined with a JLabel message.

11.2.1 welcomeA JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  welcomeA extends JApplet implements ActionListener
{
  JLabel label1;
  JTextField namebox;
  JPanel inputPanel;
  JTextArea t;
  String name;


  public void init()
  {
            label1=new JLabel("enter your name : ");
            t=new JTextArea("                                    ");
            t.setFont(new Font("Serif",Font.BOLD,24));
            namebox=new JTextField("                                ");
    setLayout(new BorderLayout());
    inputPanel=new JPanel();
    inputPanel.add(label1);
    inputPanel.add(namebox);
    add(inputPanel,BorderLayout.NORTH);
    add(t,BorderLayout.CENTER);
    namebox.addActionListener(this);
  }


public void actionPerformed( ActionEvent e)
```

```
{
            if(e.getSource()==namebox)
      {   name=namebox.getText();
                  t.setText("Welcome to java graphic programming environment "+name); }
}

   public static void main(String s[])
   {
      JFrame f = new JFrame("Welcome to java graphic programming environment Applet Graphic");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      JApplet applet = new welcomeA();
      f.getContentPane().add("Center", applet);
      applet.init();
      f.setSize(800,500);
      f.setVisible(true);
   }
}
```

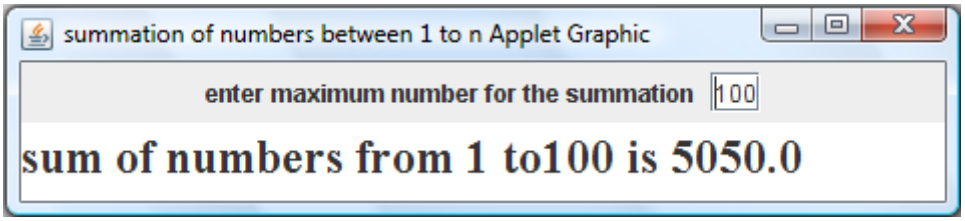


In the second example, numbers are added up to a given number which is defined in a JTextField area A seperate method is given for summation of the numbers.

11.2.2 sumA JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  sumA extends JApplet implements ActionListener
{
 JLabel label1;
 JTextField namebox;
 JPanel inputPanel;
 JTextArea t;
 int n;


 public void init()
 {
            label1=new JLabel("enter maximum number for the summation ");
            t=new JTextArea(" ");
            t.setFont(new Font("Serif",Font.BOLD,24));
            namebox=new JTextField("100");
  setLayout(new BorderLayout());
  inputPanel=new JPanel();
  inputPanel.add(label1);
  inputPanel.add(namebox);
  add(inputPanel,BorderLayout.NORTH);
  add(t,BorderLayout.CENTER);
  namebox.addActionListener(this);
 }

 public double sum(int n)
 {double tt=0;
  for(int i=0;i<=n;i++) {tt+=i;}
```

```
  return tt;
 }

public void actionPerformed( ActionEvent e)
{
            if(e.getSource()==namebox)
      {   n=Integer.parseInt(namebox.getText());
                t.setText("sum of numbers from 1 to"+n+" is "+sum(n)); }
}

   public static void main(String s[])
   {
      JFrame f = new JFrame("summation of numbers between 1 to n Applet Graphic");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      JApplet applet = new sumA();
      f.getContentPane().add("Center", applet);
      applet.init();
      f.setSize(800,500);
      f.setVisible(true);
   }
}
```

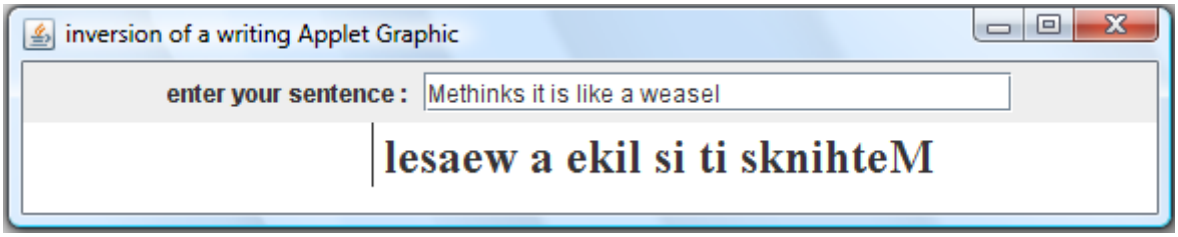


In the next example string of characters are read and inverted. In the inversion process a new class of StringBuffer is used. In order to add string into the StringBuffer class, append(char c) method is used. Each character is read from the String, by using **charAt(int i)** method. Read character string is converted form StringBuffer back to the strings by toString() method. When ctrl-return is pushed in the JTextField area Action is detected through ActionPerformed method. In order to tell computer to listen JTextField area, command **sentencebox.addActionListener(this)** command is used.

11.2.3 inverseA JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  inverseA extends JApplet implements ActionListener
{
 JLabel label1;
 JTextField sentencebox;
 JPanel inputPanel;
 JTextArea t;
 String sentence;


 public void init()
 {
            label1=new JLabel("enter your sentence : ");
            t=new JTextArea(" ");
            t.setFont(new Font("Serif",Font.BOLD,24));
            sentencebox=new JTextField("Methinks it is like a weasel                                    ");
  setLayout(new BorderLayout());
  inputPanel=new JPanel();
  inputPanel.add(label1);
```

```
    inputPanel.add(sentencebox);
    add(inputPanel,BorderLayout.NORTH);
    add(t,BorderLayout.CENTER);
    sentencebox.addActionListener(this);
  }

  public static String invert(String source)
  { int i, len = source.length();
    StringBuffer dest = new StringBuffer(len);
    for (i = (len - 1); i >= 0; i--) {
    dest.append(source.charAt(i)); }
    return dest.toString();
  }

public void actionPerformed( ActionEvent e)
{
                if(e.getSource()==sentencebox)
        {   sentence=sentencebox.getText();
                        t.setText(invert(sentence)); }
}

    public static void main(String s[])
    {
        JFrame f = new JFrame("inversion of a writing Applet Graphic");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new inverseA();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

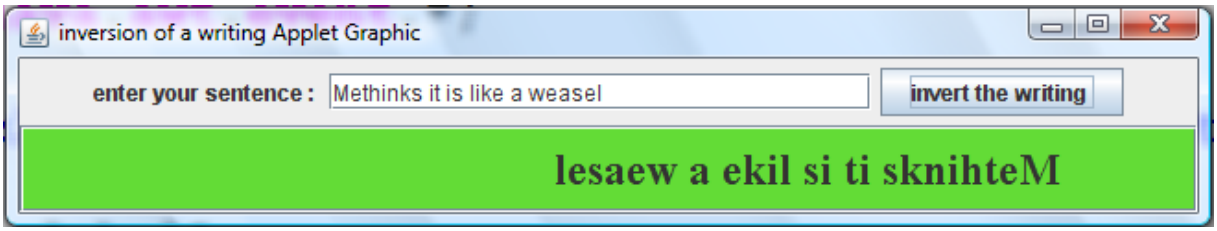


In the next example , two seperate controller, A JButton and a JTextField is used fort he same control. JTextField write the message into another JTextField area, then when JButton is pushed message is inverted and message area is painted by a random color.

11.2.4 inverseA1 JApplet graphic program

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class  inverseA1 extends JApplet implements ActionListener
{
  JLabel label1;
  JTextField sentencebox;
  JPanel inputPanel;
  JTextField t;
  String sentence;
  JButton b;
  String s;
  int red,green,blue;
```

```
 public void init()
 {
            s="Methinks it is like a weasel                                    ";
            label1=new JLabel("enter your sentence : ");
            t=new JTextField(s);
            t.setFont(new Font("Serif",Font.BOLD,24));
            sentencebox=new JTextField(s);
            b=new JButton("invert the writing " );
  setLayout(new BorderLayout());
  inputPanel=new JPanel();
  inputPanel.add(label1);
  inputPanel.add(sentencebox);
  inputPanel.add(b);
  add(inputPanel,BorderLayout.NORTH);
  add(t,BorderLayout.CENTER);
  sentencebox.addActionListener(this);
  b.addActionListener(this);
 }

 public static String invert(String source)
 { int i, len = source.length();
  StringBuffer dest = new StringBuffer(len);
  for (i = (len - 1); i >= 0; i--) {
  dest.append(source.charAt(i)); }
  return dest.toString();
 }

public void actionPerformed( ActionEvent e)
{
      if(e.getSource()==sentencebox)
      {   s=sentencebox.getText();
                t.setText(s);
              }
            else if(e.getSource()==b)
              {  t.setText(invert(t.getText()));
                red    =(int)(255.0*Math.random());
        green  =(int)(255.0*Math.random());
        blue   =(int)(255.0*Math.random());
        Color c1=new Color(red,green,blue);
        t.setBackground(c1);
              }
}

  public static void main(String s[])
  {
     JFrame f = new JFrame("inversion of a writing Applet Graphic");
     f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     JApplet applet = new inverseA1();
     f.getContentPane().add("Center", applet);
     applet.init();
     f.setSize(800,500);
     f.setVisible(true);
  }
}
```



11.2.4 inverseA1L JApplet graphic program with Lambda expressions

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
public class  inverseA1L extends JApplet
{
  JLabel label1;
  JTextField sentencebox;
  JPanel inputPanel;
  JTextField t;
  String sentence;
  JButton b;
  String s;
  int red,green,blue;

  public void init()
  {
          s="Methinks it is like a weasel                                        ";
          label1=new JLabel("enter your sentence : ");
          t=new JTextField(s);
          t.setFont(new Font("Serif",Font.BOLD,24));
          sentencebox=new JTextField(s);
          b=new JButton("invert the writing " );
   setLayout(new BorderLayout());
   inputPanel=new JPanel();
   inputPanel.add(label1);
   inputPanel.add(sentencebox);
   inputPanel.add(b);
   add(inputPanel,BorderLayout.NORTH);
   add(t,BorderLayout.CENTER);
   sentencebox.addActionListener(e1->{s=sentencebox.getText();
                                      t.setText(s);
                                     });
   b.addActionListener(e2->{
              t.setText(invert(t.getText()));
              red    =(int)(255.0*Math.random());
        green  =(int)(255.0*Math.random());
        blue   =(int)(255.0*Math.random());
        Color c1=new Color(red,green,blue);
        t.setBackground(c1);
                          } );
  }

  public static String invert(String source)
  { int i, len = source.length();
   StringBuffer dest = new StringBuffer(len);
   for (i = (len - 1); i >= 0; i--) {
   dest.append(source.charAt(i)); }
```

```
    return dest.toString();
  }

    public static void main(String s[])
    {
        JFrame f = new JFrame("inversion of a writing Applet Graphic");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new inverseA1L();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

CLASS EXERSIZES

EX1) investigate program **sumA.java** write a **JFrame** code to sum square of numbers from 0 to n

EX2) investigate program **inverseA1.java** Develop a **JFrame** code to change character size in each time the button is pushed

EX3) investigate **buttonF1.java** code. Write a **squareP.java** to draw a square instead of a star and write a graphic program in JFrame to use it. Color of the square should be changed in each time the button is pushed. In order to change the color, you can use either randomly generated color or you can use JColorChooser class.

EX4) Write a graphic program either in JApplet to enter two numbers from the graphic page and sum of the number will be written as a result.

EX5) Numeric.java was given at week 9. Enter a real number at the graphic screen, and then calculate the derivative of function f(x)=x*x-5, write the result in a JTextArea field.

```
abstract class f_x
{ abstract double func(double x);}

public class numerical
{

public static double derivative(f_x f,double x)
{  double h=0.00001;
  return (-f.func(x+2.0*h)+8.0*f.func(x+h)-8.0*f.func(x-h)+f.func(x-2.0*h))/(12.0*h);
}

public static double integral(f_x f_xnt,double a,double b)
{
//integral f(x)dx
 double r[]={-0.973906528517171,-0.865063366688984,-0.679409568299024,-0.433395394129247,-0.148874338981631,
 0.148874338981631,0.433395394129247,0.679409568299024,0.865063366688984,0.973906528517171};
 double c[]={0.066671344308684,0.149451349150580,0.219086362515982,0.269266719309996,0.295524224714752,
 0.295524224714752,0.269266719309996,0.219086362515982,0.149451349150580,0.066671344308684};
double z=0,x,y;
double k1=(b-a)/2.0;
double k2=(b+a)/2.0;
for(int i=0;i<r.length;i++)
{
```

```
x=k2+k1*r[i];
y=f_xnt.func(x);
z+=k1*c[i]*y;
}
return z;
}
}
```

```
class fb extends f_x

{ public double func(double x)

 { return x*x-5.0;}}
```

EX6) Investigate **buttonF4.java** , write a similar **JApplet** program that will be draw an equal triangle instead of star

12 MORE GRAPHIC CONTROLS

12.1 MOUSE CONTROL

Mouse is one of the most used graphic controller in graphic programming environment. Complex controls are possible by using Mouse like pushing, pushing and pulling and releasing of one of the Mouse buttons. We have two listener class for Mouse MouseListener and MouseMotionListener. **MouseListener** class has the following control methods :

public void mouseClicked(MouseEvent e)
public void mousePressed(MouseEvent e)
public void mouseReleased(MouseEvent e)
public void mouseEntered(MouseEvent e)
public void mouseExited(MouseEvent e)

MouseMotionListener class has the following control methods

public void mouseDragged(MouseEvent e)
public void mouseMoved(MouseEvent e)

event class for the Mouse control is **MouseEvent** class. Mouse event class **getX()** and **getY()** methods to get mouse coordinates.

Our first example will be a program to draw a line on the screen by moving Mouse pres, Mouse dragg and Mouse release as a serial movement. In order to establish this MouseP JPanel program is developed.

12.1.1 Mouse control JPanel class mouseP

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
```

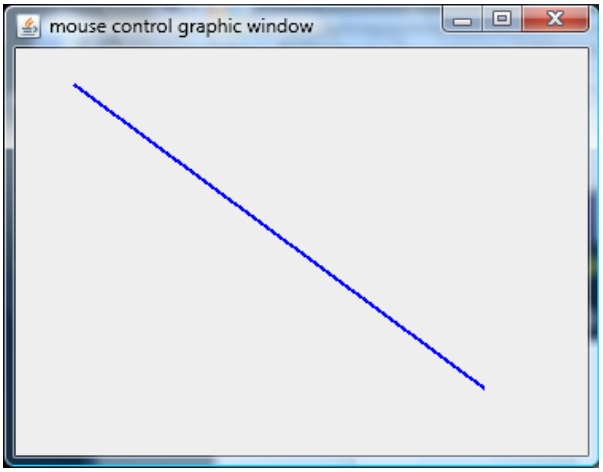
```
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  mouseP extends JPanel implements MouseListener,MouseMotionListener
{
  //push-move-release line draw
  int x1,y1,x2,y2;
  Color c;
  boolean first;
  Line2D x;

 public mouseP()
 {super();
  x1=0;y1=0;x2=0;y2=0;
  first=true;
  addMouseListener(this);
  addMouseMotionListener(this);
 }

 public mouseP(int x1i,int y1i,int x2i,int y2i)
 {super();
  x1=x1i;y1=y1i;x2=x2i;y2=y2i;
  first=true;
 addMouseListener(this);
 addMouseMotionListener(this);
 }

 public void setLine(int x1i,int y1i,int x2i,int y2i)
 {x1=x1i;y1=y1i;x2=x2i;y2=y2i;}


 public void paint(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.blue);
 g2.setStroke(new BasicStroke(2.0f));
 x=new Line2D.Double(x1,y1,x2,y2);
 g2.draw(x);
 }

//MouseListener
public void mouseClicked(MouseEvent e)
 {}

 public void mousePressed(MouseEvent e)
 {
   x1=e.getX();y1=e.getY();}

  public void mouseReleased(MouseEvent e)
 { x2=e.getX();y2=e.getY();repaint();}

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener
```

```
public void mouseDragged(MouseEvent e)
{}
public void mouseMoved(MouseEvent e)
{}

}
```

This program will be used with the previously developed FrameGraphis class. The test program will be as follows:

12.1.2 Test program for Mouse control JPanel class mouseP

```
class mouseF {
public static void main(String args[])
{FrameGraphic.plot("mouse control graphic window",new mouseP());}
}
```



Second program n points instead of lines.

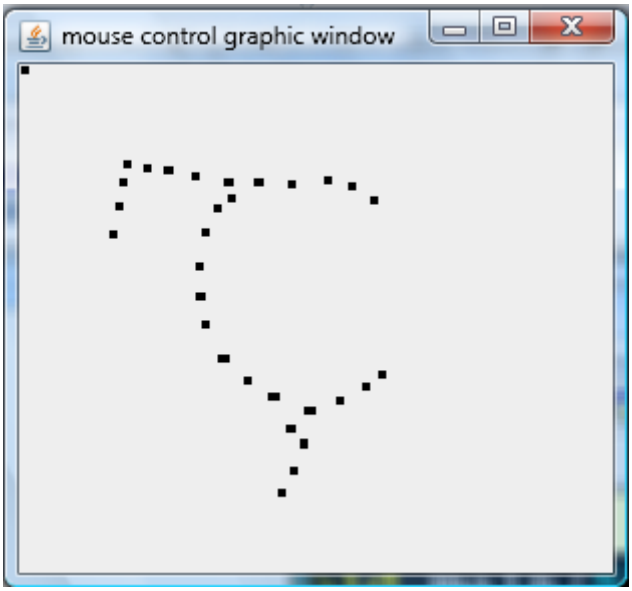
12.1.3 Mouse control JPanel class mouseP1

```
İmport java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  mouseP1 extends JPanel implements MouseListener,MouseMotionListener
{
 //push to draw a point
            int x1,y1;
 Color c;

 public mouseP1()
 {super();
```

```
 x1=0;y1=0;
 addMouseListener(this);
 addMouseMotionListener(this);
}

public mouseP1(int x1i,int y1i,int x2i,int y2i)
{super();
 x1=x1i;y1=y1i;
addMouseListener(this);
addMouseMotionListener(this);
}

public void setLine(int x1i,int y1i,int x2i,int y2i)
{x1=x1i;y1=y1i;}


public void paint(Graphics g)
   {   Graphics2D g2 = (Graphics2D)g;
       g2.setPaint(Color.black);
       Ellipse2D elips1=new Ellipse2D.Double(x1,y1,5,5);
       g2.fill(elips1);
   }

//MouseListener (mouse dinleyicisi)
public void mouseClicked(MouseEvent e)
 { x1=e.getX();y1=e.getY();repaint(); }

public void mousePressed(MouseEvent e)
 { x1=e.getX();y1=e.getY();repaint();
 }

public void mouseReleased(MouseEvent e)
 {x1=e.getX();y1=e.getY();repaint();}

public void mouseEntered(MouseEvent e)
 { }

public void mouseExited(MouseEvent e)
 { }

//MouseMotionListener (mouse hareket dinleyicisi) metotlari

public void mouseDragged(MouseEvent e)
 {}

public void mouseMoved(MouseEvent e)
 {}

}
```

12.1.4 Test program for Mouse control JPanel class mouseP1

```
class mouseF1 {
public static void main(String args[])
{FrameGraphic.plot("mouse control graphic window",new mouseP1());}
}
```



In the next Mouse example a series of lines are drawn and recorded inside of an array.

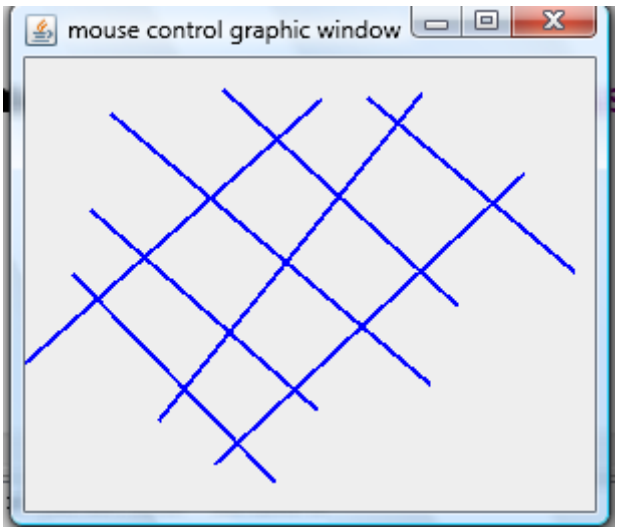
12.1.5 Mouse control JPanel class mouseP2

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  mouseP2 extends JPanel implements MouseListener,MouseMotionListener
{int x1[],y1[],x2[],y2[];
  int n;
  Line2D l[];
  Color c;
 public mouseP2()
 {super();
  x1=new int[100];
  y1=new int[100];
  l=new Line2D[100];
  x2=new int[100];
  y2=new int[100];
  n=0;
  addMouseListener(this);
  addMouseMotionListener(this);
 }

 public void setLine(int x1i,int y1i,int x2i,int y2i)
 {x1[n]=x1i;y1[n]=y1i;x2[n]=x2i;y2[n]=y2i;n++;}

 public void paint(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.blue);
```

```
 g2.setStroke(new BasicStroke(2.0f));
 for(int i=0;i<n;i++)
 {l[i]=new Line2D.Double(x1[i],y1[i],x2[i],y2[i]);
 g2.draw(l[i]);
 }}

//MouseListener (fare dinleyicisi)
public void mouseClicked(MouseEvent e)
 {}

 public void mousePressed(MouseEvent e)
 {
   x1[n]=e.getX();y1[n]=e.getY();
   System.out.println(x1[n]+" "+y1[n]);
 }

  public void mouseReleased(MouseEvent e)
 { x2[n]=e.getX();y2[n]=e.getY();n++;repaint();}

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener (fare hareket dinleyicisi) metotlari

 public void mouseDragged(MouseEvent e)
 {}
 public void mouseMoved(MouseEvent e)
 {}

}
```

12.1.6 Test program for Mouse control JPanel class mouseP2

```
class mouseF1 {
public static void main(String args[])
{
FrameGraphic.plot("mouse control graphic window",new mouseP1());
}
}
```



In the next example Mouse is cliked and a new line is added up to the previously clicked points. Poins are remained to be recorded in an array also in this version.

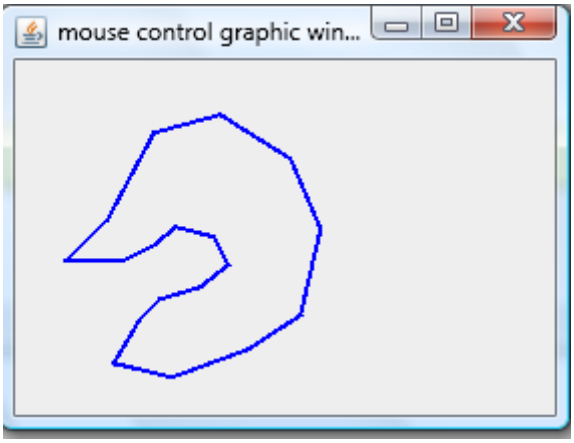
12.1.7 Mouse control JPanel class mouseP3

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  mouseP3 extends JPanel implements MouseListener,MouseMotionListener
{
  //push and push again line draw
            int x1[],y1[],x2[],y2[];
  int n;
  Line2D l[];
  Color c;

 public mouseP3()
 {super();
  x1=new int[100];
  y1=new int[100];
  l=new Line2D[100];
  n=0;
  addMouseListener(this);
  addMouseMotionListener(this);
 }


 public void setLine(int x1i,int y1i,int x2i,int y2i)
 {x1[n]=x1i;y1[n]=y1i;x2[n]=x2i;y2[n]=y2i;n++;}


 public void paint(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.blue);
 g2.setStroke(new BasicStroke(2.0f));
```

```
 for(int i=1;i<n;i++)
 {l[i]=new Line2D.Double(x1[i],y1[i],x1[i-1],y1[i-1]);
 g2.draw(l[i]);
 }
 }

//MouseListener (mouse dinleyicisi)
public void mouseClicked(MouseEvent e)
 {}

 public void mousePressed(MouseEvent e)
 {
    if(n==0) {x1[n]=e.getX();y1[n]=e.getY();System.out.println("n = "+n+" x1= "+x1[n]+"y1 =
"+y1[n]);n++;}
    else { x1[n]=e.getX();y1[n]=e.getY();repaint();System.out.println("n = "+n+"x1 = "+x1[n]+"y1 =
"+y1[n]);n++;}
 }

  public void mouseReleased(MouseEvent e)
 {}
 public void mouseEntered(MouseEvent e)
 { }
 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener (mouse hareket dinleyicisi) metotlari
 public void mouseDragged(MouseEvent e)
 {}
 public void mouseMoved(MouseEvent e)
 {}

}
```

12.1.8 Test program for Mouse control JPanel class mouseP2

```
class mouseF3 {
public static void main(String args[])
{
FrameGraphic.plot("mouse control graphic window",new mouseP3());
}
}
```



The following program is similar to the previous program, but points are connected by a quadratics instead of lines in the previous program.

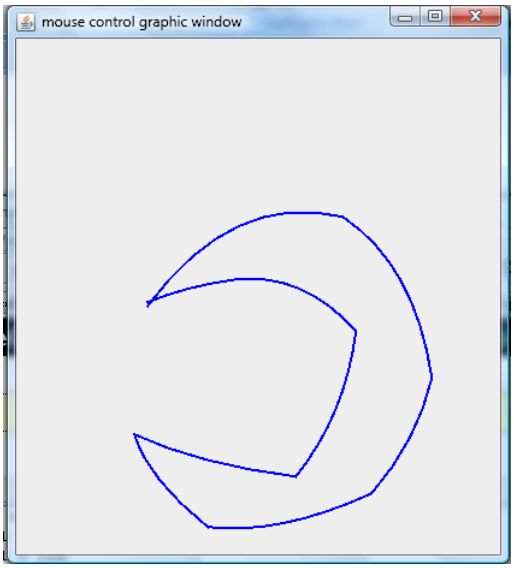
12.1.9 Mouse control JPanel class mouseP3

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  mouseP4 extends JPanel implements MouseListener,MouseMotionListener
{
  //push and push again line draw
            int x1[],y1[],x2[],y2[];
  int n;
  QuadCurve2D l[];
  Color c;

 public mouseP4()
 {super();
  x1=new int[200];
  y1=new int[200];
  l=new QuadCurve2D[100];
  n=0;
  addMouseListener(this);
  addMouseMotionListener(this);
 }
```

```
 public void setLine(int x1i,int y1i,int x2i,int y2i)
 {x1[n]=x1i;y1[n]=y1i;x2[n]=x2i;y2[n]=y2i;n++;}


 public void paint(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(Color.blue);
 g2.setStroke(new BasicStroke(2.0f));
 for(int i=2;i<n;i+=2)
 {l[i]=new QuadCurve2D.Double(x1[i],y1[i],x1[i-1],y1[i-1],x1[i-2],y1[i-2]);
 g2.draw(l[i]);
 }
 }

//MouseListener (mouse dinleyicisi)
public void mouseClicked(MouseEvent e)
 {}

 public void mousePressed(MouseEvent e)
 {
    if(n==0) {x1[n]=e.getX();y1[n]=e.getY();System.out.println("n = "+n+" x1= "+x1[n]+"y1 =
"+y1[n]);n++;}
    else { x1[n]=e.getX();y1[n]=e.getY();repaint();System.out.println("n = "+n+"x1 = "+x1[n]+"y1 =
"+y1[n]);n++;}
 }

  public void mouseReleased(MouseEvent e)
 {}

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener (mouse hareket dinleyicisi) metotlari

 public void mouseDragged(MouseEvent e)
 {}
 public void mouseMoved(MouseEvent e)
 {}
}
```

12.1.10 Test program for Mouse control JPanel class mouseP2

```
class mouseF5 {
public static void main(String args[])
{
FrameGraphic.plot("mouse control graphic window",new mouseP5());
}
}
```



12.2 CONTROL BOX

Main swing class JCheckBox is used to sign some options. JCheckBox is o controller to make some variable on or off . Constructor methods for JChecBox are :

public JCheckBox()
public JCheckBox(Icon picture)
public JCheckBox(Icon picture,boolean boxsign)
public JCheckBox(String writing)
public JCheckBox(String writing, boolean boxsign)
public JCheckBox(String writing, Icon picture)
public JCheckBox(String writing, , Icon picture ,boolean boxsign)

We have option of putting picture in the constructure. In the first example, style of the writing in the JTextField area is changed by JCheckBox selection.

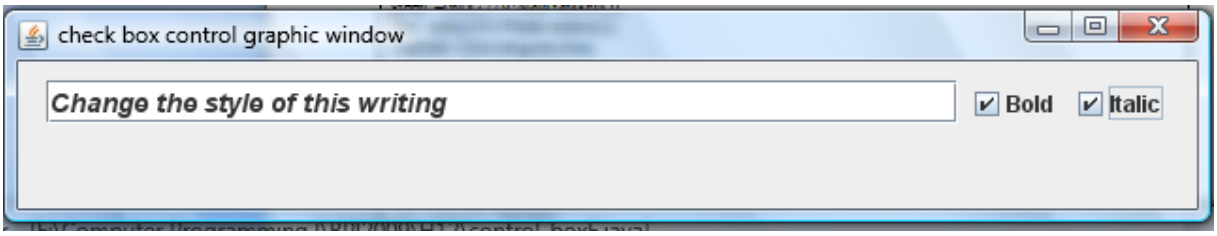
12.1.1 control_boxP controlbox check

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
```

```
public class  control_boxP extends JPanel implements ItemListener
{
  JPanel inputPanel;
  JTextField t;
  JCheckBox bold,italic;
  int normal_font,bold_font,italic_font;
  String s;

  public control_boxP()
  {
            s="Change the style of this writing";
    inputPanel=new JPanel();
    t=new JTextField(s,40);
    italic_font=Font.PLAIN;
            bold_font=Font.PLAIN;
            t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
    bold=new JCheckBox("Bold");
    italic=new JCheckBox("Italic");
    inputPanel.add(t);
    inputPanel.add(bold);
    inputPanel.add(italic);
    add(inputPanel);
    italic.addItemListener(this);
    bold.addItemListener(this);
  }

public void itemStateChanged(ItemEvent e)
{   if(e.getSource()==bold)
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {bold_font=Font.BOLD;}
      else
        {bold_font=Font.PLAIN;}
    }
    else if(e.getSource()==italic)
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {italic_font=Font.ITALIC;}
      else
        {italic_font=Font.PLAIN;}
    }
    t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
    repaint();
}

}
```

12.2.2 Test program for control_boxP

```
class control_boxF {
public static void main(String args[])
{
FrameGraphic.plot("check box control graphic window",new control_boxP());
}
}
```



Next program changes writing style by JCheckBox and color by JButton

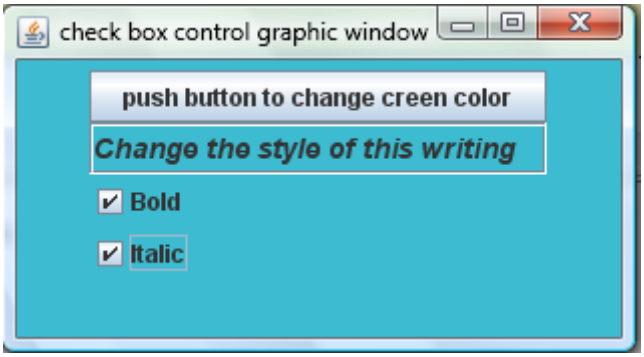
12.1.3 Control_boxP1 controlbox check

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;


public class  control_boxP1 extends JPanel implements ActionListener,ItemListener
{
 JPanel inputPanel;
 JButton b;
 JTextField t;
 JCheckBox bold,italic;
 int red,green,blue;
 int normal_font,bold_font,italic_font;
 public Color renk;
 String s;

 public control_boxP1()
 {
          s="Change the style of this writing";
          renk=Color.blue;
  inputPanel=new JPanel();
  inputPanel.setLayout(new GridLayout(4,1));
  b=new JButton("push button to change creen color");
  t=new JTextField(s);
  italic_font=Font.PLAIN;
          bold_font=Font.PLAIN;
          t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
  bold=new JCheckBox("Bold");
  italic=new JCheckBox("Italic");
  inputPanel.add(b,BorderLayout.NORTH);
  inputPanel.add(t);
  inputPanel.add(bold);
  inputPanel.add(italic);
  add(inputPanel);
  b.addActionListener(this);
  italic.addItemListener(this);
  bold.addItemListener(this);
 }
```

```
public void itemStateChanged(ItemEvent e)
{   if(e.getSource()==bold)
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {bold_font=Font.BOLD;}
      else
        {bold_font=Font.PLAIN;}
    }
    else if(e.getSource()==italic)
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {italic_font=Font.ITALIC;}
      else
        {italic_font=Font.PLAIN;}
    }
    t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
    repaint();
}


public void actionPerformed( ActionEvent e)
{
//tesadüfi renk belirle düğmeye her basılışta rengi değiştir.
red=(int)(255.0*Math.random());
green  =(int)(255.0*Math.random());
blue   =(int)(255.0*Math.random());
renk=new Color(red,green,blue);
//input panelin rengini değiştiriyoruz
inputPanel.setBackground(renk);
//ana panelin rengini değiştiriyoruz
setBackground(renk);
//yazı kutusunun rengini değiştiriyoruz
t.setBackground(renk);
bold.setBackground(renk);
italic.setBackground(renk);
repaint();
}


}
```

12.2.4 Test program for control_boxP1

```
class control_boxF1 {
public static void main(String args[])
{
FrameGraphic.plot("check box control graphic window",new control_boxP1());
}
}
```



12.3 RADIO CONTROL

Java swing JRadioButton class is similar to JcheckBox class. Constructor class :

public JRadioButton()
public JRadioButton(Icon picture)
public JRadioButton(Icon picture,boolean boxsign)
public JRadioButton(String writing)
public JRadioButton(String writing, boolean boxsign)
public JRadioButton(String writing, Icon picture)
public JRadioButton(String writing, , Icon picture ,boolean boxsign)

In addition to JRadioButton one additional support class ButtonGroup. This class constructs a group of controls operating together.Defination of this cass

```
public class ButtonGroup extends Object implements Serializable
{
protected Vector Buttons;
public ButtonGroup();
public void add(AbstractButton ab);
public Enumeration getElements();
public ButtonModel getSelection();
public boolean isSelected(ButtonModel bm);
public void remove(AbstractButton ab);
public void setSelected(ButtonModel bm,boolean b);
}
```

Now, let us look at an example of JRadioButton class

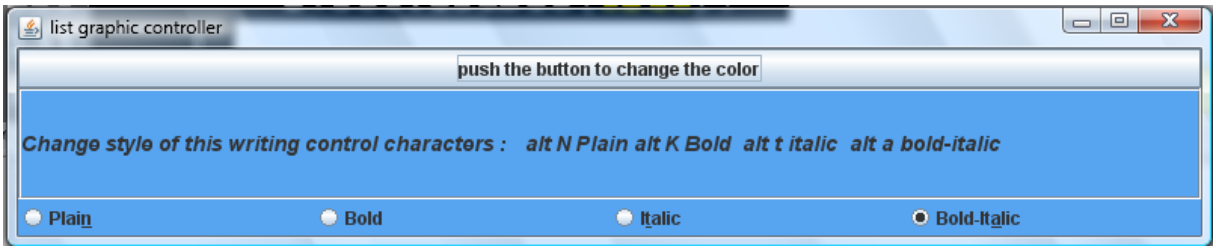
12.3.1 radio_control class, An example of JRadioControl class

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;


public class  radio_control extends JApplet implements ActionListener,ItemListener
{
```

```
JPanel inputPanel;
JButton b;
JTextField t;
JRadioButton plain,bold,italic,bolditalic;
ButtonGroup gurup;
int kirmizi,yesil,mavi;
int bold_font,italic_font;
public Color col;
String s;
public void init()
{
        s="Change style of this writing control characters :   alt N Plain alt K Bold  alt t italic  alt a bold-italic";
        col=Color.blue;
  inputPanel=new JPanel();
  setLayout(new BorderLayout());
  inputPanel.setLayout(new GridLayout(1,4));
  b=new JButton("push the button to change the color");
  t=new JTextField(s);
  italic_font=Font.PLAIN;
        bold_font=Font.PLAIN;
        JPanel  Check=new JPanel();
  Check.setLayout(new GridLayout(4,0));
  gurup=new ButtonGroup();
  plain=new JRadioButton("Plain");
  plain.setMnemonic('N');
  plain.setSelected(true);
  plain.addItemListener(this);
  gurup.add(plain);
  inputPanel.add(plain);
  bold=new JRadioButton("Bold");
  bold.setMnemonic('K');
  bold.setSelected(false);
  bold.addItemListener(this);
  gurup.add(bold);
  inputPanel.add(bold);

 italic=new JRadioButton("Italic");
 italic.setMnemonic('t');
 italic.setSelected(false);
 italic.addItemListener(this);
 gurup.add(italic);
 inputPanel.add(italic);

 bolditalic=new JRadioButton("Bold-Italic");
 bolditalic.setMnemonic('a');
 bolditalic.setSelected(false);
 bolditalic.addItemListener(this);
 gurup.add(bolditalic);
 inputPanel.add(bolditalic);

        t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
  inputPanel.add(t);

  add(b,BorderLayout.NORTH);
  add(t,BorderLayout.CENTER);
  add(inputPanel,BorderLayout.SOUTH);
  b.addActionListener(this);
```

```
  }

public void itemStateChanged(ItemEvent e)
{    Object kutu=e.getItemSelectable();
 if(kutu==plain)
   {
   bold_font=Font.PLAIN;
   italic_font=Font.PLAIN;
   }
   else if(kutu==bold)
   {
   bold_font=Font.BOLD;
   italic_font=Font.PLAIN;
   }
 else if(kutu==italic)
   {
   bold_font=Font.PLAIN;
   italic_font=Font.ITALIC;
   }
 else if(kutu==bolditalic)
   {
   bold_font=Font.BOLD;
   italic_font=Font.ITALIC;
   }
   t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
   repaint();
}


public void actionPerformed( ActionEvent e)
{
//tesadüfi col belirle düğmeye her basılışta rengi değiştir.
kirmizi=(int)(255.0*Math.random());
yesil  =(int)(255.0*Math.random());
mavi   =(int)(255.0*Math.random());
col=new Color(kirmizi,yesil,mavi);
//input panelin rengini değiştiriyoruz
inputPanel.setBackground(col);
//ana panelin rengini değiştiriyoruz
setBackground(col);
//yazı kutusunun rengini değiştiriyoruz
t.setBackground(col);
plain.setBackground(col);
bold.setBackground(col);
italic.setBackground(col);
bolditalic.setBackground(col);
repaint();
}

    public static void main(String s[])
    {
        JFrame f = new JFrame("list graphic controller");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new radio_control();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
```

```
    }
}
```



If Lambda expressions version of the same program is given:

12.3.1 radio_control class, An example of JRadioControl class, Radio expressions form

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class  radio_control_L extends JApplet
{

  JPanel inputPanel;
  JButton b;
  JTextField t;
  JRadioButton plain,bold,italic,bolditalic;
  ButtonGroup gurup;
  int kirmizi,yesil,mavi;
  int bold_font,italic_font;
  public Color col;
  String s;
  public void init()
  {
            s="Change style of this writing control characters :   alt N Plain alt K Bold  alt t italic  alt a bold-italic";
            col=Color.blue;
    inputPanel=new JPanel();
    setLayout(new BorderLayout());
    inputPanel.setLayout(new GridLayout(1,4));
    b=new JButton("push the button to change the color");
    t=new JTextField(s);
    italic_font=Font.PLAIN;
            bold_font=Font.PLAIN;
            JPanel  Check=new JPanel();
    Check.setLayout(new GridLayout(4,0));
    gurup=new ButtonGroup();
    plain=new JRadioButton("Plain");
    plain.setMnemonic('N');
    plain.setSelected(true);
    plain.addItemListener(e1->{bold_font=Font.PLAIN;
```

```
                    italic_font=Font.PLAIN;
                    t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
                    repaint();});
 gurup.add(plain);
 inputPanel.add(plain);
 bold=new JRadioButton("Bold");
 bold.setMnemonic('K');
 bold.setSelected(false);
 bold.addItemListener(e2->{bold_font=Font.BOLD;
                   italic_font=Font.PLAIN;
                   t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
                   repaint();});
 gurup.add(bold);
 inputPanel.add(bold);
 italic=new JRadioButton("Italic");
 italic.setMnemonic('t');
 italic.setSelected(false);
 italic.addItemListener(e2->{bold_font=Font.PLAIN;
                    italic_font=Font.ITALIC;
                    t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
                    repaint();});
gurup.add(italic);
inputPanel.add(italic);
bolditalic=new JRadioButton("Bold-Italic");
bolditalic.setMnemonic('a');
bolditalic.setSelected(false);
bolditalic.addItemListener(e4->{bold_font=Font.BOLD;
                      italic_font=Font.ITALIC;
                      t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
                      repaint();});
 gurup.add(bolditalic);
 inputPanel.add(bolditalic);
        t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
 inputPanel.add(t);
 add(b,BorderLayout.NORTH);
 add(t,BorderLayout.CENTER);
 add(inputPanel,BorderLayout.SOUTH);
 b.addActionListener(e5->{
                  kirmizi=(int)(255.0*Math.random());
                  yesil  =(int)(255.0*Math.random());
                  mavi   =(int)(255.0*Math.random());
                  col=new Color(kirmizi,yesil,mavi);
                  inputPanel.setBackground(col);
                  setBackground(col);
                  t.setBackground(col);
                  plain.setBackground(col);
                  bold.setBackground(col);
                  italic.setBackground(col);
                  bolditalic.setBackground(col);
                  repaint();
                  });
}
```

```
    public static void main(String s[])
    {
        JFrame f = new JFrame("list graphic controller");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new radio_control_L();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

12.4 LIST CONTROL JLIST, JCOMBOBOX AND JSCROLLPANE FORMAT STATEMENT

In order to create a list, swing graphic controller **JList** can be used. When Jlist method is added into a Panel, it can be added directly. It should be added by using JscrollPane formatting. Jlist uses ListSelectionListener Windows listening method. This method is defined in javax.swing.event library. When a selection is done by using JList one or more than one selection can be done. Constructor methods of JLists are as follows:

public JList<E>();
public JList<E>(ListModel lm);
public JList<E> (Object o[]);
public JList<E> (Vector v);
In this definition E can be any class. This type of general class definition is called **Generic** classes. We will investigate Generic classes later in more details.

Some important methods of the class used in selection process :

public int GetSelectedIndex() : index number of selected (or last selected) list object
public int[] getSelectedIndices() : integer array of selected list objects

public Object getSelectedValue() : returns the last selected value as Object

public object[] getSelectedValues() : returns selected values as an array of Object type

void setSelectionMode(int mode) : set the selection mode(either to select one element or more than one element from the list). İnt mode can be taken one of the following values:

ListSelectionModel.SINGLE_SELECTION (one element selection)
ListSelectionModel .INTERVAL_SELECTION (group selection)
ListSelectionModel .MULTIPLE_INTERVAL_SELECTION (more than one group selection)

Public void setVisibleRowCount(int sayı) : set number of elements to be shown in the list

If the visible row count is set a smaller number than the number of elements in the list, a scroll bar will be apar at the side of the list. This scroll will be created by JScrollPane page formatting class.

Constroctor methods of JScrollPane is as follows

public JScrollPane();

public JScrollPane(int type_of_raw_scroll_bar,int type_of_column_scroll_bar);

public JScrollPane(Component name_of_the JPanel_or_Component);

public JScrollPane(Component name_of_the JPanel_or_Component , int type_of_raw_scroll_bar,int type_of_column_scroll_bar);

name_of_the JPanel_or_Component shows the name of Component or JPanel ScrolPAne ıs located. The type of scrol bar controls are :

JScrollPane.VERTICAL_SCROLLBAR_ALWAYS
JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED
JScrollPane.VERTICAL_SCROLLBAR_NEVER
JScrollPane. HORIZONTAL _SCROLLBAR_ALWAYS
JScrollPane. HORIZONTAL _SCROLLBAR_AS_NEEDED
JScrollPane. HORIZONTAL _SCROLLBAR_NEVER

Now some examples can be given for JList controller. In the first example a color is selected and the backgound color of the window is changed.

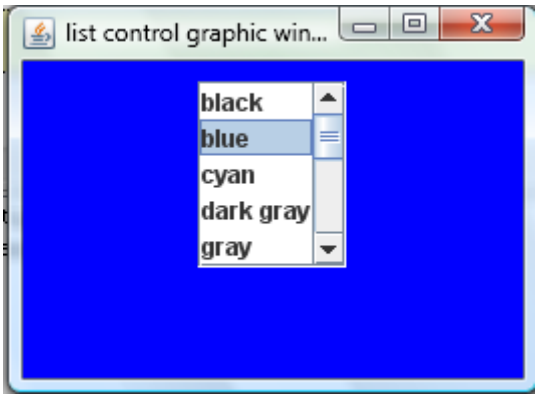
12.4.1 listP class : an example of JList class

```
 import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;

public class  listP extends JPanel implements ListSelectionListener
{ JPanel inputPanel;
  JList<String> colorList;
  String colornames[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color colors[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                  Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                  Color.red,Color.white,Color.yellow};
  public listP()
  {
             colorList=new JList<String>(colornames);
             colorList.setVisibleRowCount(5);
```

```
    inputPanel=new JPanel();
    inputPanel.add(new JScrollPane(colorList));
    add(inputPanel);
    colorList.addListSelectionListener(this);
  }

public void valueChanged( ListSelectionEvent e)
{
Color r1=colors[colorList.getSelectedIndex()];
inputPanel.setBackground(r1);
setBackground(r1);
repaint();
}
}
```

12.4.2 A test program for listP JPanel

```
class listF {
public static void main(String args[])
{FrameGraphic.plot("list control graphic window",new listP());}
}
```



In the second example, one list is changing the shape drawn in the window, the second list is changing the color of the shape.

12.4.3 shapeP class : a subclass to draw shapes as an example of JList class

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class shapeP extends JPanel
{
Color c;
int shape;
//shape = 0 ellipse
//shape = 1 star

public shapeP(Color c1,int shape1)
{c=c1;shape=shape1;}
```

```
public void setColor(Color c1)
{c=c1;}

public void setShape(int shape1)
{shape=shape1;}

public void paintComponent(Graphics g)
{
super.paintComponent(g);
if(shape==0)
{
int x[]={55,67,109,73,83,55,27,37,1,43};
int y[]={0,36,36,54,96,72,96,54,36,36};
Graphics2D g2=(Graphics2D)g;
GeneralPath star=new GeneralPath();
star.moveTo(x[0],y[0]);
for(int i=1;i<x.length;i++)
   {star.lineTo(x[i],y[i]);}
star.closePath();
g2.setColor(c);
g2.fill(star);
}
else if(shape==1)
{
    Graphics2D g2=(Graphics2D)g;
    g2.setFont(new Font("Serif",Font.BOLD,24));
    g2.setColor(c);
    g2.setStroke(new BasicStroke(2.0f));
    Ellipse2D x=new Ellipse2D.Double(50,50,500,200);
    g2.fill(x);
}}}
```

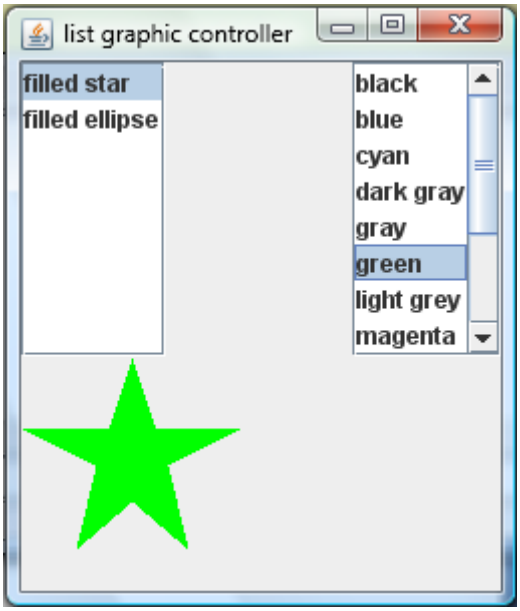
12.4.4 listP1 JPanel class : an example of JList class

```
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  listP1  extends JPanel implements ListSelectionListener
{
  shapeP shape;
  JPanel inputPanel;
  JPanel outputPanel;
  JList<String> shapelist;
  JList<String> colorlist;
  String shapenames[]={"filled star","filled ellipse"};
  String colornames[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color colors[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                  Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                  Color.red,Color.white,Color.yellow};

  public listP1()
  {
                int i=0;
                shape=new shapeP(Color.black,i);
                colorlist=new JList<String>(colornames);
                colorlist.setVisibleRowCount(5);
```

```
                shapelist=new JList<String>(shapenames);
    inputPanel=new JPanel();
    inputPanel.setBackground(Color.YELLOW);
    inputPanel.setLayout(new BorderLayout());
    this.setLayout(new BorderLayout());
    outputPanel=new JPanel();
    outputPanel.setLayout(new BorderLayout());
    outputPanel.add(shape,BorderLayout.CENTER);
    inputPanel.add(new JScrollPane(colorlist),BorderLayout.EAST);
    inputPanel.add(new JScrollPane(shapelist),BorderLayout.WEST);
    outputPanel.add(shape,BorderLayout.CENTER);
    add(inputPanel,BorderLayout.NORTH);
    add(outputPanel,BorderLayout.CENTER);
    colorlist.addListSelectionListener(this);
    shapelist.addListSelectionListener(this);
  }

public void valueChanged( ListSelectionEvent e)
{
if(e.getSource()==colorlist)
{
Color r1=colors[colorlist.getSelectedIndex()];
shape.setColor(r1);
}
if(e.getSource()==shapelist)
{shape.setShape(shapelist.getSelectedIndex());}
repaint();
}


}
```

12.4.5 listA1 JApplet class : an example of JList class

```
 import java.awt.event.*;

public class  listA1 extends JApplet
{
 JPanel pp;

  public void init()
  {
  pp=new listP1();
  add(pp);
  }

   public static void main(String s[])
   {
       JFrame f = new JFrame("JLabel, JTextFiled, JTextArea graphic window");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       JApplet applet = new listA1();
       f.getContentPane().add("Center", applet);
       applet.init();
       f.setSize(800,1000);
       f.setVisible(true);
   }
}
```



JComboBox class is similar to JList class. Constructor classes of JComboBox is as follows:

public JComboBox<E>();
public JComboBox <E>(ListModel lm);
public JComboBox <E>(E[] girdi_nesneleri);
public JList<E>(Vector|E> girdi_nesneleri);
In this definitions E can be any class. This type of general class definition is called **Generic** classes. We will investigate Generic Classes Later in more details

12.4.6 comboP JComboBox example JPanel program

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;

public class  comboP extends JPanel implements ActionListener
{ JPanel inputPanel;
  JComboBox<String> colorList;
  String colornames[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color colors[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                  Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                  Color.red,Color.white,Color.yellow};
  public comboP()
  {
              colorList=new JComboBox<String>(colornames);
    inputPanel=new JPanel();
    inputPanel.add(new JScrollPane(colorList));
    add(inputPanel);
    colorList.addActionListener(this);
  }

public void actionPerformed( ActionEvent e)
```

```
{
Color r1=colors[colorList.getSelectedIndex()];
inputPanel.setBackground(r1);
setBackground(r1);
repaint();
}
}
```

12.4.7 comboP JComboBox JFrame program

```
class comboF {
public static void main(String args[])
{FrameGraphic.plot("ComboBox control",new comboP());}}
```

The second example is a selectable JComboBox. In this one JComboBox behaves as both JTextField and JComboBox (or JList) together.

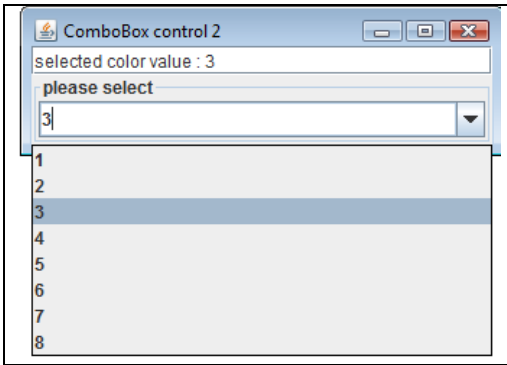
12.4.8 comboP2 JComboBox example JPanel program

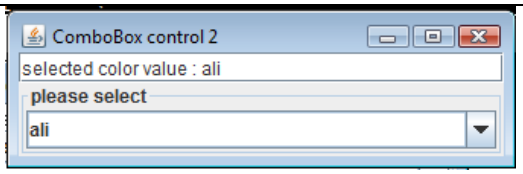
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class comboP2 extends JPanel implements ActionListener
{   private static final long serialVersionUID = 99075867L;
            String[] s = { "1", "2", "3", "4", "5", "6", "7", "8" };
   ComboBoxModel<String> mode;
   JTextField jtf;
            public comboP2() {
                        JFrame f = new JFrame("JComboBox2");
                        Container contentPane = f.getContentPane();
                        jtf=new JTextField(s[0]);
                        mode = new UserDefineComboBoxModel();
                        JComboBox<String> combo = new JComboBox<String>(mode);
                        combo.setEditable(true);
                        combo.setBorder(BorderFactory.createTitledBorder("please select"));
                        setLayout(new BorderLayout());
                        add(jtf,BorderLayout.NORTH);
                        add(combo,BorderLayout.CENTER);
                        combo.addActionListener(this);
            }

  public void actionPerformed( ActionEvent e)
  {   JComboBox<String> cb = (JComboBox<String>)e.getSource();
     String scolor= (String)cb.getSelectedItem();
     jtf.setText("selected color value : "+scolor);
  }

            class UserDefineComboBoxModel<String> extends AbstractListModel implements
                                    ComboBoxModel {
                        String item = null;
                        public Object getElementAt(int index)
                        {return s[index++];}
                        public int getSize()
                        {return s.length;}
                        public void setSelectedItem(Object anItem) {item = (String) anItem;}
```

```
                    public Object getSelectedItem() {return item;}
          }
}
```

12.4.9 comboP JComboBox JFrame program

```
class comboF2 {
public static void main(String args[])
{FrameGraphic.plot("ComboBox control 2",new comboP2());}}
```





12.5 MENU CONTROL

Anybody using Windows type graphic user interphases used to using menus, lists that opens up from the top of the windows. Menus of java GUI consist of JMenuBar, JMenu and JMenuItem classes. Inside of Java swing menus elements such as JRadioButtonMenuItem, JCheckBoxMenuItem can also be used. **JmenuBar** constroctor method is:

public JmenuBar();

Some important methods of JMenuBar are :

public JMenu add(JMenu menu); :
public Component getComponent();
public Component getComponentAtIndex(int indeks);
public Insets getMargin();
public Jmenu getMenu(int)
public int getMenuCount();
public SingleSelectionModel getSelectionModel();
public MenuElement[] getSubElements();
protected void paintBorder(Graphics g);
public void processMouseEvent(MouseEvent e,MouseElement path[],MouseSelectionManager mgr);
public void setBorderPainted(boolean b);
public void setHelpMenu(Jmenu menu);
public void setMargin(Insets in);
pubic void setSelected(Component c);

Jmenu constroctor methods :

public JMenu();
public JMenu(String Label);
public JMenu(String label,boolean menudenayır);

Some important JMenu methods :

public JMenuItem add(Action act);
public JMenuItem add(JMenuItem item);
public Component add(Component c);
public void add(String Jlabel);
public void addMenuListener(MenuListener l);
public void addSeperator();
public JMenuItem getItem(int pos);
public int getItemCount();
public Component getMenuComponent(int pos);
public int getMenuComponentCount();
public Component[] getMenuComponents(public MenuElement[] getSubElements();
public JMenuItem insert(JMenuItem item,int pos);
public void insert(String etiket,int pos);
public void remove(int pos);
public void remove(MenuItem item);
public void removeAll();
public void removeMenuListener(MenuListener l);
public void setAccelerator(KeyStroke s);
public void setDealy(int delay);
public void setMenuLocation(int x,int y);
public void setPopUpMenuVisible(boolean b)
public void setSelected(boolean b);

JMenuItem constroctor methods :
public JMenuItem();
public JMenuItem(Icon picture);
public JMenuItem(String Label);
public JMenuItem(String Label,int mnemonic);
public JMenuItem(String Label,Icon picture);
some of the methods :
public Component getComponent();
public menuElement[] getSubElements();

JCheckBoxMenuItem, is a subclass of JMenuItem. It behaves sımılar to JCheckBox class. Contructor classes are :
public JCheckBoxMenuItem();
public JCheckBoxMenuItem(Icon pictue);
public JCheckBoxMenuItem(String Label);
public JCheckBoxMenuItem(String Label,Icon picture);
public JCheckBoxMenuItem(String Label,Boolean isselected);
public JCheckBoxMenuItem(String Label,Icon picture, Boolean isselected);

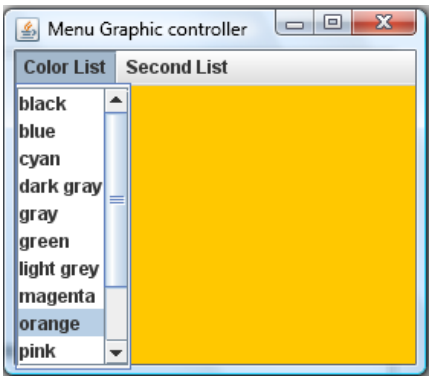
JRadioButtonMenuItem is also a subclass of JMenuItem. Constroctor methods :
public JRadioButtonMenuItem ();
public JRadioButtonMenuItem (Icon picture);
public JRadioButtonMenuItem (String Label);
public JRadioButtonMenuItem (String Label,Icon picture);

Now we can looked at some examples. In the first example, page is painted according to color selected from the menu.

12.5.1 Menu1 program as a menu example

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;
import java.awt.Graphics;

public class  Menu1 extends JFrame implements ListSelectionListener
{
  JPanel inputPanel;
  JList colorlist;
  JMenuBar menuBar;
  JMenu m1,m2;
  String colornames[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color colors[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                  Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                  Color.red,Color.white,Color.yellow};

  public Menu1()
  {
            super("Menu Graphic controller");
    inputPanel=new JPanel();
            colorlist=new JList(colornames);
    colorlist.setVisibleRowCount(10);
    colorlist.addListSelectionListener(this);
    menuBar=new JMenuBar();
            setJMenuBar(menuBar);
            m1 = new JMenu("Color List");
    m1.setMnemonic('M');
    m1.getAccessibleContext().setAccessibleDescription("Programdaki sadece bu Menunun alt listeleri mevcuttur");
    m1.add(new JScrollPane(colorlist));
    menuBar.add(m1);
    m2 = new JMenu("Second List");
    menuBar.add(m2);
    add(inputPanel);
  }


public void valueChanged( ListSelectionEvent e)
{
Color r1=colors[colorlist.getSelectedIndex()];
inputPanel.setBackground(r1);
}

public static void main(String s[])
    {
```

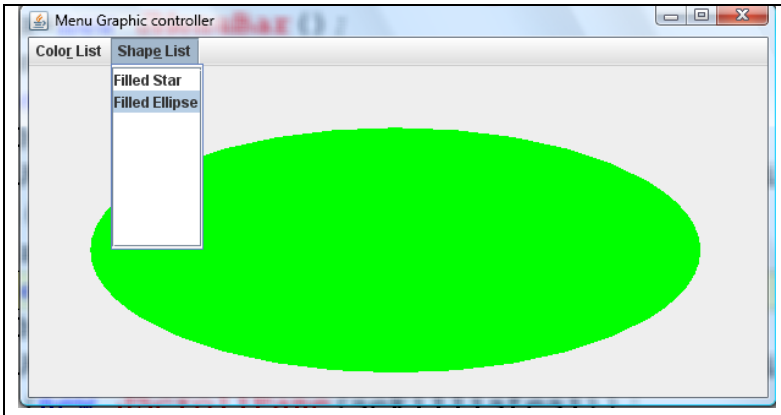
```
        Menu1 f = new Menu1();
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```

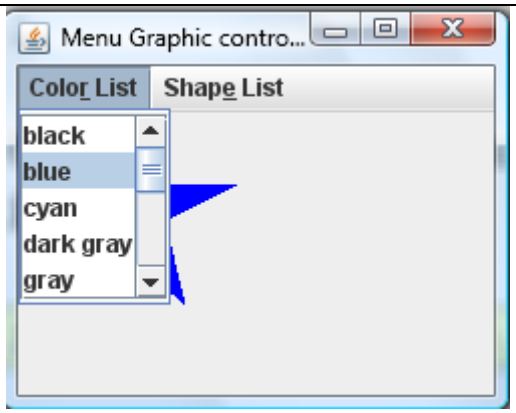


12.5.2 Menu2 program as a menu example

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;
import java.awt.Graphics;

public class  Menu2 extends JFrame implements ListSelectionListener
{
  Container c;
  shapeP sekil;
  JMenuBar menuBar;
  JMenu m1,m2;
  JPanel outputPanel;
  JList sekillistesi;
  JList renklistesi;
  String sekilisimleri[]={"Filled Star","Filled Ellipse"};
  String renkisimleri[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color renkler[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                   Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                   Color.red,Color.white,Color.yellow};

   public Menu2()
  {
              super("Menu Graphic controller");
              int i=0;
              sekil=new shapeP(Color.black,i);
              renklistesi=new JList(renkisimleri);
```

```
            renklistesi.setVisibleRowCount(5);
            sekillistesi=new JList(sekilisimleri);
    c=getContentPane();
    outputPanel=new JPanel();
    outputPanel.setLayout(new BorderLayout());
    outputPanel.add(sekil,BorderLayout.CENTER);
    menuBar=new JMenuBar();
            setJMenuBar(menuBar);
            m1 = new JMenu("Color List");
    m1.setMnemonic('R');
    m1.getAccessibleContext().setAccessibleDescription("Şekillerin hangi renkte olacağını seçer");
    m1.add(new JScrollPane(renklistesi));
    menuBar.add(m1);
    m2 = new JMenu("Shape List");
    m2.setMnemonic('e');
    m2.getAccessibleContext().setAccessibleDescription("Şekillerin hangi şekil olacağını seçer");
    m2.add(new JScrollPane(sekillistesi));
    menuBar.add(m2);
    c.add(outputPanel,BorderLayout.CENTER);
    renklistesi.addListSelectionListener(this);
    sekillistesi.addListSelectionListener(this);
  }


public void valueChanged( ListSelectionEvent e)
{
if(e.getSource()==renklistesi)
{
Color r1=renkler[renklistesi.getSelectedIndex()];
sekil.setColor(r1);
}
if(e.getSource()==sekillistesi)
{sekil.setShape(sekillistesi.getSelectedIndex());}

repaint();
}

public static void main(String s[])
    {
        Menu2 f = new Menu2();
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```





12.6 JSLIDER GRAPHIC CONTROL

JSlider is a controller to control a variable between given limits. For example percentages can be controlled by a slider. Constructor classes for JSlider class:

public JSlider();
public JSlider(BoundedRAngeModel brm);
public JSlider(int konum);
public JSlider(int minimumdeğer, int maksimumdeğer);

In the example program radius of a circle will be changed by reading the value of the radius from JSlider class.

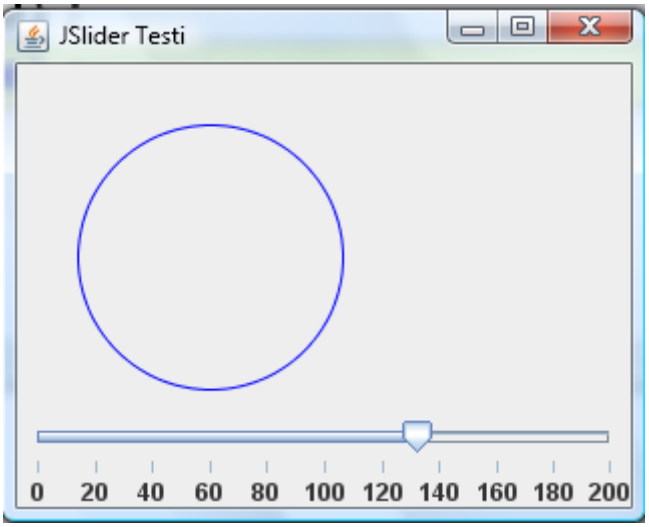
12.6.1 ovalP an example JPanel class to draw a variable diameter radius.

```
import javax.swing.*;
import java.awt.*;
import java.awt.geom.*;

public class ovalP extends JPanel
{
int xi,yi,radius;

public ovalP(int xi1, int yi1,int radius1)
{
xi=xi1;
yi=yi1;
radius=(radius1 >= 0 ? radius1:10);
}

public void setOvalP(int xi1, int yi1,int radius1)
{
xi=xi1;
yi=yi1;
radius=(radius1 >= 0 ? radius1:10);
repaint();
}

public void paintComponent(Graphics g)
```

```
{
 super.paintComponent(g);
 g.setColor(Color.BLUE);
 Graphics2D g2=(Graphics2D)g;
 g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
 RenderingHints.VALUE_ANTIALIAS_ON);
 Ellipse2D elips1=new Ellipse2D.Double(xi,yi,radius,radius);
 g2.draw(elips1);
}
}
```

12.6.2 SliderTest JSlider example class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;

public class SliderTest extends JFrame implements ChangeListener
{
   //
   private ovalP p;
   private JSlider cap;
   Color r=Color.lightGray;
   // pencereyi baslatma metodu
   public SliderTest()
   {
   super("JSlider Test");
   p=new ovalP(30,30,30);
   cap=new JSlider(SwingConstants.HORIZONTAL,0,200,10);
   cap.setMajorTickSpacing(20);
   cap.setPaintTicks(true);
   cap.setPaintLabels(true);
   cap.addChangeListener(this);
   add(p,BorderLayout.CENTER);
   add(cap,BorderLayout.SOUTH);
   }
   // girdi alanindaki olan olaylari dinleme metodu
   public void stateChanged(ChangeEvent e)
   {
    p.setOvalP(30, 30,cap.getValue());
    repaint();
   }

    //=====================================================
     public static void main(String[] args)
     {
     SliderTest f= new SliderTest();
     f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     f.setSize(350,200);
     f.setVisible(true);
     }
}
```



12.7 JTABLE GRAPHIC CLASS

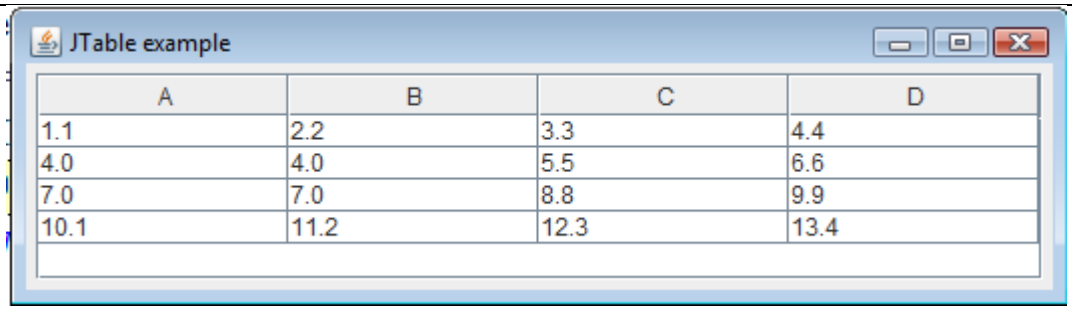
JTable creates outputs in table form. In order to Show using of the class a JTable example class is created to take printout pf a matrix in JTable

Program 12.7.1 JTableP Jpanel example

```
import java.io.*;
import java.util.*;
import javax.swing.*;
 import javax.swing.table.*;
 import java.awt.*;
    public class JTableP extends JPanel
    {
                private static final long serialVersionUID = 43567894L;
                public void kur(Object a[][],String heading[])
                {
                int n=a.length;
                int m=a[0].length;
                JTable jt=new JTable(a,heading);
                jt.setPreferredScrollableViewportSize(new Dimension(500,80));
                jt.setFillsViewportHeight(true);
    add(new JScrollPane(jt));
    }
    public void kur(Object a[][])
                { int n=a[0].length;
                  String heading[]=new String[n];
                  char b='\u0041';
      for(int i=0;i<n;i++)
      {heading[i]=""+b;b++;}
      kur(a,heading);
    }
    public JTableP(Object a[][],String heading[])
    {kur(a,heading);}
                public JTableP(Object a[][])
                { kur(a);}
                public JTableP(double b[][])
                { Object a[][]=doubletoDouble(b);
                  kur(a);
                }
                public JTableP(double b[][],String heading[])
                { Object a[][]=doubletoDouble(b);
                  kur(a,heading);
                }
                public JTableP(int b[][])
```

```
            { Object a[][]=inttoInteger(b);
              kur(a);
            }
            public JTableP(int b[][],String heading[])
            { Object a[][]=inttoInteger(b);
              kur(a,heading);
            }
            public static void print(Object a[][],String heading[],String bb)
{
            JTableP pp=new JTableP(a,heading);
JFrame frame=new JFrame(bb);
frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
frame.getContentPane().add(pp);
frame.pack();
frame.setVisible(true);
            }
            public static void print(Object a[][],String bb)
            {
              int n=a[0].length;
              String heading[]=new String[n];
              char b='\u0041';
  for(int i=0;i<n;i++)
  {heading[i]=""+b;b++;}
  JTableP pp=new JTableP(a,heading);
              JFrame frame=new JFrame(bb);
  frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
  frame.getContentPane().add(pp);
  frame.pack();
  frame.setVisible(true);
            }
            public static void print(Object a[][])
            {                int n=a[0].length;
              String heading[]=new String[n];
              char b='\u0041';
  for(int i=0;i<n;i++)
  {heading[i]=""+b;b++;}
  JTableP pp=new JTableP(a,heading);
              JFrame frame=new JFrame("JTable");
  frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
  frame.getContentPane().add(pp);
  frame.pack();
  frame.setVisible(true);
            }
            public static void print(double a[][],String heading[],String bb)
            {print(doubletoDouble(a),heading,bb);}
            public static void print(double a[][],String bb)
            {print(doubletoDouble(a),bb);}
            public static void print(double a[][])
            {print(doubletoDouble(a));}
            public static void print(int a[][],String heading[],String bb)
            {print(inttoInteger(a),heading,bb);}
            public static void print(int a[][],String bb)
            {print(inttoInteger(a),bb);}
            public static void print(int a[][])
            {print(inttoInteger(a));}
            public static Double[][] doubletoDouble(double xi[][])
{ int n=xi.length;
int m=xi[0].length;
Double X[][]=new Double[n][m];
for(int j=0;j<m;j++)
  {for(int i=0;i<n;i++)
  {X[i][j]=new Double(xi[i][j]);}
  }
return X;
}
public static Integer[][] inttoInteger(int xi[][])
```

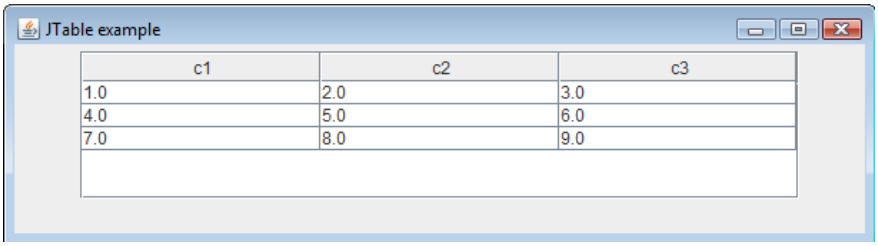
```
  { int n=xi.length;
  int m=xi[0].length;
  Integer X[][]=new Integer[n][m];
  for(int j=0;j<m;j++)
    {for(int i=0;i<n;i++)
    {X[i][j]=new Integer(xi[i][j]);}
    }
  return X;
 }

            public static void main(String arg[])
            {double a[][]={{1.1,2.2,3.3,4.4},{4,4,5.5,6.6,7,7},{7,7,8.8,9.9,10.1},{10.1,11.2,12.3,13.4}};
            print(a,"JTable example");}
}
```



We can directly use previous example through print methods or input JPanel into any JFrame and use it.

Program 12.7.2 JTabletest JPanel example

```
            import java.io.*;
            import java.util.*;
            import javax.swing.*;
  import javax.swing.table.*;
  import java.awt.*;
 public class JTabletest
 {
            public static void main(String arg[])
            {
            double a[][]={{1,2,3},{4,5,6},{7,8,9}};
            String s[]={"c1","c2","c3"};
            String s1="JTable example";
            JTableP pp=new JTableP(a,s);
            FrameGraphic.plot(s1,pp);
 }}
```



Historically JTable class was only used through JModel Class. Using JModel class to create JTables still have advantages. In the next example JTable class defined through a JModel class.

Program 12.7.3 gm TableModel example

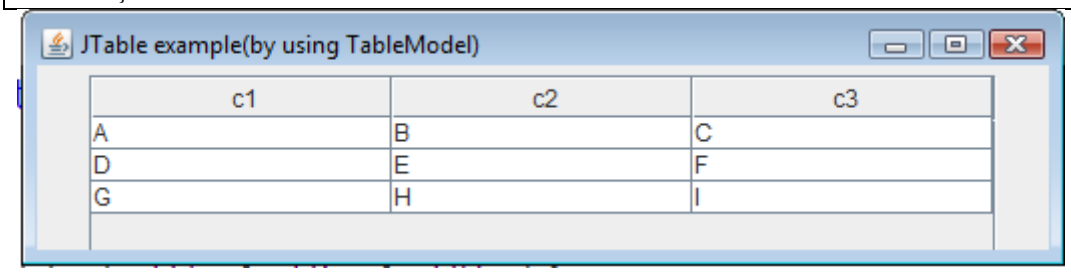
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
import java.io.*;
```

```
import java.util.*;

class gm extends AbstractTableModel implements TableModelListener
{  private static final long serialVersionUID = 7432664L;
  Object[][] veri;
  String[] baslik;

  public gm(Object[][] verii,String basliki[])
  {
    int n=verii.length;
    int m=verii[0].length;
    veri=new Object[n][m];
    baslik=new String[m];
    for(int j=0;j<m;j++)
    {for(int i=0;i<n;i++)
       {veri[i][j]=verii[i][j];}
       baslik[j]=basliki[j];
    }
    addTableModelListener(this);
  }

  public int getRowCount() {return veri.length;}

  public int getColumnCount() {return baslik.length;}

  public Object getValueAt(int satir,int sutun) {return veri[satir][sutun];}

  public String getColumnName(int c) {return baslik[c];}

  public void setValueAt(Object val, int row, int col)
  {veri[row][col] = val;}


  public void tableChanged(TableModelEvent e)
  {    }

  public void setValues(Object[][] verii,String basliki[])
  { veri=verii;
    baslik=basliki;
  }

  public Object[][] getValues(){return veri;}

  public boolean isCellEditable(int row, int col) {return true;}
}
```

Program 12.7.4 JTable created by using TableModel

```
                import java.io.*;
                import java.util.*;
      import javax.swing.*;
      import javax.swing.table.*;
      import  java.awt.*;
    public class JTabletest1
    {
              public static void main(String arg[])
              {
              String a[][]={{"A","B","C"},{"D","E","F"},{"G","H","I"}};
              String s[]={"c1","c2","c3"};
              String s1="JTable example(by using TableModel)";
              gm g=new gm(a,s);
              JTable pp=new JTable(g);
              JPanel jp=new JPanel();
              jp.add(new JScrollPane(pp));
              JFrame jf=new JFrame(s1);
              jf.add(jp);
              jf.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
```

```
    jf.setSize(800,500);
    jf.setVisible(true);
            }

            }
```



12.8 FORMATTING GRAPHIC PAGE

Graphic controllers can be located into the desired area and desired shape by using graphic format controllers or Layout controls. There are several different Layouts to format graphic pages in Java. Some of these already shown inour previous examples. For example BorderLayout is frequently used in the examples.

Border Layout

BorderLayout divides screen into 5 area **WEST,EAST,NORTH,SOUTH** and **CENTER**. Constructor methods are:

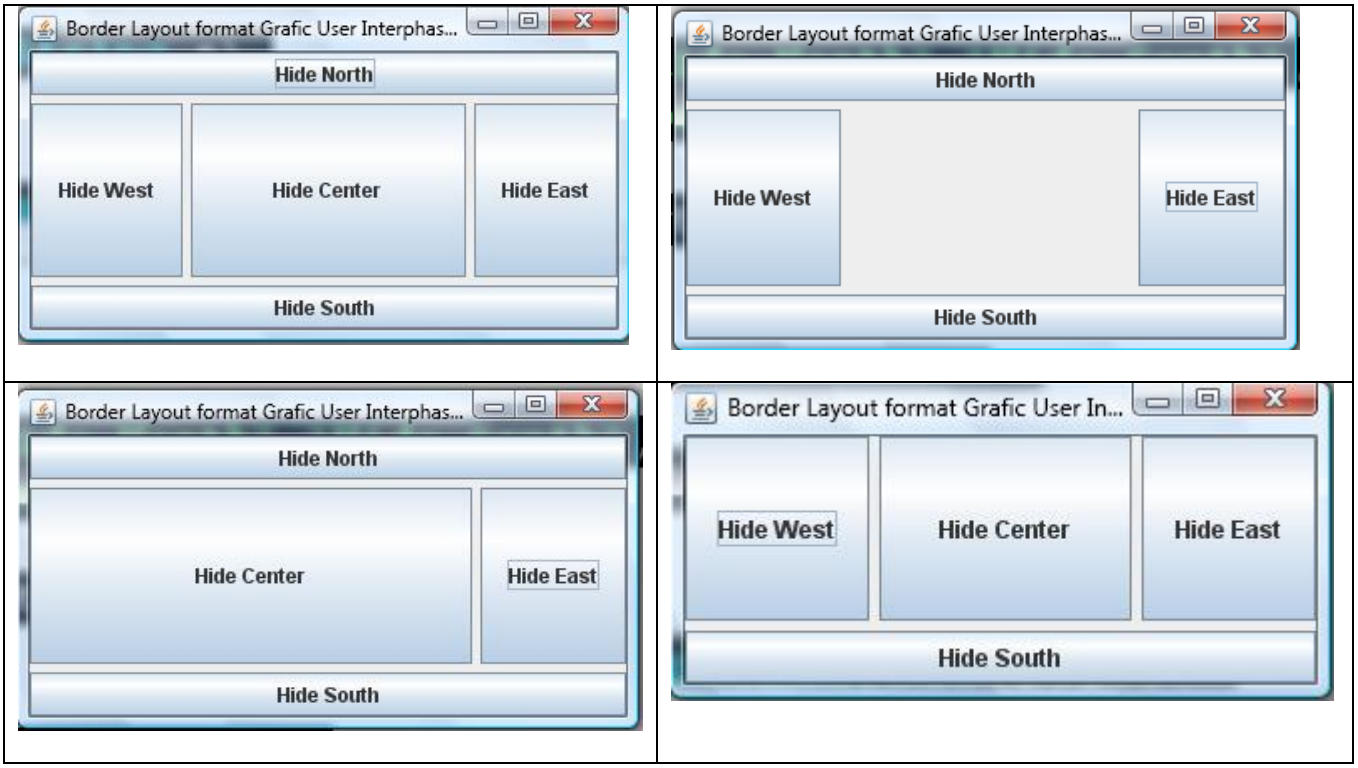
public BorderLayout()
public BorderLayout(int horizontal_space,int vertical_space);

horizonal_space and vertical_space is taken as pixels. WE will explain these with an example. In the example the button pushed will be hide.

12.8.1 BorderLayoutTest example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
 public class BorderLayoutTest extends JFrame implements ActionListener
 {
 private JButton B[];
 private String S[]={"Hide North","Hide South","Hide East",
 "Hide West","Hide Center"};

 public BorderLayoutTest(String s)
 {
 super(s);
 setLayout(new BorderLayout(5,5));
 B=new JButton[5];
 for(int i=0;i<B.length;i++)
 {
 B[i]=new JButton(S[i]);
 B[i].addActionListener(this);
 }
 add(B[0],BorderLayout.NORTH);
 add(B[1],BorderLayout.SOUTH);
 add(B[2],BorderLayout.EAST);
```

```
add(B[3],BorderLayout.WEST);
add(B[4],BorderLayout.CENTER);
}

public void actionPerformed(ActionEvent e)
{
for(int i=0;i<B.length;i++)
{
if(e.getSource()==B[i])
B[i].setVisible(false);
else B[i].setVisible(true);
}
validate();
}
    public static void main(String[] args)
    {
      BorderLayoutTest f = new BorderLayoutTest("Border Layout format Grafic User Interphase(GUI) ");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      f.setSize(800,500);
      f.setVisible(true);
    }
}
```



Flow Layout

FlowLayout class will put different controllers in a row following each other. Constructor methods:

public Flowlayout()
public Flowlayout(int formatregion)
public Flowlayout(int formatregion, int horizontal_space,int vertical_space);

formatregion variable can be taken the values

FlowLayout.RIGHT,

FlowLayout.CENTER or

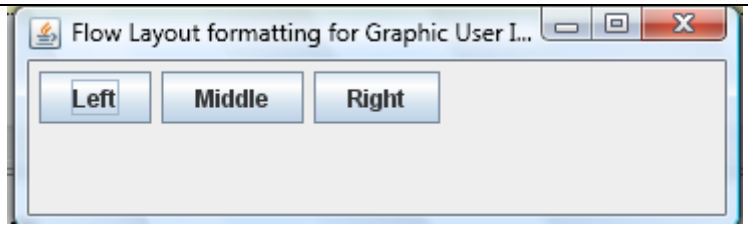
FlowLayout.LEFT

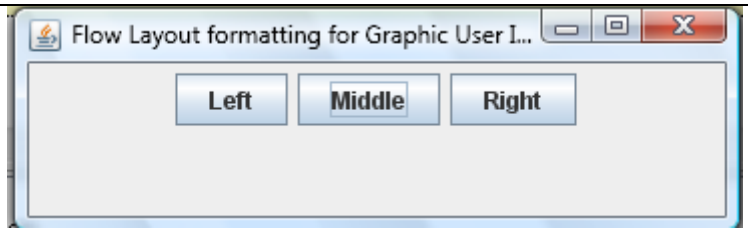
In the example program controllers is changing space of the buttons.

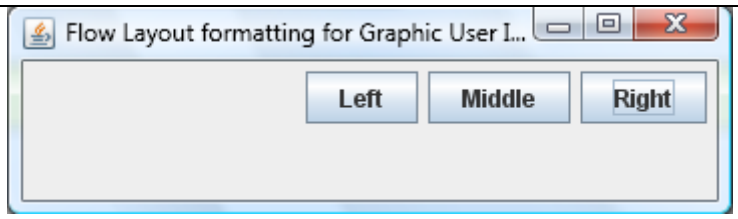
12.8.2 FlowLayoutTest example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class FlowLayoutTest extends JFrame implements ActionListener
{
private JButton B[];
private Container c;

public FlowLayoutTest(String s)
{
super(s);
setLayout(new FlowLayout());
B=new JButton[3];
B[0]=new JButton("Left");
B[1]=new JButton("Middle");
B[2]=new JButton("Right");
for(int i=0;i<B.length;i++)
{
B[i].addActionListener(this);
add(B[i]);
}
}

public void actionPerformed(ActionEvent e)
{
int position=FlowLayout.LEFT;;
if(e.getSource()==B[0])
{
position=FlowLayout.LEFT;
}
else if(e.getSource()==B[1])
{
position=FlowLayout.CENTER;
}
else if(e.getSource()==B[2])
{
position=FlowLayout.RIGHT;
}
setLayout(new FlowLayout(position));
validate();
}
    public static void main(String[] args)
    {
```

```
        FlowLayoutTest f = new FlowLayoutTest("Flow Layout formatting for Graphic User Interphase (GUI) testi");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(800,500);
        f.setVisible(true);
      }
  }
```







Grid Layout

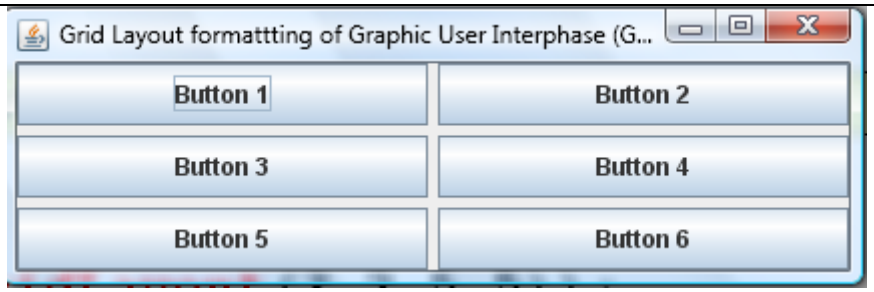
GridLAyout class divides the graphic area into the equal size grid elements. Constroctor methods are:

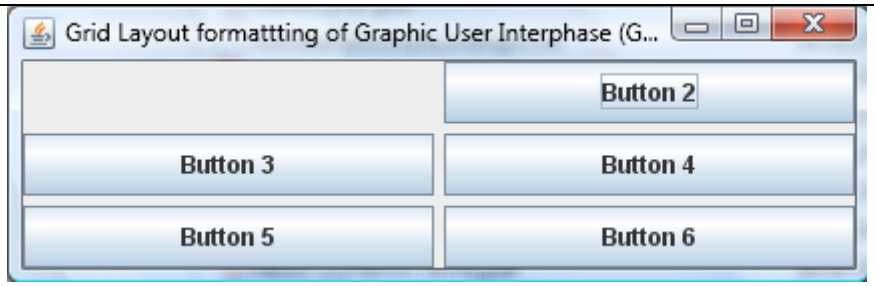
public GridLayout(int rows,int columns)
publicGridLayout(int rows,int columns,int horizontal_space,int vertical_space);

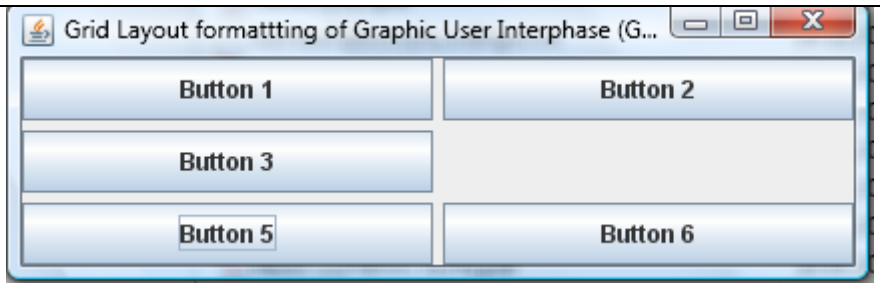
12.8.3 FlowLayoutTest example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
 public class GridLayoutTest extends JFrame implements ActionListener
 {
 private JButton B[];
 private String S[]={"Button 1","Button 2","Button 3",
 "Button 4","Button 5","Button 6"};

 public GridLayoutTest(String s)
 {
 super(s);
 setLayout(new GridLayout(3,2,5,5));
 B=new JButton[6];
 for(int i=0;i<B.length;i++)
```

```
{
B[i]=new JButton(S[i]);
B[i].addActionListener(this);
add(B[i]);
}
}

public void actionPerformed(ActionEvent e)
{
for(int i=0;i<B.length;i++)
{
if(e.getSource()==B[i])
B[i].setVisible(false);
else B[i].setVisible(true);
}
validate();
}
    public static void main(String[] args)
    {
      GridLayoutTest f = new GridLayoutTest("Grid Layout formattting of Graphic User Interphase (GUI)");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      f.setSize(800,500);
      f.setVisible(true);
    }
}
```







JOptionPane class

JOptionPane class is a general graphic class for input and output of the variables through a predefined graphic usr interface. This class is widely used so far for standart String type input and output. But the class taken Object as a general input and output. By using Object class input and output any kind of data can e transferred through this class.

Some of the methods in the JOptionPane is s follows:

public static String showInputDialog(Object message)
public static String showInputDialog(Component mainwindow,Object message)
public static String showInputDialog(Component mainwindow,Object message,String heading,int messagetype)
public static String showInputDialog(Component mainwindow,Object message,String heading,int messagetype, Icon resim, Object[] listselectionvalue, Object entrancevalue)
public static String showInternalInputDialog(Component mainwindow,Object message)
public static String showInternalInputDialog(Component mainwindow,Object message, String heading,int messagetype)
public static String showInternalInputDialog(Component mainwindow,Object message,String heading,int messagetype, Icon resim, Object[] listselectionvalue,, Object girişdeğeri)
public static void showMessageDialog(Component mainwindow,Object message);
public static void showMessageDialog(Component mainwindow,Object message,String heading,int messagetype);
public static void showMessageDialog(Component mainwindow,Object message,String heading,int messagetype, Icon resim);
public static void showInternalMessageDialog(Component mainwindow,Object message);
public static void showInternalMessageDialog(Component mainwindow,Object message,String heading,int messagetype);
public static void showInternalMessageDialog(Component mainwindow,Object message,String heading,int messagetype, Icon resim);
public static int showConfirmDialog(Component mainwindow,Object message);
public static int showConfirmDialog(Component mainwindow,Object message,String heading,int messagetype);
public static int showConfirmDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype);
public static int showConfirmDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype,Icon resim);
public static int showInternalConfirmDialog(Component mainwindow,Object message);
public static int showInternalConfirmDialog(Component mainwindow,Object message,String heading,int messagetype);
public static int showInternalConfirmDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype);
public static int showInternalConfirmDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype,Icon resim);
public static int showOptionDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype,Icon resim, Object[] listselectionvaluesObject listedekiilkdeğer);
public static int showInternalOptionDialog(Component mainwindow,Object message,String heading,int opsiyontürü,int messagetype,Icon resim, Object[] listselectionvaluesObject listedekiilkdeğer);

messagetype variable can take the following values

JOptionPane.ERROR_MESSAGE
JOptionPane.INFORMATION_MESSAGE
JOptionPane.PLAIN_MESSAGE
JOptionPane.QUESTION_MESSAGE
JOptionPane.WARNING_MESSAGE

Option type variable can take following constants

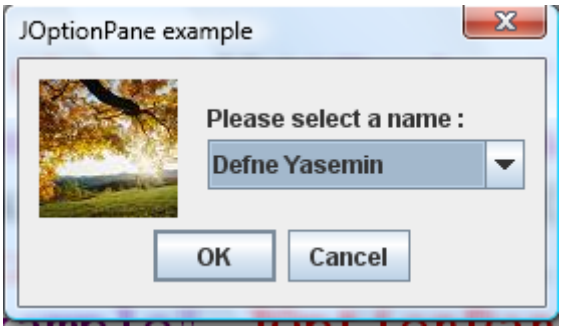
JOptionPane.DEFAULT_OPTION : Gives an ok button
JOptionPane.OK_CANCEL_OPTION : gives ok or cancel button
JOptionPane.YES_NO_CANCEL_OPTION : Yes , No or Cancel buttons
JOptionPane.YES_NO_OPTION : Ye sor No buttons

When OptionDialog is used, following values can be used.

JOptionPane.CANCEL_OPTION
JOptionPane.CLOSED_OPTION
JOptionPane.NO_OPTION
JOptionPane.OK_OPTION
JOptionPane.YES_OPTION
JOptionPane.UNINITILIZED_VALUE

12.8.4 JOptionPane1 Test example

```
import javax.swing.*;
import java.awt.*;

public class JOptionPane1
{
   public static void main(String[] args)
   {
   Object[] o=new Object[]{"Turhan","Meral","İrfan","Hatice","Osman",
   "Nurhan","Birsen","Defne Yasemin","Selin","Mustafa","Ahmet","Mehmet","Ayhan"};
   ImageIcon picture=new ImageIcon("fall.jpg");
   String name=(String)JOptionPane.showInputDialog(null,"Please select a name : ",
   "JOptionPane example",JOptionPane.QUESTION_MESSAGE,picture,o,"Turhan");
   String s1="selected name="+name;
   String s2="JOptionPane example";
   JOptionPane.showMessageDialog(null,s1,s2,JOptionPane.PLAIN_MESSAGE);
   }
```

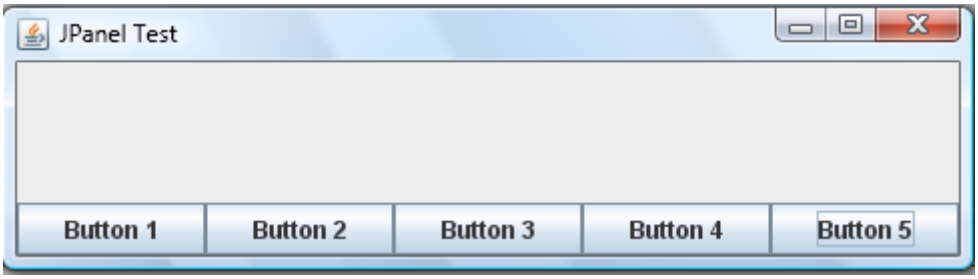


Using JPanel class for formatting

Another class that is frequently used is JPanel. JPanel can be used for formatting purposes, because any number of them can be placed inside each other to control the shape of the screen.

12.8.5 JPane1test example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class JPaneltest extends JApplet
{
  private JPanel buttonPanel;
  private JButton buttons[];

  public void init()
  {
  buttonPanel=new JPanel();
  buttons=new JButton[5];
  buttonPanel.setLayout(new GridLayout(1,buttons.length));
  for(int i=0;i<buttons.length;i++)
  {
    buttons[i]=new JButton("Button "+(i+1));
    buttonPanel.add(buttons[i]);
  }
  setLayout(new BorderLayout());
  add(buttonPanel,BorderLayout.SOUTH);
  }
    public static void main(String s[])
    {
        JFrame f = new JFrame("JPanel Test");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new JPaneltest();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.pack();
        f.setSize(new Dimension(545,400));
        f.setVisible(true);
    }
}
```

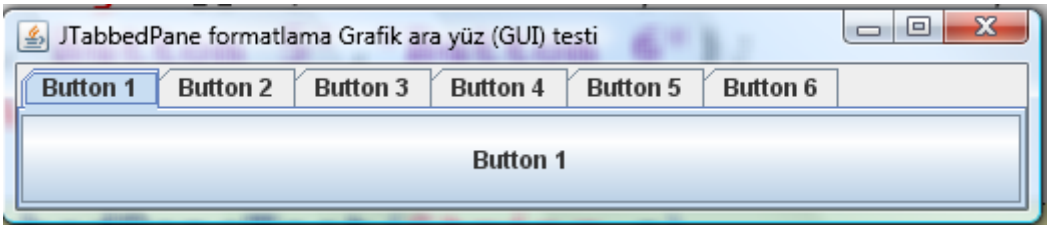


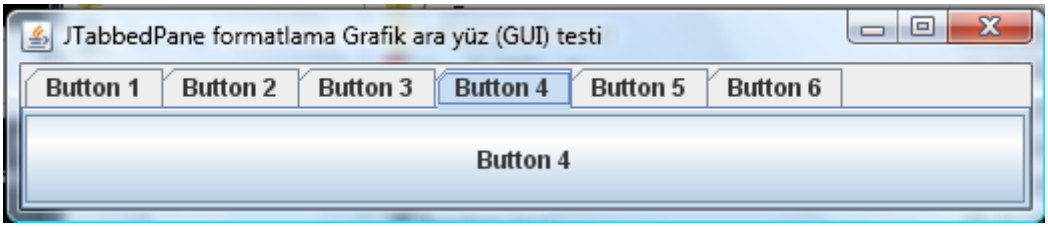
Using JTabbedPane class for formatting

In the JTabbedPane class graphic screen will be divide up as several pages. Different pages can be accessed through the page panes.

12.8.6 JButtonB example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
 public class JTabbedPaneTest extends JFrame implements ActionListener
 {
 private JButton B[];
 private String S[]={"Button 1","Button 2","Button 3",
 "Button 4","Button 5","Button 6"};
 private Container c;

 public JTabbedPaneTest(String s)
 {
 super(s);
 setLayout(new BorderLayout(5,5));
 B=new JButton[6];
 JTabbedPane jtp=new JTabbedPane();
 for(int i=0;i<B.length;i++)
 {
 B[i]=new JButton(S[i]);
 B[i].addActionListener(this);
 jtp.addTab(S[i],B[i]);
 }
 add(jtp);
 }

 public void actionPerformed(ActionEvent e)
 {
  for(int i=0;i<B.length;i++)
 {
 if(e.getSource()==B[i])
 B[i].setText(S[i]+" Buttonye basıldı");
 else B[i].setText(S[i]);
 }
 validate();
 }
     public static void main(String[] args)
     {

       JTabbedPaneTest f = new JTabbedPaneTest("JTabbedPane formatlama Grafik ara yüz (GUI) testi");
       f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );       f.setSize(800,500);
       f.setVisible(true);
     }
 }
```





Using JScrollPane class for formatting

JScrollPane class is looked at previously in JList formatting therefore we will just give a further example here. In the JCheckbox example, graphic elements are added to the page by using JScrollPane, through

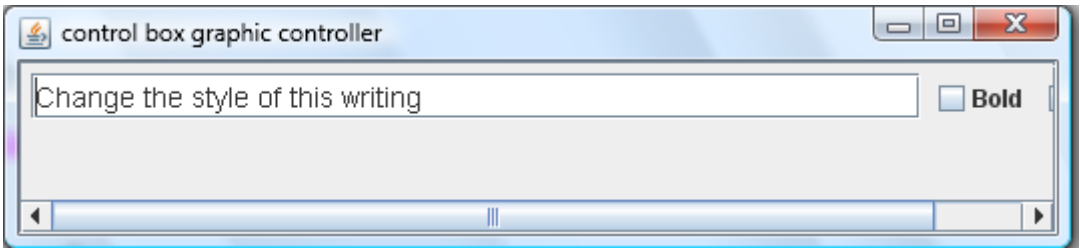
add(new JScrollPane(inputPanel));

In the example program, JSCrolPane will be added to a previously used example

12.8.7 JScrollPaneTest example

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class  JScrollPaneTest extends JApplet implements ItemListener
{
  JPanel inputPanel;
  JTextField t;
  JCheckBox bold,italic;
  int normal_font,bold_font,italic_font;
  String s;
  public void init()
  {
            s="Change the style of this writing";
    inputPanel=new JPanel();
    t=new JTextField(s,40);
    italic_font=Font.PLAIN;
            bold_font=Font.PLAIN;
            t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
    bold=new JCheckBox("Bold");
    italic=new JCheckBox("Italic");
    inputPanel.add(t);
    inputPanel.add(bold);
    inputPanel.add(italic);
    add(new JScrollPane(inputPanel));
    italic.addItemListener(this);
    bold.addItemListener(this);
  }

public void itemStateChanged(ItemEvent e)
{   if(e.getSource()==bold)
```

```
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {bold_font=Font.BOLD;}
      else
        {bold_font=Font.PLAIN;}
    }
    else if(e.getSource()==italic)
    { if(e.getStateChange()==ItemEvent.SELECTED)
        {italic_font=Font.ITALIC;}
      else
        {italic_font=Font.PLAIN;}
    }
    t.setFont(new Font("TimesRoman",bold_font+italic_font,14));
    repaint();
}


    public static void main(String s[])
    {
        JFrame f = new JFrame("control box graphic controller");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new JScrollPaneTest();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.setSize(800,500);
        f.setVisible(true);
    }
}
```



WEEK 12 CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) Following programs draws a star in the window by using Mouse clik and drag. Number of Sides and color of the star can be changed through menu controls.

```
public class polar
{
//polar coordinate system

protected double R,teta;

public polar()
{
R=0;
teta=0;
}

public polar(double Ri,double tetai)
{
R=Ri;
```

```
teta=tetai;
}

public polar(polar y)
{
R=y.R();
teta=y.teta();
}

public void inputR(double Ri)
{
R=Ri;
}

public void inputTeta(double tetai)
{
teta=tetai;
}


public void inputPolar(double Ri,double tetai)
{
R=Ri;
teta=tetai;
}

public void inputCartesian(double xi,double xj)
{
R=Math.sqrt(xi*xi+xj*xj);
teta=Math.atan2(xj,xi);
}

public double R()
{
return R;
}

public double teta()
{
return teta;
}

public double xi()
{
return R*Math.cos(teta);
}

public double xj()
{
return R*Math.sin(teta);
}

public void add(polar y)
{
 kartesienGir((xi()+y.xi()),(xj()+y.xj()));
}
```

```
public void substract(polar y)
{
  kartesienGir((xi()-y.xi()),(xj()-y.xj()));
}

public polar read_polar()
{
return this;
}

public boolean equals(polar v)
{
boolean b=((R==v.R())&&(teta==v.teta()));
return b;
}

public boolean greater_than(polar v)
{
  return (this.R>v.R());
}

public boolean smaller_than(polar v)
{
  return (this.R<v.R());
}

public String toString()
{
 return ""+R+"*exp("+teta+"i) ";
}
}
```

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  starP extends JPanel implements MouseListener,MouseMotionListener
{
  int x1,y1,x2,y2;
  int starRadius;
  Color c;
  int side;
  boolean first;
  GeneralPath x;

 public starP(Color c1,int n)
 {super();
  side=n;
  c=c1;
```

```
 x1=100;y1=100;x2=x1;y2=y1+30;
 first=true;
 addMouseListener(this);
 addMouseMotionListener(this);
}

public void setColor(Color c1)
{c=c1;}

public void setside(int n)
{side=n;}

public void setCizgi(int x1i,int y1i,int x2i,int y2i)
{x1=x1i;y1=y1i;x2=x2i;y2=y2i;}

public GeneralPath star(int x,int y, int n,int starRadius,double aci)
{// bu yıldız cizime teta=pi/2+aci radyandan baslar
  GeneralPath star1=new GeneralPath();
  double teta=2.0*Math.PI/n;
  double R=starRadius;
  double r=starRadius*0.25;
  polar P1=new polar();
  polar P2=new polar();
  polar P3=new polar();
  for(int i=0;i<n;i++)
  {
    double teta1=teta*i+Math.PI/2.0+aci;
    double teta2=teta/2+teta1;
    P1.inputPolar(R,teta1);
    P2.inputPolar(r,teta2);
    if(i==0) star1.moveTo((x+(int)P1.xi()),(y-(int)P1.xj()));
    else     star1.lineTo((x+(int)P1.xi()),(y-(int)P1.xj()));
    star1.lineTo((x+(int)P2.xi()),(y-(int)P2.xj()));
    double teta3=teta*(i+1)+Math.PI/2.0+aci;
    P3.inputPolar(R,teta3);
    star1.lineTo((x+(int)P3.xi()),(y-(int)P3.xj()));
  }
  star1.closePath();
  return star1;
 }

public void paintComponent(Graphics g)
{
super.paintComponent(g);
Graphics2D g2=(Graphics2D)g;
g2.setFont(new Font("Serif",Font.BOLD,24));
g2.setColor(c);
g2.setStroke(new BasicStroke(2.0f));
starRadius=(int)Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
x=star(x1,y1,side,starRadius,0);
g2.fill(x);
}

//MouseListener (fare dinleyicisi)
public void mouseClicked(MouseEvent e)
{}

public void mousePressed(MouseEvent e)
```

```
 {
   x1=e.getX();y1=e.getY();}

  public void mouseReleased(MouseEvent e)
 { x2=e.getX();y2=e.getY();repaint();}

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }

//MouseMotionListener methods

 public void mouseDragged(MouseEvent e)
 {}
 public void mouseMoved(MouseEvent e)
 {}

}
```

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  star extends JApplet implements ListSelectionListener
{
  starP sekil;
  JMenuBar menuBar;
  JMenu m1,m2;
  JPanel outputPanel;
  JList sekilList;
  JList colorList;
  String sekilNames[]={"three sided star","four sided star","five sided star",
  "six sided star","seven sided star","eight sided star","nine sided star","ten sided star"};
  String colorNames[]={"black","blue","cyan","dark gray","gray",
        "green","light grey","magenta","orange","pink","red","white","yellow"};
  Color colors[]={Color.black,Color.blue,Color.cyan,Color.darkGray,Color.gray,
                  Color.green,Color.lightGray,Color.magenta,Color.orange,Color.pink,
                  Color.red,Color.white,Color.yellow};

  public void init()
  {
              int i=3;
              sekil=new starP(Color.black,i);
              colorList=new JList(colorNames);
              colorList.setVisibleRowCount(4);
              sekilList=new JList(sekilNames);
              sekilList.setVisibleRowCount(4);
```

```
    outputPanel=new JPanel();
    outputPanel.setLayout(new BorderLayout());
    outputPanel.add(sekil,BorderLayout.CENTER);
    menuBar=new JMenuBar();
            setJMenuBar(menuBar);
            m1 = new JMenu("Color List");
    m1.setMnemonic('R');
    m1.getAccessibleContext().setAccessibleDescription("Starların hangi colorte olacağını seçer");
    m1.add(new JScrollPane(colorList));
    menuBar.add(m1);
    m2 = new JMenu("Star List");
    m2.setMnemonic('Y');
    m2.getAccessibleContext().setAccessibleDescription("Starların kaç köşeli olacağını seçer");
    m2.add(new JScrollPane(sekilList));
    menuBar.add(m2);
    add(outputPanel,BorderLayout.CENTER);
    colorList.addListSelectionListener(this);
    sekilList.addListSelectionListener(this);
  }


public void valueChanged( ListSelectionEvent e)
{
if(e.getSource()==colorList)
{
Color r1=colors[colorList.getSelectedIndex()];
sekil.setColor(r1);
}
if(e.getSource()==sekilList)
{sekil.setside(sekilList.getSelectedIndex()+3);}
repaint();
}

    public static void main(String s[])
    {
        JFrame f = new JFrame("Star control program push mouse and drag to draw");
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        JApplet applet = new star();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.pack();
        f.setSize(new Dimension(545,400));
        f.setVisible(true);
    }

}
```

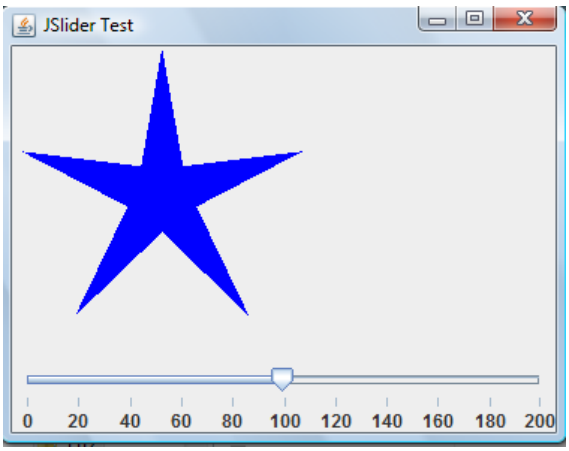
Investigate these programs. The following program is given to draw multi-sided shapes (triangle, rectangle, pentagon, hexagon etc.). By changing star.java to create multiangle.java create a program to draw multisided shaped instead of stars in the star.java program.

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
```

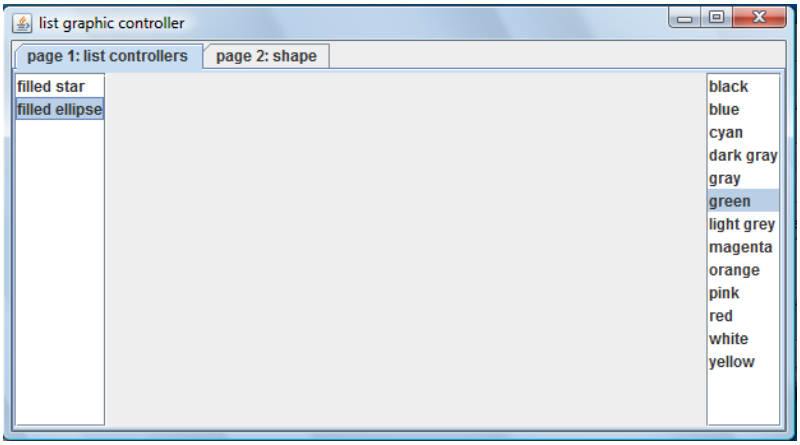
```
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;


public class  multiangleP extends JPanel implements MouseListener,MouseMotionListener
{
  int x1,y1,x2,y2;
  Color c;
  int Corner;
  boolean first;
  GeneralPath x;

 public multiangleP(Color c1,int n)
 {super();
  Corner=n;
  c=c1;
  x1=0;y1=0;x2=0;y2=0;
  first=true;
  addMouseListener(this);
  addMouseMotionListener(this);
 }

 public void setColor(Color c1)
 {c=c1;}

 public void setCorner(int n)
 {Corner=n;}

 public void setCizgi(int x1i,int y1i,int x2i,int y2i)
 {x1=x1i;y1=y1i;x2=x2i;y2=y2i;}

 public GeneralPath multiangle(int x,int y, int n,int multiangleboyu,double aci)
 {
   GeneralPath multiangle=new GeneralPath();
   double teta=2.0*Math.PI/n;
   double R=multiangleboyu;
   polar P1=new polar();
   for(int i=0;i<n;i++)
   {
     double teta1=teta*i+Math.PI/2.0+aci;
     P1.inputPolar(R,teta1);
     if(i==0) multiangle.moveTo((x+(int)P1.xi()),(y-(int)P1.xj()));
     else     multiangle.lineTo((x+(int)P1.xi()),(y-(int)P1.xj()));
   }
   multiangle.closePath();
   return multiangle;
  }

 public void paintComponent(Graphics g)
 {
 super.paintComponent(g);
 Graphics2D g2=(Graphics2D)g;
 g2.setFont(new Font("Serif",Font.BOLD,24));
 g2.setColor(c);
 g2.setStroke(new BasicStroke(2.0f));
```

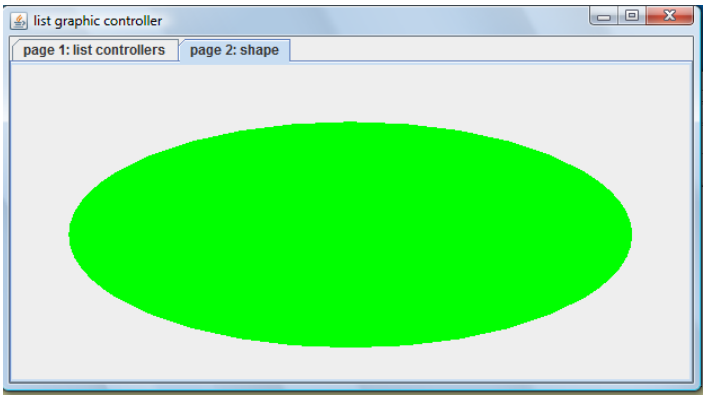
```
 int multiangleboyu=(int)Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
 x=multiangle(x1,y1,Corner,multiangleboyu,0);
 g2.fill(x);
 }

//MouseListener
public void mouseClicked(MouseEvent e)
 {}

 public void mousePressed(MouseEvent e)
 {
   x1=e.getX();y1=e.getY();}

  public void mouseReleased(MouseEvent e)
 { x2=e.getX();y2=e.getY();repaint();}

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener

 public void mouseDragged(MouseEvent e)
 {}
 public void mouseMoved(MouseEvent e)
 {}

}
```

EX2) OvalP.java and SliderTest.java is given above. Investigate these programs. By using starP.java ctreate SliderTest1.java to draw a 5 sided star and change the size of the star through changing the slider position.



EX3) Investigate shapeP and listA1.java. ChangelistA1.java to a new program listA2.java so that page will be divided into two pages by using a JTabbedPane layout formatting. In the first page list controls will be located. In the second page the shape will be located


list graphic controller
page 1: list controllers
page 2: shape
filled star
filled ellipse
black
blue
cyan
dark gray
gray
green
light grey
magenta
orange
pink
red
white
yellow


list graphic controller
page 1: list controllers
page 2: shape

13 EXCEPTION HANDLING (ERROR CONTROL)

13.1 GENERAL DEFINITIONS

Java Language has an error catching mechanism called exception handling. If an error possibility is foreseen fort he program part, exception is trown by the program. Then we chtch the exception and continue to carry out the program, eventhough an error happens. An exception handling block in java is as follows:

```
try { java statements;
}
catch( one_of_the_jav_Exception_types e1)
{ exception control terms }
( one_of_the_jav_Exception_types e2)
{ exception control terms }
finally
{
exception control terms
{ java statements;
}
```

Exception and error control types in java is given belove:

Java.lang error list

AbstractMethodError
ClassCircularityError
ClassFormatError
Error
ExceptionInInitializerError
IllegalAccessError
IncompatibleClassChangeError
InstantiationError
InternalError
LinkageError
NoClassDefFoundError
NoSuchFieldError
NoSuchMethodError
OutOfMemoryError
StackOverflowError
ThreadDeath
UnknownError
UnsatisfiedLinkError
VerifyError
VirtualMachineError

Java.lang Exception list

ArithmeticException
ArrayIndexOutOfBoundsException

ArrayStoreException
ClassCastException
ClassNotFoundException
CloneNotSupportedException
Exception
IllegalAccessException
IllegalArgumentException
IllegalMonitorStateException
IllegalStateException
IllegalThreadStateException
IndexOutOfBoundsException
InstantiationException
InterruptedException
NegativeArraySizeException
NoSuchFieldException
NoSuchMethodException
NullPointerException
NumberFormatException
RuntimeException
SecurityException
StringIndexOutOfBoundsException

Java.util exception list

EmptyStackException
MissingResourceException
NoSuchElementException
TooManyListenersException

Java.io exception list

CharConversionException
EOFException
FileNotFoundException
IOException
InterruptedIOException
InvalidClassException
InvalidObjectException
NotActiveException
NotSerializableException
ObjectStreamException
OptionalDataException
StreamCorruptedException
SyncFailedException
UTFDataFormatException
UnsupportedEncodingException
WriteAbortedException

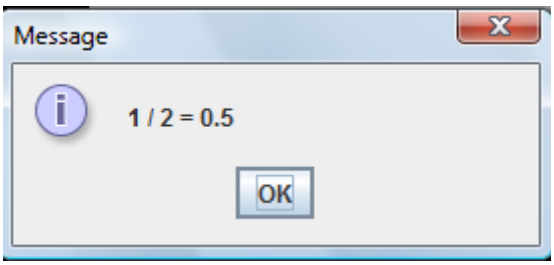
When an error happened in java, it will carried to try-catch group whith the same Exception group given at the description of the method error happend. Exception with the keyword **throws** will also be located after the name and input block of the method. Inside of the method when and where error supposed to be happen keyword throw and a new Exception object is used. In order to explain these structures, few examples will be given in the first example two numbers are divided. If the divisor is zero number goes to infinity. In order to avoid this, an exception can be used. This exception will be created as a class extended from ArithmeticException class

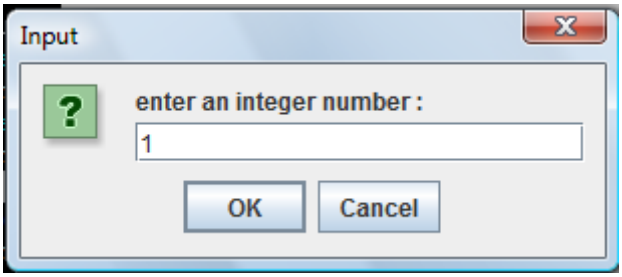
13.1.1 dividebyzeroException class

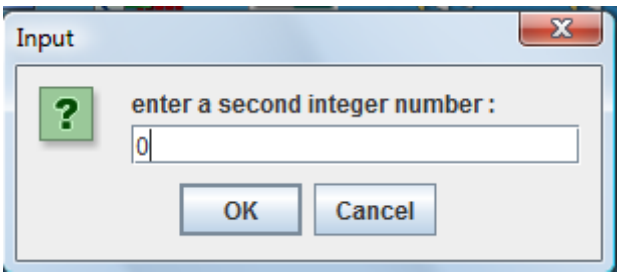
```
public class dividebyzeroException extends ArithmeticException
{     public dividebyzeroException()  { super("number divided by zero "); }
}
```

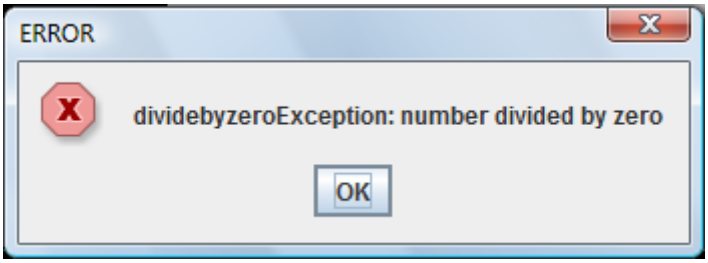
13.1.2 dividebyzeroTest class

```
import javax.swing.*;
import javax.swing.table.*;
import java.text.DecimalFormat;

public class dividebyzeroTest
{
public static double divide(int s1,int s2) throws dividebyzeroException
{
if(s2==0) throw new dividebyzeroException();
return (double) s1/s2;
}

public static void main(String args[])
     {
        String s;
        double result=0;
        int number1,number2;
        number1=Integer.parseInt(
        JOptionPane.showInputDialog("enter an integer number : "));
        number2=Integer.parseInt(
        JOptionPane.showInputDialog("enter a second integer number : "));
        try{
        result=divide(number1,number2);
        JOptionPane.showMessageDialog(null,number1+" / "+number2+" = "+result);
          }  catch(dividebyzeroException e1)
{JOptionPane.showMessageDialog(null,e1.toString(),"ERROR",JOptionPane.ERROR_MESSAGE);}
   }
}
```







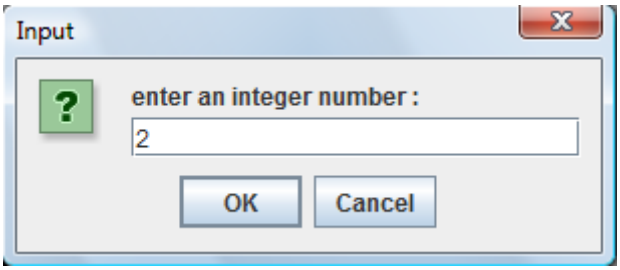


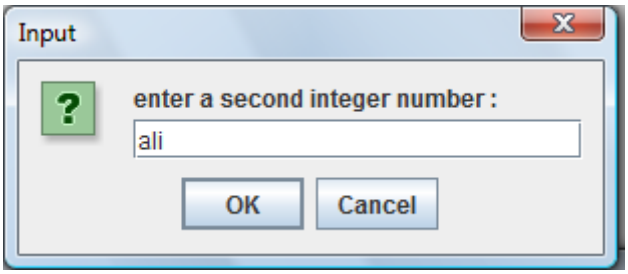
In the second Example two Exception will be used in series. The first one will be the dividebyzeroException defined above, the second one will be NumberFormatException. Two Number required in the program belove are integers, in case any any number or string entered, A Number format exception will be thrown. In this program, while loop is also used loop continiues until correct input statements are given.

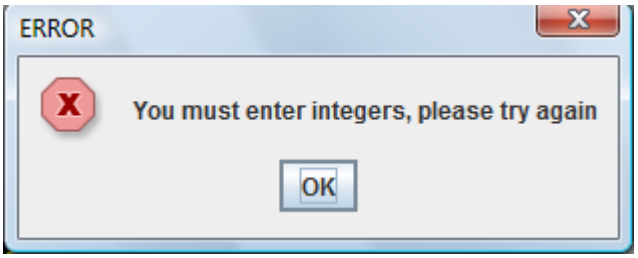
13.1.3 dividebyzeroTest1 class

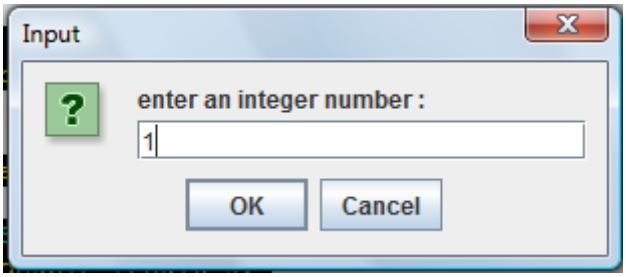
```
Import javax.swing.* ;
import java.text.DecimalFormat;

public class dividebyzeroTest1
{
public static double divide(int s1,int s2) throws dividebyzeroException
{
if(s2==0) throw new dividebyzeroException();
return (double) s1/s2;
}

public static void main(String args[])
    {
      String s;
      double result=0;
      int number1,number2;
      boolean continueloop=true;
      while(continueloop)
      {
      try{
            number1=Integer.parseInt(
```

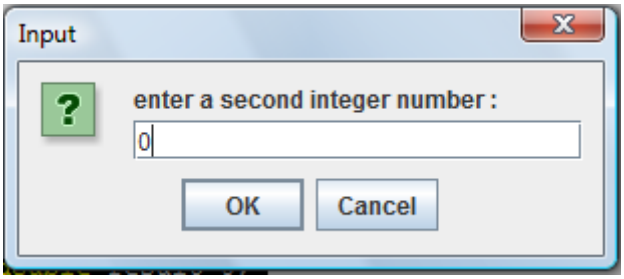
```
        JOptionPane.showInputDialog("enter an integer number : "));
        number2=Integer.parseInt(
        JOptionPane.showInputDialog("enter a second integer number : "));
        result=divide(number1,number2);
        continueloop=false;
        JOptionPane.showMessageDialog(null,number1+" / "+number2+" = "+result);
          }
       catch(NumberFormatException e2)
       {JOptionPane.showMessageDialog(null,"You must enter integers, please try
again","ERROR",JOptionPane.ERROR_MESSAGE);}
       catch(dividebyzeroException e1)
       {JOptionPane.showMessageDialog(null,e1.toString(),"ERROR",JOptionPane.ERROR_MESSAGE);}
       }
    }
}
```

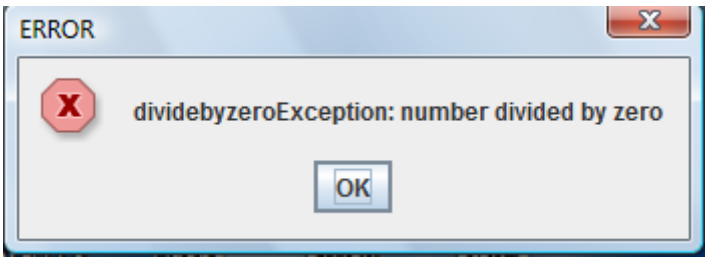


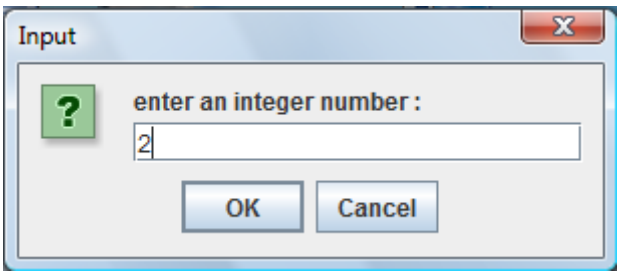


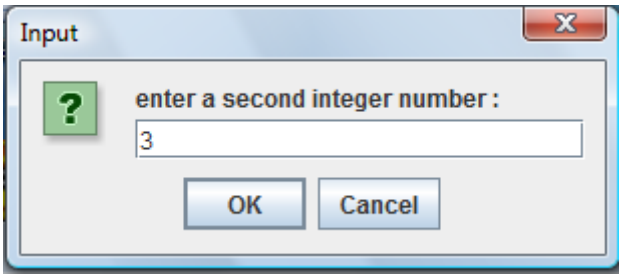


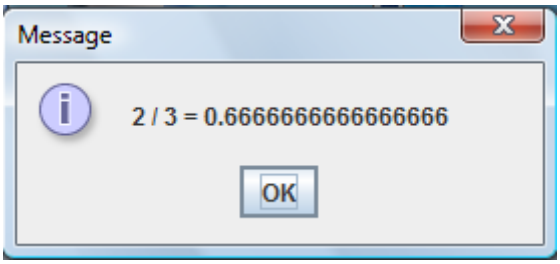












In the third example a class previously used to explain class structure, box class will be used with zero_or_negative_sizeException that will be created from ArithmeticException.

13.1.4 zero_or_negative_sizeException class

```
public class zero_or_negative_sizeException extends ArithmeticException
{     public zero_or_negative_sizeException()  { super("the given dimension is zero or negative "); }
}
```

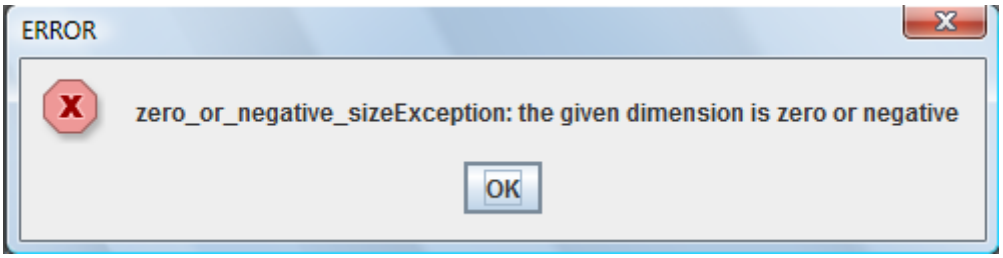
13.1.5 box class with zero_or_negative_sizeException

```
class box
{
private double length;
private double width;
private double height;

box(double l,double w, double h) throws zero_or_negative_sizeException
{ if(l<=0 || w<=0 || h<=0) throw new zero_or_negative_sizeException();
  else
  {
  length=l;
  width=w;
  height=h;
  }
}

box(box bi) throws zero_or_negative_sizeException
{if(bi.length<=0 || bi.width<=0 || bi.height<=0) throw new zero_or_negative_sizeException();
  else
  {
  length=bi.length;
  width=bi.width;
  height=bi.height;
  }
}

// This methods are written to access private length,
// width and height variables
public double read_length() {return length;}
public double read_width() {return width;}
public double read_height() {return height;}
//This methods are written to change private length,
// width and height variables
public void write_length(double l) throws zero_or_negative_sizeException
{if(l<=0 ) throw new zero_or_negative_sizeException();
length=l;
}
public void write_width(double w) throws zero_or_negative_sizeException
{if(w<=0 ) throw new zero_or_negative_sizeException();
width=w;
}
public void write_height(double h) throws zero_or_negative_sizeException
{if(h<=0 ) throw new zero_or_negative_sizeException();
height=h;
}

double volume()
{return length*width*height;}
```

```
public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n";
  return s;
}
}
```

13.1.6 boxtest class with zero_or_negative_sizeException

```
import javax.swing.JOptionPane;

class boxtest
{
   public static void main(String args[])
   {
   double volume;
   try{
      box firstbox= new box(1.0,1.2,0.2);
      String s="First box : \n"+firstbox.toString()+"\n";
      firstbox.write_length(3.0);
      firstbox.write_width(2.0);
      firstbox.write_height(0.0);
      s+="value changed first box : \n"+firstbox.toString()+"\n";
      JOptionPane.showMessageDialog(null,s,
      "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
      }  catch(zero_or_negative_sizeException e1)
   {
JOptionPane.showMessageDialog(null,e1.toString(),"ERROR",JOptionPane.ERROR_MESSAGE);}
     System.exit(0);
   }
}
```



The last example will be classes circle and cylinder, **zero_or_negative_sizeException** will also be used with these classes.

13.1.7 point class

```
public class point
{   protected double x, y;

    public point(double a, double b)
    {enterpoint(a,b);}

    public point(point A)
    {enterpoint(A.readX(),A.readY());}

    public void enterpoint(double a, double b)
    { x=a;y=b;}

    public double readX()
    {return x;}

    public double readY()
    {return y;}

    public String toString()
    {return "["+x+","+y+"]";}

    public String name(){return "point";}
}
```

13.1.7 circle class with zero_or_negative_sizeException

```
public class circle extends point
{
    protected double radius;

    public circle(double r, double a, double b) throws zero_or_negative_sizeException
    {  super(a,b);
       enterradius(r);
    }

    public circle(double r, point n) throws zero_or_negative_sizeException
    {  super(n.readX(),n.readY());
       enterradius(r);
    }

    public circle(circle c) throws zero_or_negative_sizeException
    {  super(c.x,c.y);
       enterradius(c.radius);
    }

    public void enterradius(double r) throws zero_or_negative_sizeException
    {   if(r > 0.0)
            radius=r;
```

```
        else
            throw new zero_or_negative_sizeException();
    }

    public double readradius()
    {return radius;}

    public double area()
    {return Math.PI*radius*radius;}

    public String toString()
    { return "Center = "+"["+x+","+y+"]"+"; radius="+radius;}

    public String name(){return "circle";}
}
```

13.1.8 point class with zero_or_negative_sizeException

```
public class cylinder extends circle
 {
   protected double height;

 public cylinder( double h,circle d) throws zero_or_negative_sizeException
 { super(d);
   height=h;
 }

 public cylinder( double h, double r, double a, double b) throws zero_or_negative_sizeException
 { super(r,a,b);
   height=h;
 }

 public void enterheight (double h ) throws zero_or_negative_sizeException
 { if(h>0) height=h;
   else
     throw new zero_or_negative_sizeException();
 }

 public double readheight(){return height;}

 public double area()
 {return 2*super.area()+2*Math.PI*radius*height;}

 public double volume(){return super.area()*height;}

 public String toString()
 {return super.toString()+";height = "+height;}

 public String name(){return "cylinder";}
}
```

```
import javax.swing.*;

class inheritancetest1
```

```
{
  public static void main(String args[])
  {
  try{
  point n=new point(1.0,2.0);
  circle d1=new circle(3.0,n);
  circle d2=new circle(0.5,2.0,1.0);
  cylinder s1=new cylinder(5.0,d1);
  cylinder s2=new cylinder(5.0,0.0,1.0,1.0);
  String s=n.toString()+"\n "+
  d1.name()+" "+d1.toString()+"\n "+
  d2.name()+" "+d2.toString()+"\n "+
  s1.name()+" "+s1.toString()+"\n "+
  s2.name()+" "+s2.toString()+"\n ";
  JOptionPane.showMessageDialog(null,s,
    "classes: point circle cylinder protected inheritance test",JOptionPane.PLAIN_MESSAGE);
  }  catch(zero_or_negative_sizeException e1)
  {  JOptionPane.showMessageDialog(null,e1.toString(),"ERROR",JOptionPane.ERROR_MESSAGE);}
  System.exit(0);}
}
```

The last example will be rational numbers. If the denominator of the rational number is zero, a division exception will occurs.

13.1.9 rational number class with dividebyzeroException

```
//=========================================
// rational  number description
// Dr. Turhan Coban
//=========================================
import javax.swing.JOptionPane; // giriş çıkış


// class rational numbers


public class rational {
   // class variables
   protected int   numerator;
   protected int   denominator;

   // constructor methods

public rational()
{
numerator=0;
denominator=1;
}

public rational(int nnumerator,int ndenominator) throws dividebyzeroException
{
numerator=nnumerator;
if(ndenominator==0) throw new dividebyzeroException();
denominator=ndenominator;
}

public rational(int numer)
{
```

```
numerator=numer;
denominator=1;
}

public rational(rational c ) throws dividebyzeroException
{
numerator=c.numerator();
if(c.denominator()==0) throw new dividebyzeroException();
denominator=c.denominator();
}

//calculation methods

public int leastcommondivisor()
{
// least common divisor
int n=numerator;
int m=denominator;
if(n==0)
   return m;
if(m==0)
   return n;
while(m != n)
{
  if(n>m)
     n=n-m;
  else
     m=m-n;
}
return n;
}

public void simplify()
{
//simplyfy the rational number by using least common divisor
int isaret=1;
if(numerator<0)
{
 isaret=-isaret;
 numerator=-numerator;
}
if(denominator<0)
{
 isaret=-isaret;
 denominator=-denominator;
}
if(denominator==0) throw new dividebyzeroException();
int ebob=leastcommondivisor();
ebob=Math.abs(ebob);
numerator=isaret*numerator/ebob;
denominator=denominator/ebob;
}


// entry and exit methods

public int numerator()
{
```

```
return numerator;
}

public int denominator()
{
return denominator;
}

public void input_numerator(int r)
{
numerator=r;
}

public void input_denaminator(int i) throws dividebyzeroException
{
if(i==0) throw new dividebyzeroException();
denominator=i;
}

public void input_rational(rational sag)
{
numerator=sag.numerator();
denominator=sag.denominator();
}

public void input_rational(int nr,int ni)
{
numerator=nr;
denominator=ni;
}

public void input_rational(int nr)
{
numerator=nr;
denominator=1;
}

public void input_rational(double d)
{
// load rational equavalent of a real number
// accuracy of the real number is lost in the process
input_rational((int)d*10000,10000);
simplify();
}

public double toDouble()
{
//double number equivalent of the rational numbers
return ((double)numerator/(double)denominator);
}

public static rational convert_to_rational(double d)
{
// load rational equavalent of a real number
// accuracy of the real number is lost in the process
rational b=new rational();
b.input_rational((int)d*10000,10000);
b.simplify();
```

```
return b;
}


public void add(rational  sag)
{
//addition of rational numbers
numerator = numerator*sag.denominator() + sag.numerator()*denominator;
denominator = denominator*sag.denominator();
simplify();
}

public void add(int sag)
{
//addition of rational numbers
numerator = numerator + sag*denominator;
simplify();
}

public void substract(rational  sag)
{
//substraction  of rational numbers
numerator = numerator*sag.denominator() - sag.numerator()*denominator;
denominator = denominator*sag.denominator();
simplify();
}

public void substract(int sag)
{
numerator = numerator - sag*denominator;
simplify();
}

public void multiply(rational sag )
{
numerator = numerator*sag.numerator();
denominator = denominator*sag.denominator();
simplify();
}

public void multiply(int sag )
{
numerator = numerator*sag;
simplify();
}

public void divide(rational sag )
{
numerator = numerator*sag.denominator();
denominator = denominator*sag.numerator();
simplify();
}

public void divide(int sag )
{
denominator = denominator*sag;
simplify();
}
```

```
public static rational add(rational sol, rational sag)
{
 int r1= sol.numerator() * sag.denominator() + sag.numerator() * sol.denominator();
 int i1= sol.denominator()* sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational add(rational sol, int sag)
{
 int r1=sol.numerator() + sag*sol.denominator();
 int i1=sol.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational add(int sol, rational sag)
{
 int r1=sag.numerator() + sol*sag.denominator();
 int i1=sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

//===================================
public static rational substract(rational sol, rational sag)
{
 // iki rational sayının addmını return deyimiyle rational olarak
 // aktarır
 int r1= sol.numerator() * sag.denominator() - sag.numerator() * sol.denominator();
 int i1= sol.denominator()* sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational substract(rational sol, int sag)
{
 // bir rational ve bir numerator sayının addmını return deyimiyle rational olarak
 // aktarır
 int r1=sol.numerator() - sag*sol.denominator();
 int i1=sol.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

public static rational substract(int sol, rational sag)
{
```

```
 int r1=sag.numerator() - sol*sag.denominator();
 int i1=sag.denominator();
 rational sonuc;
 sonuc=new rational(r1,i1);
 sonuc.simplify();
 return sonuc;
}

//===================================

public static rational multiply(rational sol, rational sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator()*sag.numerator(),sol.denominator()*sag.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational multiply(rational sol, int sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator()*sag,sol.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational multiply(int sol, rational sag)
{
 rational sonuc;
 sonuc=new rational(sag.numerator()*sol,sag.denominator());
 sonuc.simplify();
 return sonuc;
}

public static rational divide(rational sol, rational  sag)
{
 double a=sag.numerator()*sag.numerator()+sag.denominator()*sag.denominator();
 rational sonuc;
 sonuc=new rational((int)(sol.numerator()*sag.denominator()),(int)(sol.denominator()*sag.numerator()));
 sonuc.simplify();
 return sonuc;
}

public static rational divide(rational sol, int  sag)
{
 rational sonuc;
 sonuc=new rational(sol.numerator(),(sol.denominator()*sag));
 sonuc.simplify();
 return sonuc;
}

public boolean smaller_than(rational sol,rational sag)
{
// less then comparison of two rational numbers
return (sol.toDouble() < sag.toDouble());
}

public boolean smaller_than_equals(rational sol,rational sag)
```

```
{
// less then and equals comparison of two rational numbers
return (sol.toDouble() <= sag.toDouble());
}

public boolean greater_than(rational sol,rational sag)
{
// greater_than then comparison of two rational numbers
return sol.toDouble() > sag.toDouble();
}

public boolean greater_than_equals(rational sol,rational  sag)
{
// greater_than then and equals comparison of two rational numbers
return sol.toDouble() >= sag.toDouble();
}


public boolean equals(rational sol,rational sag)
{
// equals comparison of two rational numbers
return sol.toDouble() == sag.toDouble();
}

public boolean not_equals(rational sol,rational  sag)
{
// not equals comparison of two rational numbers
return sol.toDouble() != sag.toDouble();
}

public static String toString(rational  value)
{
String b="";
if(Math.abs(value.denominator())!=1)
  {
    b=b+"( "+value.numerator()+" / "+value.denominator()+" )";
  }
else
  {
    b=b+value.numerator()+" ";
  }
return b;
}

public String toString()
{
// yazima hazır rational formda String deşiskenini iletir.
String b="";
if(Math.abs(denominator())!=1)
  {
    b=b+"( "+numerator+" / "+denominator()+" )";
  }
else
  {
    b=b+numerator()+" ";
  }
return b;
}
```

```
            };

//end of class rational
```

13.1.10 rationaltest class with dividebyzeroException

```
import javax.swing.JOptionPane;
public class rationaltest
{
public static void main(String arg[])
{
try{
rational r1=new rational(1,3);
rational r2=new rational(1,0);
r2.add(r1);
 JOptionPane.showMessageDialog(null,r2.toString(),
    "rational number class",JOptionPane.PLAIN_MESSAGE);
    }  catch(dividebyzeroException e1)
{JOptionPane.showMessageDialog(null,e1.toString(),"ERROR",JOptionPane.ERROR_MESSAGE);}    }
}
```

WEEK 1(13) CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) Class company is given as :

```
public class company
{ public String name;
  public double capital,balance,profit;
  public int process;
  //this method establish the company

  //constructor method
  public company(String is,double Ri)
  {name=is;
  capital=Ri;
  balance=Ri;
  profit=0;
  process=0;
  }

  public void  buy(double x)
  {balance-=x;
   process++;
   if(balance<=0)
     {System.out.println("company is broken "+process);}
  }

  public void  sell(double x)
  {balance+=x;
   profit=balance-capital;
   process++;
  }

  public String acount()
```

```
  { String s="------------"+name+"--------------------\n";
  if(balance<=0)
     s+="company is broken \n";
  else
     s+=" capital = "+capital+" TL\n";
     s+=" balance = "+balance+" TL\n";
     s+=" profit = "+profit+" TL\n";
     s+=" number of processes = "+process+" \n";
  return s;
  }
}
```

A class **companyisbrokenException** is defined as :

```
public class companyisbrokenException extends Exception
{     public companyisbrokenException()  { super("the company is broken"); }
}
```

Change the class company to throw a **companyisbrokenException** if company balance is less than or equal to zero. Write a test program to try this class. The test program previous to Exception case was:

```
import javax.swing.*;



public class companytest
{  public static void main(String args[])
   {
   company T=new company("Defne Holding",1.0e6);
   for(int i=0;i<100;i++) {T.buy(100);T.sell(400);}
   String s=T.acount();

   company A=new company("Ali Limited",1000.0);
   for(int i=0;i<100;i++) {A.buy(100);A.buy(90);}
   s+=A.acount();

   JOptionPane.showMessageDialog(null,s, "company class test",JOptionPane.PLAIN_MESSAGE);
   }
}
```

EX2 Following programs are given

```
public class minuslogarithmException extends ArithmeticException
{     public minuslogarithmException()  { super("logarithm of a minus number "); }
}
```

```
import javax.swing.*; // program poweres class Scanner

public class logarithmTest
{
    public static double log(double x) throws minuslogarithmException
```

```
    {
      // ln(x) = 1+y^2/3+y^4/5+y^6/7+...   y=(x-1)/(x+1)
      if(x<0) throw new minuslogarithmException();
      double power=1;
      double ln=1;
      double y=(x-1)/(x+1);
      double n=1;
      do
      {
      power*=y*y;
      ln+=power /(2.0*n+1.0);
      n++;
      } while(n<=100000);
      ln*=2*y;
              return ln;
    }
            // main method begins execution of Java application
    public static void main( String args[] )
    {

      String s;
      double x=Double.parseDouble(JOptionPane.showInputDialog("Enter a real number: ")); // read number
      s="log("+x+") = "+log(x)+"   "+"Math library ="+Math.log(x);
      JOptionPane.showMessageDialog(null,s,"natural logarithm calculation",JOptionPane.PLAIN_MESSAGE);
    } // end method main
} // end class
```

When we run the program and enter a negative number as input following error happens and program stops

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" logarithmTest
Exception in thread "main" minuslogarithmException: logarithm of a minus number
            at logarithmTest.log(logarithmTest.java:8)
            at logarithmTest.main(logarithmTest.java:28)
> Terminated with exit code 1.
```

Write a try-catch block in the main program to prevent it. Write the program code to the computer yourself. It is not provided.

HOMEWORK 1(13)

HW1) In the rational number example, **dividebyzeroException** is used. Add up ArithmeticException so that if a different kind of input than integer is given in constructors and other methods it will throw an exception. Write an example program to try this exeption.

HW2) Following class complex is given

```
public class complex
{ public double real;
  public double imaginary;

  public complex(double g,double s)
  {real=g;
   imaginary=s;
  }
```

```
  public complex(complex s1)
  {real=s1.real;
   imaginary=s1.imaginary;
  }

  public void add(complex s1)
  {real+=s1.real;
   imaginary+=s1.imaginary;
  }

  public  static complex add(complex s1,complex s2)
  {complex z3=new complex((s1.real+s2.real),(s1.imaginary+s2.imaginary));
   return z3;
  }

  public String toString()
  { String s=+real+" + i*"+imaginary+" \n";
    return s;
  }
}
```

In a main program two double numbers are read and a complex number is created. At input if given number is not a real number program will throw a NumberFormatException and write the format error. Write the main program.

14 ADVANCED DATA STRUCTURES

14.1 GENERIC METHODS

In the classes and methods so far the type of the input and output variables are defined . Sometimes the code will be shorter if the types of the variable is defined as generic. In the class arrayprint 4 different data type is defined and an overloaded individual method is written for each data type, then the same name can be used to printout each data type.

14.1.1 arrayprint class to print 4 different data type by overloaded printArray methods

```
public class arrayprint
{
public static void printArray(Integer[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}
public static void printArray(Double[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}
public static void printArray(Character[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}
public static void printArray(String[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}
public static void main(String arg[])
{
Integer[] i={1,2,3,4,5,6};
Double[] d={1.1,2.2,3.3,4.4,5.5,6.6};
Character[] c={'a','b','c','d','f','g'};
String[] s={"ali","veli","49","elli"};
printArray(i);
printArray(d);
printArray(c);
printArray(s);
}

}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" arrayprint1
 1 2 3 4 5 6
 1.1 2.2 3.3 4.4 5.5 6.6
 a b c d f g
 ali veli 49 elli
```

```
> Terminated with exit code 0.
```

To make program code shorter array terms

14.1.2 arrayprint1 class to print 4 different data type by genereic printArray method

```
public class arrayprint1
{
public static<E> void printArray( E[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}

public static void main(String arg[])
{
Integer[] i={1,2,3,4,5,6};
Double[] d={1.1,2.2,3.3,4.4,5.5,6.6};
Character[] c={'a','b','c','d','f','g'};
String[] s={"ali","veli","49","elli"};
printArray(i);
printArray(d);
printArray(c);
printArray(s);
}
}
```

The result will be exactly the same. In this second example a general class definition, called generic class, is used. In order to do this class definition is replaced with generic definition **<E>.** The same of general usage could be achived at least for printing function through the class Object. Because all the classes extended from class Object.

```
public class arrayprint2
{
public static void printArray( Object[] x)
{for(int i=0;i<x.length;i++)
  System.out.print(" "+x[i]);
 System.out.println();
}

public static void main(String arg[])
{
Integer[] i={1,2,3,4,5,6};
Double[] d={1.1,2.2,3.3,4.4,5.5,6.6};
Character[] c={'a','b','c','d','f','g'};
String[] s={"ali","veli","49","elli"};
printArray(i);
printArray(d);
printArray(c);
printArray(s);
}
}
```

14.2 GENERIC CLASSES

The generic class concept can also be applied to create new classes.

14.2.1 Generic matrix class

```
public class matrix<E>
 {
 public E[][] A;
 //constructor methods
 public matrix(E[][] Ai)
 {   try{
            A = (E[][]) new Object[Ai.length][Ai[0].length];
    }catch(EmptyMatrixException e){}
     catch(FullMatrixException f){};
  input_matrix(Ai);
 }

 public matrix(matrix< E > M)
 {
            try{
            A=(E[][])new Object[M.A.length][M.A[0].length];
      }catch(EmptyMatrixException e){}
     catch(FullMatrixException f){};
            input_matrix(M.A);
 }

 public matrix(int n,int m)
 { try{
            A=(E[][])new Object[n][m];
             }catch(EmptyMatrixException e){}
     catch(FullMatrixException f){};
 }

 public void input_matrix(E[][] Ai)
 {   for(int i=0;i<A.length;i++)
            { for(int j=0;j<A[0].length;j++)
              {A[i][j]=Ai[i][j];}
    }
 }


 public String toString(matrix< E > M)
 { int n=M.A.length;
  int m=M.A[0].length;
  String s="";
 for(int i=0;i<n;i++)
            { for(int j=0;j<m;j++)
              {s+=M.A[i][j]+" ";}
            s+="\n";
            }
 return s;
 }

 public String toString()
 {
  int n=A.length;
  int m=A[0].length;
  String s1="";
  for(int i=0;i<n;i++)
```

```
  {  for(int j=0;j<m;j++)
   { s1+=String.format(" %s ",A[i][j]);}
  s1+="\n";
  }
 return s1;
 } //end of method toString

 }//end of class matrix
```

14.2.2 Generic matrix test class matrixtest

```
public class matrixtest
{public static void main(String arg[])
{Double[][] a={{1.1,2.2},{3.3,4.4}};
 Integer[][] b={{1,2},{3,4}};
 Character[][] c={{'a','b'},{'c','d'}};
 matrix<Double> A=new matrix<Double>(a);
 matrix<Integer> B=new matrix<Integer>(b);
 matrix<Character> C=new matrix<Character>(c);
 System.out.println(A.toString());
 System.out.println(B.toString());
 System.out.println(C.toString());
}
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" matrixtest
 1.1  2.2
 3.3  4.4

 1  2
 3  4

 a  b
 c  d
> Terminated with exit code 0.
```

It should be note that array structure is defined as Object an then transfermed to the generic E[][] . Generic classes will be used in advanced data structures such as lists, trees etc.

14.3 USING LAMBDA EXPRESSIONS WITH GENERIC CLASSES

Using of Lambda expressions are introduced at chapter 9 and referenced again in chapter 11 and 12. IN Java 8 several built in Functional interfaces are defined to be specifically used ias Lambda expressions. Some of the most comon interfaces are listed in the table below.

Table 14.3 Important functional interfaces in Java

Interface name	Argument	Returns
Predicate<T>	T	boolean
Consumer<T>	T	void
Function<T,R>	T	R
BiFunction<T,W,R>	(T,W)	R

Supplier<T>	None	T
UnaryOperator<T>	T	T
BinaryOperator<T>	(T,T)	T
Binary Function<T,W,R>	(T,W)	R

Now, let us see utilization of these interfaces with Lambda expressions with generic classes

14.3.1 Predicate Function with Lambda expression usage

```
import java.util.function.*;
public class  Predicate_L
{
  public static void main(String[] arg)
  {
  Predicate<Integer> atLeast5 = x -> x > 5;
  Boolean b=atLeast5.test(2);
  System.out.println(b);
  }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" Predicate_L
false

> Terminated with exit code 0.
```

14.3.2 Consumer Function with Lambda expression usage

```
import java.util.function.*;
public class  Function_L
{ public static void main(String[] arg)
  { Function<Double,Double> y = x -> {return x*x-2.0*+5;};
    System.out.println(z.apply(2.0));
  }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" Consumer_L
-6.0

> Terminated with exit code 0.
```

14.3.3 Function Function with Lambda expression usage

```
import java.util.function.*;
public class  Function_L
{ public static void main(String[] arg)
  { Function<Double,Double> y = x -> x*x-2.0*x+5;
    System.out.println(y.apply(2.0));
  }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" Function_L
5.0

> Terminated with exit code 0.
```

14.3.4 Function Function with Lambda expression usage

```
import java.util.function.*;
public class  Function_L1
{ public static void main(String[] arg)
  { Function<Double,Boolean> y = x -> (x*x-2.0*x+5)>2;
    System.out.println(y.apply(2.0));
  }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" Function_L1
true

> Terminated with exit code 0.
```

14.3.5 BiFunction Function with Lambda expression usage

```
import java.util.function.*;
public class  BiFunction_L
{ public static void main(String[] arg)
  { BiFunction<Double,Double,Boolean> z = (x,y) -> (x*x+y*y)>3;
    System.out.println(z.apply(2.0,3.0));
  }
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" BiFunction_L
true

> Terminated with exit code 0.
```

14.4 STRINGTOKENIZER CLASS

If a sentence should be divided up to its words, StringTokenizer java library class is ideal fort his task. StringTokenizer class divides A string into a set of String array by using empty space between the words. Methods of the StringTokenizer class:

StringTokenizer(String str) : Constructs a string tokenizer for the specified string.

StringTokenizer(String str, String delim) : Constructs a string tokenizer for the specified string.

StringTokenizer(String str, String delim, boolean returnDelims) :Constructs a string tokenizer for the specified string.

İnt	countTokens() : Calculates the number of times that this tokenizer's nextToken method can be called before it generates an exception.
Boolean	hasMoreElements() : Returns the same value as the hasMoreTokens method.
Boolean	hasMoreTokens() : Tests if there are more tokens available from this tokenizer's string.
Object	nextElement() : Returns the same value as the nextToken method, except that its declared return value is Object rather than String.
String	nextToken() : Returns the next token from this string tokenizer.
String	nextToken(String delim) : Returns the next token in this string tokenizer's string.

14.3.1 StringTokenizerTest class

```
import java.io.*;
import java.util.*;

public class StringTokenizerTest
{
  public static void main(String arg[])
  {
  String s="Once upon a time in a country far far away";
  System.out.println("sentence : "+s);
  StringTokenizer t=new StringTokenizer(s);
  System.out.println("word count : "+t.countTokens());
  int i=0;
  while(t.hasMoreTokens())
  {System.out.println("word index : "+(i++)+" word : "+t.nextToken());}
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" StringTokenizerTest
sentence : Once upon a time in a country far far away
word count : 10
word index : 0 word : Once
word index : 1 word : upon
word index : 2 word : a
word index : 3 word : time
word index : 4 word : in
word index : 5 word : a
word index : 6 word : country
word index : 7 word : far
word index : 8 word : far
word index : 9 word : away

> Terminated with exit code 0.
```

Our first example is an application of this class. String s is loaded into a StringTokenizer class object t. Then by using

```
while(t.hasMoreTokens())
  {System.out.println("word index : "+(i++)+" word : "+t.nextToken());}
  }
```

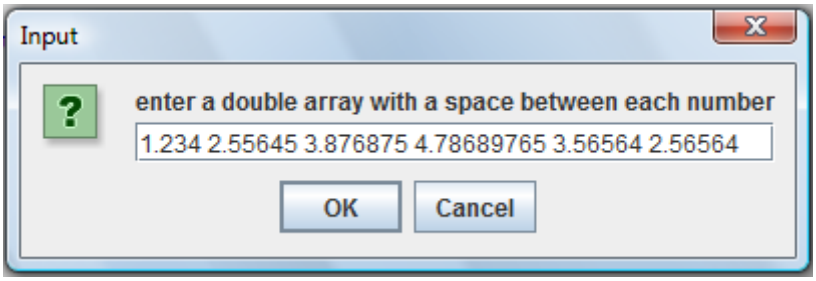
Token divided up into the words.

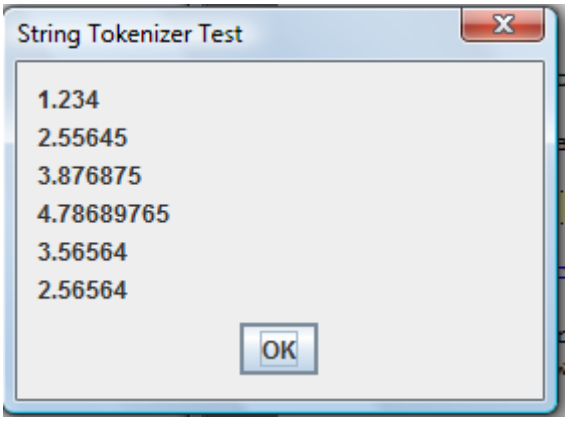
In the second example a double array is read by using JOptionPane class the size of the class is processed by using StringTokenizer class and loaded into array and result is printed by using print method. Otomatic array counting in an array input is one of the most important application of the StringTokenizer class.

14.3.2 StringTokenizerTest1 class

```
import java.io.*;
```

```
import java.util.*;
import javax.swing.*;

public class StringTokenizerTest1
{
    public static double[] enterdoublearray(String s)
    {
    String s1=JOptionPane.showInputDialog(s);
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }

    public static String print(double a[])
    {
    String s="";
    int n=a.length;
    int i=0;
    while(i<n) {s+=a[i]+"\n";}
    return s;
    }

  public static void main(String arg[])
  {
  double x[]=enterdoublearray("enter a double array with a space between each number");
  JOptionPane.showMessageDialog(null,print(x),"String Tokenizer
Test",JOptionPane.PLAIN_MESSAGE);
  }
}
```





Of course you can write your own program to do the same task as well if required

```
import javax.swing.*;
public class problem15C
{
public static int countcharacters(char a,String s)
{
int n=s.length();
int number=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) number++;}
return number;
}
public static String[] word(char a,String s)
{
int nn=countcharacters(a,s);
int n=s.length();
String s1[]=new String[nn+1];
String s2="";
int j=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) {s1[j]=s2;s2="";j++;}
 else            {s2+=s.charAt(i);}
}
s1[nn]=s2;
return s1;
}

public static String[] word(String s)
{
char a=' ';
return word(a,s);
}

public static String toString(String s[])
{int n=s.length;
 String ss="";
```

```
 for(int i=0;i<n;i++)
 {ss+="i = "+i+" "+s[i]+"\n";}
  return ss;
}

public static void main(String arg[])
{
String s="Once upon a time in a country far far away";
System.out.println("number of characters = "+countcharacters(' ',s));
String s1[]=word(s);
System.out.println(toString(s1));
}
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" problem15C
number of characters = 9
i = 0 Once
i = 1 upon
i = 2 a
i = 3 time
i = 4 in
i = 5 a
i = 6 country
i = 7 far
i = 8 far
i = 9 away

> Terminated with exit code 0.
```

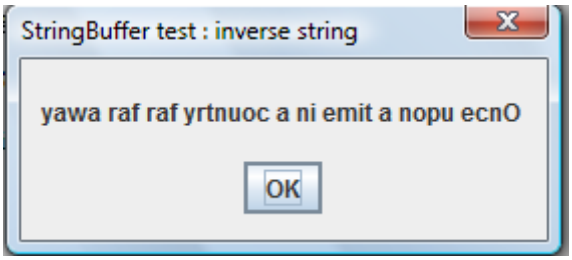
14.4 STRINGBUFFER CLASS

StringBuffer class is sister class of String. It will dynamically add parts to the strings or change the strings. Two examples are given in here in the first example, String is reversed by using StringBuffer class. In the seccond example a word is added up into the string.

14.4.1 StringBuffer class test

```
import java.io.*;
import java.util.*;
import javax.swing.*;

public class StringBufferTest
{
        public static String reverseIt(String source) {
            int i, len = source.length();
            StringBuffer dest = new StringBuffer(len);
            for (i = (len - 1); i >= 0; i--) {
                dest.append(source.charAt(i));
            }
            return dest.toString();
        }
```

```
 public static void main(String arg[])
 {
 String s=JOptionPane.showInputDialog("enter a string");
 JOptionPane.showMessageDialog(null,reverseIt(s),"StringBuffer test : inverse string",JOptionPane.PLAIN_MESSAGE);
 }
}
```



14.4.2 StringBuffer class test

```
import java.io.*;

class javaCoffee{

    public static void main (String args[])
    {
    StringBuffer sb = new StringBuffer("Drink java Coffee!");
            sb.insert(6, "hot ");
            System.out.println(sb.toString());
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" javaCoffee
Drink hot java Coffee!

> Terminated with exit code 0.
```

Constructor Summary

StringBuffer()
Constructs a string buffer with no characters in it and an initial capacity of 16 characters.

StringBuffer(CharSequence seq)
Constructs a string buffer that contains the same characters as the specified CharSequence.

StringBuffer(int capacity)
Constructs a string buffer with no characters in it and the specified initial capacity.

StringBuffer(String str)
Constructs a string buffer initialized to the contents of the specified string.

Methods of the StringBuffer class is given belove:

Method Summary

StringBuffer	append(boolean b) Appends the string representation of the boolean argument to the sequence.
StringBuffer	append(char c) Appends the string representation of the char argument to this sequence.
StringBuffer	append(char[] str) Appends the string representation of the char array argument to this sequence.
StringBuffer	append(char[] str, int offset, int len) Appends the string representation of a subarray of the char array argument to this sequence.
StringBuffer	append(CharSequence s) Appends the specified CharSequence to this sequence.
StringBuffer	append(CharSequence s, int start, int end) Appends a subsequence of the specified CharSequence to this sequence.
StringBuffer	append(double d) Appends the string representation of the double argument to this sequence.
StringBuffer	append(float f) Appends the string representation of the float argument to this sequence.
StringBuffer	append(int i) Appends the string representation of the int argument to this sequence.
StringBuffer	append(long lng) Appends the string representation of the long argument to this sequence.
StringBuffer	append(Object obj) Appends the string representation of the Object argument.
StringBuffer	append(String str) Appends the specified string to this character sequence.
StringBuffer	append(StringBuffer sb) Appends the specified StringBuffer to this sequence.
StringBuffer	appendCodePoint(int codePoint) Appends the string representation of the codePoint argument to this sequence.
İnt	capacity() Returns the current capacity.
Char	charAt(int index) Returns the char value in this sequence at the specified index.
İnt	codePointAt(int index) Returns the character (Unicode code point) at the specified index.
İnt	codePointBefore(int index) Returns the character (Unicode code point) before the specified index.
İnt	codePointCount(int beginIndex, int endIndex) Returns the number of Unicode code points in the specified text range of this sequence.
StringBuffer	delete(int start, int end) Removes the characters in a substring of this sequence.
StringBuffer	deleteCharAt(int index) Removes the char at the specified position in this sequence.
Void	ensureCapacity(int minimumCapacity) Ensures that the capacity is at least equal to the specified minimum.

Void	getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) Characters are copied from this sequence into the destination character array dst.
İnt	indexOf(String str) Returns the index within this string of the first occurrence of the specified substring.
İnt	indexOf(String str, int fromIndex) Returns the index within this string of the first occurrence of the specified substring, starting at the specified index.
StringBuffer	insert(int offset, boolean b) Inserts the string representation of the boolean argument into this sequence.
StringBuffer	insert(int offset, char c) Inserts the string representation of the char argument into this sequence.
StringBuffer	insert(int offset, char[] str) Inserts the string representation of the char array argument into this sequence.
StringBuffer	insert(int index, char[] str, int offset, int len) Inserts the string representation of a subarray of the str array argument into this sequence.
StringBuffer	insert(int dstOffset, CharSequence s) Inserts the specified CharSequence into this sequence.
StringBuffer	insert(int dstOffset, CharSequence s, int start, int end) Inserts a subsequence of the specified CharSequence into this sequence.
StringBuffer	insert(int offset, double d) Inserts the string representation of the double argument into this sequence.
StringBuffer	insert(int offset, float f) Inserts the string representation of the float argument into this sequence.
StringBuffer	insert(int offset, int i) Inserts the string representation of the second int argument into this sequence.
StringBuffer	insert(int offset, long l) Inserts the string representation of the long argument into this sequence.
StringBuffer	insert(int offset, Object obj) Inserts the string representation of the Object argument into this character sequence.
StringBuffer	insert(int offset, String str) Inserts the string into this character sequence.
İnt	lastIndexOf(String str) Returns the index within this string of the rightmost occurrence of the specified substring.
İnt	lastIndexOf(String str, int fromIndex) Returns the index within this string of the last occurrence of the specified substring.
İnt	length() Returns the length (character count).
İnt	offsetByCodePoints(int index, int codePointOffset) Returns the index within this sequence that is offset from the given index by codePointOffset code points.
StringBuffer	replace(int start, int end, String str) Replaces the characters in a substring of this sequence with characters in the specified String.
StringBuffer	reverse() Causes this character sequence to be replaced by the reverse of the sequence.
Void	setCharAt(int index, char ch) The character at the specified index is set to ch.
Void	setLength(int newLength) Sets the length of the character sequence.
CharSequence	subSequence(int start, int end)

Returns a new character sequence that is a subsequence of this sequence.

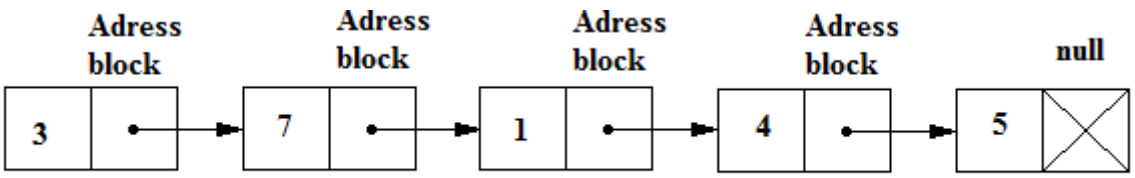
String	substring(int start) Returns a new String that contains a subsequence of characters currently contained in this character sequence.
String	substring(int start, int end) Returns a new String that contains a subsequence of characters currently contained in this sequence.
String	toString() Returns a string representing the data in this sequence.
Void	trimToSize() Attempts to reduce storage used for the character sequence.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

14.5 DYNAMIC MEMORY ALLOCATTED CLASSES : LIST, ARRAYLIST, VECTOR, LINKEDLIST,QUE CLASSES

So far if a set of variable is required, array structures are used. In the array structures memory allocation is static, when you declare the array, you also declare the memory adresses. This adresses can not be change during running of the program. In modern programming changing the array size while running the program (Dynamic allocation) is quite important. Dynamic array structures are created by a node structure. Each node carries a number or other type object information plus an adress to the next connection node block. Nodes can be connected in this fashion and the last element in the node simply does not carry any adres so array will be ended in this element. If you would like to add another element to this list, you simply add a new adres to the last element, and a new node can be added into the structure. Or you can add another node to the front and give the adres of the first node. This type of data structures called List in general



Java has built in List structures. But in order to understand the structure, a List structure program is created for you. Note that List class is defined a Linked List. It can be added or removed to the list from either end.

14.5.1 ListNode and List classes as a dynamic data allocation example

```
Class ListNode
{
 Object data;
 ListNode next;
 ListNode(Object o) {this(o,null);}
 ListNode(Object o,ListNode nextNode)
 {
 data=o;
 next=nextNode;
 }
```

```
  Object getObject() {return data;}
  ListNode getnext() {return next;}
}

public class List
{
//List
  private ListNode firstNode;
  private ListNode lastNode;
  private String name;
  public List(String s)
  {
  name=s;
  firstNode=lastNode=null;
  }
  public List(){this("List ");}
  public synchronized void insertAtFront(Object insertItem)
  {
  if( isEmpty() )
      firstNode = lastNode = new ListNode(insertItem);
  else
     firstNode = new ListNode(insertItem,firstNode);
  }
  public synchronized void insertAtBack(Object insertItem)
  {
  if(isEmpty())
      firstNode=lastNode=new ListNode(insertItem);
  else
     lastNode=lastNode.next=new ListNode(insertItem);
  }
  public synchronized Object removeFromFront()
  throws EmptyListException
  {
  Object removeItem=null;
  if(isEmpty())
     throw new EmptyListException(name);
  removeItem=firstNode.data;
  if(firstNode.equals(lastNode))
     firstNode=lastNode=null;
  else
     firstNode=firstNode.next;
  return removeItem;
  }
  public synchronized Object removeFromBack() throws EmptyListException
  {
  Object removeItem=null;
  if(isEmpty())
     throw new EmptyListException(name);
  removeItem=lastNode.data;
  if(firstNode.equals(lastNode))
     firstNode=lastNode=null;
  else
  {
     ListNode current=firstNode;
     while(current.next != lastNode)
        current=current.next;
     lastNode=current;
     current.next=null;
```

```
  }
  return removeItem;
  }
  public boolean isEmpty() {return firstNode==null; }
  public void print() throws EmptyListException
  {
    if(isEmpty())
    {  throw new EmptyListException(name); }
    System.out.print(" "+name+" : ");
    ListNode current=firstNode;
    while(current!=null)
    {
    System.out.print(current.data.toString()+" ");
    current=current.next;
    }
    System.out.println("\n");
  }
}
```

14.5.2 List class test program

```
Public class ListTest
{
  public static void main(String args[])
  {
    List objList=new List();
    Boolean b=new Boolean(true);
    Character c=new Character('$');
    Integer i=new Integer(34567);
    String s=new String("hello");
    objList.insertAtFront(b);
    objList.print();
    objList.insertAtFront(c);
    objList.print();
    objList.insertAtFront(i);
    objList.print();
    objList.insertAtFront(s);
    objList.print();
    Object removedObj;
    try{
      removedObj=objList.removeFromFront();
      System.out.print(removedObj.toString()+" removed");
      objList.print();
      removedObj=objList.removeFromFront();
      System.out.print(removedObj.toString()+" removed");
      objList.print();
      removedObj=objList.removeFromBack();
      System.out.print(removedObj.toString()+" removed");
      objList.print();
      removedObj=objList.removeFromBack();
      System.out.print(removedObj.toString()+" removed");
      objList.print();
      }
      catch(EmptyListException e) {System.out.println("\n"+e.toString());}
  }
}
```

Program output will be:

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" ListTest
List  : true
List  : $ true
List  : 34567 $ true
List  : hello 34567 $ true
hello removed List  : 34567 $ true
34567 removed List  : $ true
true removed List  : $
$ removed
EmptyListException:  List  List is empty

> Terminated with exit code 0.
```

As you see, a new elements can easily be added or removed from the structure.

In java structures to define Lists are available. ArrayList, LinkedList and Vector classes can be used for the same purpose. Therefore user defined Dynamic array structures are not needed. Example problems are given to show us how this classes can be used. Our first example will be use ArrayList class as dynamically allocated array structure. A Collection class is also used. This class is utilised to sort, reverse, mix type of operations of dynamic array structures.

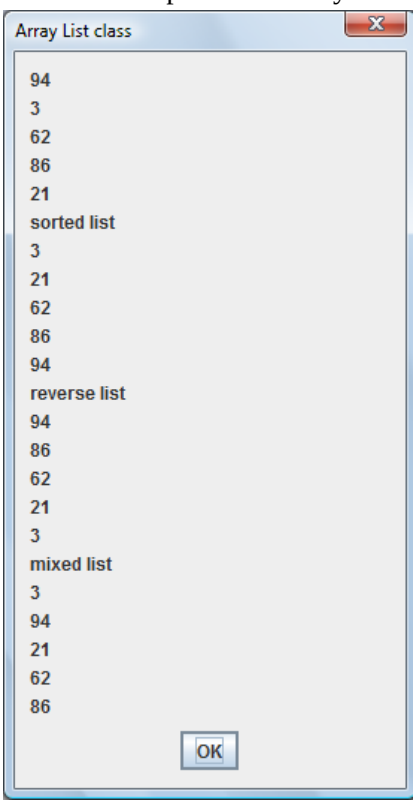
14.5.3 Java library class ArrayList test program

```
import java.util.*;
import javax.swing.*;

public class ArrayList1
{
  public static String print(Collection c)
  {
    String s="";
    Iterator i=c.iterator();
    while(i.hasNext()) {s+=i.next()+"\n";}
    return s;
  }

  public static void main(String args[])
  {
  ArrayList<Integer> a=new ArrayList<Integer>();
  for(int i=0;i<5;i++)
  {int sayi=((int)(Math.random()*100));
   a.add(new Integer(sayi));
  }
  String s1="Array List class";
  String s=print(a);
  s+="sorted list\n";
  Collections.sort(a);
  s+=print(a);
  s+="reverse list\n";
  Collections.reverse(a);
  s+=print(a);
   s+="mixed list\n";
  Collections.shuffle(a);
  s+=print(a);
  JOptionPane.showMessageDialog(null,s,s1,
  JOptionPane.PLAIN_MESSAGE);
  }}
```

In the program ArrayList is defined by the constructor **ArrayList<Integer> a=new ArrayList<Integer>();** **<Integer>** java object variable type Integer. ArrayList uses **Generic** type input variables. Any java Object type can be used in ArrayList or other dynamically define java data structures. For example, if it is desired to define String type ArrayList constructor will be :

ArrayList<String> a=new ArrayList<String>( );

A class **Iterator** is used to collect data from the ArrayList class. The main advantages of this class is that it will stop automatically when the array is empty. The result of the program :



In the next program instead of using Iterator class, elements at the ArrayList is reached directly by using **size()** and get() methods of ArrayList class. Result will be similar.

14.5.4 Java library class ArrayListList test program

```
import java.util.*;
import javax.swing.*;

public class ArrayList2
{

  public static void main(String args[])
  {

  ArrayList<Integer> a=new ArrayList<Integer>();
  for(int i=0;i<5;i++)
  {int sayi=((int)(Math.random()*100));
   a.add(new Integer(sayi));
  }
  String s1="Array List class";
  String s="";
```

```
 for (int i=0, n=a.size(); i < n; i++) {s+=a.get(i)+"\n";};
 JOptionPane.showMessageDialog(null,s,s1,
 JOptionPane.PLAIN_MESSAGE);
 }
}
```

An example program showing using Lambda expressions together with ArrayList

14.5.4 Java library class ArrayList List test program with lambda expression usage (predicate Function)

```
import java.util.function.Predicate;
import java.util.*;
import javax.swing.*;

public class ArrayList3 {

   public static void main(String [] arg)  {

      ArrayList<Integer> list =new  ArrayList<Integer>();
      int sayi;
      for(int i=0;i<10;i++)
      {sayi=(int)(20*Math.random());
       list.add(new Integer(sayi));
      }
      System.out.println("Print all numbers:");
      evaluate(list, (n)->true);
      System.out.println("Print no numbers:");
      evaluate(list, (n)->false);
      System.out.println("Print even numbers:");
      evaluate(list, (n)-> n%2 == 0 );
      System.out.println("Print odd numbers:");
      evaluate(list, (n)-> n%2 == 1 );
      System.out.println("Print numbers greater than 5:");
      evaluate(list, (n)-> n > 5 );
   }

   public static void evaluate(ArrayList<Integer> list, Predicate<Integer> predicate) {
      for(Integer n: list)
      {if(predicate.test(n)) {
       System.out.println(n + " ");
      }
      }
   }

}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" ArrayList3
Print all numbers:
```

```
17
19
13
8
9
5
18
19
10
5
Print no numbers:
Print even numbers:
8
18
10
Print odd numbers:
17
19
13
9
5
19
5
Print numbers greater than 5:
17
19
13
8
9
18
19
10

> Terminated with exit code 0.
```

In the next example we define two sets as ArrayList<String>, then calculate union and intersections of the sets

14.5.5 SetA.java a class example using ArrayList

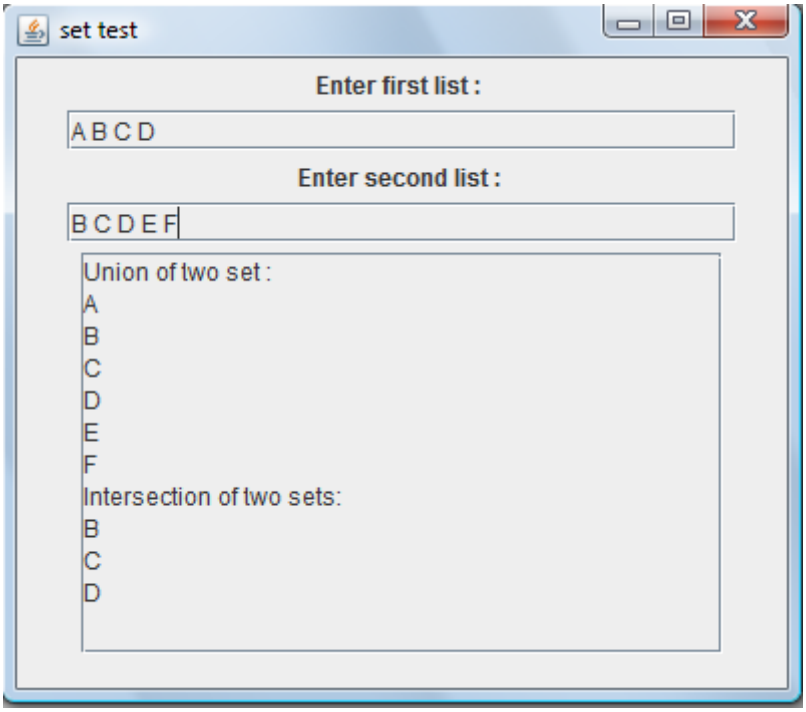
```
 import java.util.*;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

class SetA
{
   public ArrayList<String> union(ArrayList<String> list1, ArrayList<String> list2)
   {
      ArrayList<String> unionArray = new ArrayList<String>();
```

```
        String s1,s2;
        Iterator n1=list1.iterator();
        Iterator n2=list2.iterator();
        while(n1.hasNext())
        {  s1=(String)n1.next();
           unionArray.add(s1);
        }
        while(n2.hasNext())
        {
           s2=(String)n2.next();
           if(!unionArray.contains(s2))
              unionArray.add(s2);
        }
        return unionArray;
    }

    public ArrayList<String> intersection(ArrayList<String> list1, ArrayList<String> list2)
    {
        ArrayList<String> intersectionArray = new ArrayList<String>();
        String s;
        Iterator n = list1.iterator();
        while(n.hasNext())
        {
           s=(String)n.next();
           if(list2.contains(s))
              intersectionArray.add(s);
        }
        return intersectionArray;
    }
}

public class  SetTest extends JFrame implements ActionListener
{
    SetA    set = new SetA();   // SetA
    JLabel L1, P1, P2;
    JTextField T1,T2;
    JTextArea output;

//  dinamik boyutlu degisken sinifi Vector'u kullanarak
//  iki vector listesi yarat
    ArrayList<String> list1 = new ArrayList<String>();
    ArrayList<String>list2  = new ArrayList<String>();

//  Sonuclari yine vektor cinsinden de§iskenler
//  kullanarak aktar
    ArrayList<String> ansUn;
    ArrayList<String> ansInt;
    Container c;

    public SetTest()
    {
        super("set test");
        c=getContentPane();
        c.setLayout(new FlowLayout());
        P1 = new JLabel("Enter first list : ");
        T1 = new JTextField(30);
        T1.setBackground(c.getBackground());
        P2 = new JLabel("Enter second list : ");
```

```
        T2 = new JTextField(30);
        T2.setBackground(c.getBackground());
        output = new JTextArea();
        output.setEditable(false);
        output.setText("");
        output.setBackground(c.getBackground());
        c.add(P1);
        c.add(T1);
        T1.addActionListener(this);
        c.add(P2);
        c.add(T2);
        T2.addActionListener(this);
        JScrollPane jp=new  JScrollPane(output);
        jp.setPreferredSize(new Dimension(320,200));
        c.add(jp);
    }

    public void inputA(String string, ArrayList<String> list)
    {
        StringTokenizer tokens = new StringTokenizer(string);
        while(tokens.hasMoreTokens()) {
            String test = tokens.nextToken();
            list.add(test);
        }
    }

    public void outputA(ArrayList<String> v)
    {
        Iterator enm = v.iterator();
        while(enm.hasNext())
        {
            String ans = (String)enm.next();
            output.append(ans + "\n");
        }
    }

    public void actionPerformed(ActionEvent e)
    {
         if(e.getSource()==T1)
         {
            String stringToTokenize = T1.getText();
            inputA(stringToTokenize,list1);
         }
        else if(e.getSource()==T2)
        {
            String stringToTokenize = T2.getText();
            inputA(stringToTokenize,list2);
//          calculate union and interface sets for two input set
            ansUn = set.union(list1,list2);
            ansInt = set.intersection(list1,list2);
//          show results in the output field
            output.setText("");
            output.append("Union of two set : \n");
            outputA(ansUn);
            output.append("intersection of two set : \n");
            outputA(ansInt);
        }
    } //end of actionperformed method
```

```
    public static void main(String[] args)
    {
    SetTest pencere = new SetTest();
    pencere.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    pencere.setSize(400,350);
    pencere.setVisible(true);
    }
  }
```

The program output:



Methods of the ArrayList class can be given as Table

Constructor Summary

ArrayList()
Constructs an empty list with an initial capacity of ten.

ArrayList(Collection c)
Constructs a list containing the elements of the specified collection, in the order they are returned by the collection's iterator.

ArrayList(int initialCapacity)
Constructs an empty list with the specified initial capacity.

Method Summary

void	add(int index, Object element) Inserts the specified element at the specified position in this list.
boolean	add(Object o) Appends the specified element to the end of this list.

boolean addAll(Collection c)
Appends all of the elements in the specified Collection to the end of this list, in the order that they are returned by the specified Collection's Iterator.

boolean addAll(int index, Collection c)
Inserts all of the elements in the specified Collection into this list, starting at the specified position.

void clear()
Removes all of the elements from this list.

Object clone()
Returns a shallow copy of this ArrayList instance.

boolean contains(Object elem)
Returns true if this list contains the specified element.

void ensureCapacity(int minCapacity)
Increases the capacity of this ArrayList instance, if necessary, to ensure that it can hold at least the number of elements specified by the minimum capacity argument.

Object get(int index)
Returns the element at the specified position in this list.

int indexOf(Object elem)
Searches for the first occurence of the given argument, testing for equality using the equals method.

boolean isEmpty()
Tests if this list has no elements.

int lastIndexOf(Object elem)
Returns the index of the last occurrence of the specified object in this list.

Object remove(int index)
Removes the element at the specified position in this list.

protected void removeRange(int fromIndex, int toIndex)
Removes from this List all of the elements whose index is between fromIndex, inclusive and toIndex, exclusive.

Object set(int index, Object element)
Replaces the element at the specified position in this list with the specified element.

int size()
Returns the number of elements in this list.

Object[] toArray()
Returns an array containing all of the elements in this list in the correct order.

Object[] toArray(Object[] a)
Returns an array containing all of the elements in this list in the correct order; the runtime type of the returned

array is that of the specified array.

void trimToSize()

Trims the capacity of this ArrayList instance to be the list's current size.

It should be noted that by using method ***a*.add(*i*, *e*)** a new node can be added into any plave in the dynamic array structure.

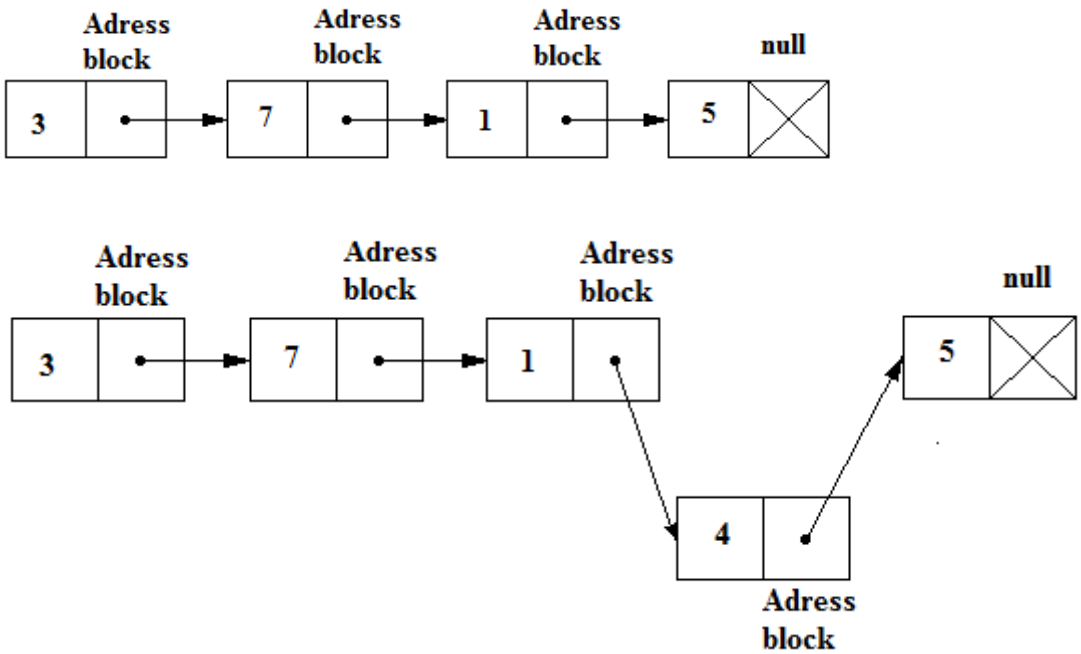


Figure 14.3.1 Adding up a new adress in between two elements in ArrayList or Vector classes

Vector class is very similar to ArrayList class. In fact they can replace each other in utilisation

14.5.6 Java library class Vector test program

```
İmport java.util.*;
import javax.swing.*;

public class Vector1
{
 public static String print(Collection c)
 {
  String s="";
  Iterator i=c.iterator();
  while(i.hasNext()) {s+=i.next()+"\n";}
  return s;
 }

 public static void main(String args[])
 {

 Vector<Integer> a=new Vector<Integer>();
 for(int i=0;i<5;i++)
 {int sayi=((int)(Math.random()*100));
```

```
  a.add(new Integer(sayi));
 }
 String s1="Array List class";
 String s=print(a);
 s+="sorted list\n";
 Collections.sort(a);
 s+=print(a);
 JOptionPane.showMessageDialog(null,s,s1,
 JOptionPane.PLAIN_MESSAGE);
 }
}
```

A second example in the Vector class

14.5.7 Java library class Vector test program

```
import java.util.*;
import javax.swing.JOptionPane;

public class Planets1{
 public static String print(Collection c)
 {
  String s="";
  Iterator i=c.iterator();
  while(i.hasNext()) {s+=i.next()+"\n";}
  return s;
 }

 public static void main (String args[]) {
  String names[] = {"Mercur", "Venus", "Earth",
   "Mars", "Jupiter", "Saturn", "Uranus",
   "Neptun", "Pluto"};
  Vector<String> Planets = new Vector<String> ();
  for(int i=0;i<names.length;i++)
  { Planets.add(names[i]);}
  String ss=print(Planets);
  JOptionPane.showMessageDialog(null,ss);
 }
}
```

Methods of the Vector class is given in the table belove

Field Summary

protected int capacityIncrement
The amount by which the capacity of the vector is automatically incremented when its size becomes greater than its capacity.

protected int elementCount
The number of valid components in this Vector object.

protected Object[] elementData
The array buffer into which the components of the vector are stored.

Fields inherited from class java.util.AbstractList

modCount

Constructor Summary

Vector()
Constructs an empty vector so that its internal data array has size 10 and its standard capacity increment is zero.

Vector(Collection c)
Constructs a vector containing the elements of the specified collection, in the order they are returned by the collection's iterator.

Vector(int initialCapacity)
Constructs an empty vector with the specified initial capacity and with its capacity increment equal to zero.

Vector(int initialCapacity, int capacityIncrement)
Constructs an empty vector with the specified initial capacity and capacity increment.

Method Summary

void add(int index, Object element)
Inserts the specified element at the specified position in this Vector.

boolean add(Object o)
Appends the specified element to the end of this Vector.

boolean addAll(Collection c)
Appends all of the elements in the specified Collection to the end of this Vector, in the order that they are returned by the specified Collection's Iterator.

boolean addAll(int index, Collection c)
Inserts all of the elements in in the specified Collection into this Vector at the specified position.

void addElement(Object obj)
Adds the specified component to the end of this vector, increasing its size by one.

int capacity()
Returns the current capacity of this vector.

void clear()
Removes all of the elements from this Vector.

Object clone()
Returns a clone of this vector.

boolean contains(Object elem)
Tests if the specified object is a component in this vector.

boolean containsAll(Collection c)
Returns true if this Vector contains all of the elements in the specified Collection.

void copyInto(Object[] anArray)
Copies the components of this vector into the specified array.

Object	elementAt(int index) Returns the component at the specified index.
Enumeration	elements() Returns an enumeration of the components of this vector.
void	ensureCapacity(int minCapacity) Increases the capacity of this vector, if necessary, to ensure that it can hold at least the number of components specified by the minimum capacity argument.
boolean	equals(Object o) Compares the specified Object with this Vector for equality.
Object	firstElement() Returns the first component (the item at index 0) of this vector.
Object	get(int index) Returns the element at the specified position in this Vector.
int	hashCode() Returns the hash code value for this Vector.
int	indexOf(Object elem) Searches for the first occurence of the given argument, testing for equality using the equals method.
int	indexOf(Object elem, int index) Searches for the first occurence of the given argument, beginning the search at index, and testing for equality using the equals method.
void	insertElementAt(Object obj, int index) Inserts the specified object as a component in this vector at the specified index.
boolean	isEmpty() Tests if this vector has no components.
Object	lastElement() Returns the last component of the vector.
int	lastIndexOf(Object elem) Returns the index of the last occurrence of the specified object in this vector.
int	lastIndexOf(Object elem, int index) Searches backwards for the specified object, starting from the specified index, and returns an index to it.
Object	remove(int index) Removes the element at the specified position in this Vector.
boolean	remove(Object o) Removes the first occurrence of the specified element in this Vector If the Vector does not contain the element, it is unchanged.
boolean	removeAll(Collection c) Removes from this Vector all of its elements that are contained in the specified Collection.
void	removeAllElements() Removes all components from this vector and sets its size to zero.
boolean	removeElement(Object obj) Removes the first (lowest-indexed) occurrence of the argument from this vector.
void	removeElementAt(int index) Deletes the component at the specified index.
protected void	removeRange(int fromIndex, int toIndex) Removes from this List all of the elements whose index is between fromIndex, inclusive and toIndex, exclusive.
boolean	retainAll(Collection c) Retains only the elements in this Vector that are contained in the specified Collection.
Object	set(int index, Object element) Replaces the element at the specified position in this Vector with the specified element.

void setElementAt(Object obj, int index)
Sets the component at the specified index of this vector to be the specified object.

void setSize(int newSize)
Sets the size of this vector.

int size()
Returns the number of components in this vector.

List subList(int fromIndex, int toIndex)
Returns a view of the portion of this List between fromIndex, inclusive, and toIndex, exclusive.

Object[] toArray()
Returns an array containing all of the elements in this Vector in the correct order.

Object[] toArray(Object[] a)
Returns an array containing all of the elements in this Vector in the correct order; the runtime type of the returned array is that of the specified array.

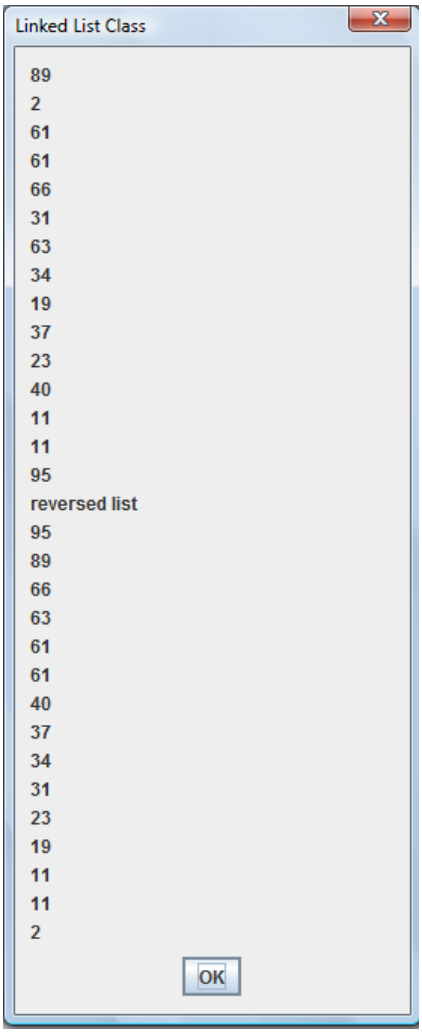
String toString()
Returns a string representation of this Vector, containing the String representation of each element.

void trimToSize()
Trims the capacity of this vector to be the vector's current size.

Utilisation of **LinkedList** is similar. There is one difference at ArrayList and Vector a new element can be added in any point in the array. In the Linked List it can be added only to the either end. Remember that the example program 14.3.1(List) was a LinkedList

14.5.8 Java library class LinkedListList test program

```
import java.util.*;
import javax.swing.*;

public class LinkedList1
{
 public static String print(Collection c)
 {
  String s="";
  Iterator i=c.iterator();
  while(i.hasNext()) {s+=i.next()+"\n";}
  return s;
 }

 public static void main(String args[])
 {

 LinkedList<Integer> a=new LinkedList<Integer>();
 for(int i=0;i<15;i++)
 {int sayi=((int)(Math.random()*100));
  a.add(new Integer(sayi));
 }
 String s1="Linked List Class";
 String s=print(a);
 s+="tersten sıraya sokulmuş liste\n";
 Collections.sort(a,Collections.reverseOrder());
 s+=print(a);
 JOptionPane.showMessageDialog(null,s,s1,
 JOptionPane.PLAIN_MESSAGE);
 }
}
```



The methods of the LinkedList is given in the table belove

Constructor Summary

LinkedList()

Constructs an empty list.

LinkedList(Collection<? extends E> c)

Constructs a list containing the elements of the specified collection, in the order they are returned by the collection's iterator.

Method Summary

boolean	add(E o) Appends the specified element to the end of this list.
void	add(int index, E element) Inserts the specified element at the specified position in this list.
boolean	addAll(Collection<? extends E> c) Appends all of the elements in the specified collection to the end of this list, in the order that they are returned by the specified collection's iterator.
boolean	addAll(int index, Collection<? extends E> c) Inserts all of the elements in the specified collection into this list, starting at the specified position.
void	addFirst(E o)

Inserts the given element at the beginning of this list.

void	addLast(E o) Appends the given element to the end of this list.
void	clear() Removes all of the elements from this list.
Object	clone() Returns a shallow copy of this LinkedList.
boolean	contains(Object o) Returns true if this list contains the specified element.
E	element() Retrieves, but does not remove, the head (first element) of this list.
E	get(int index) Returns the element at the specified position in this list.
E	getFirst() Returns the first element in this list.
E	getLast() Returns the last element in this list.
int	indexOf(Object o) Returns the index in this list of the first occurrence of the specified element, or -1 if the List does not contain this element.
int	lastIndexOf(Object o) Returns the index in this list of the last occurrence of the specified element, or -1 if the list does not contain this element.
ListIterator<E>	listIterator(int index) Returns a list-iterator of the elements in this list (in proper sequence), starting at the specified position in the list.
boolean	offer(E o) Adds the specified element as the tail (last element) of this list.
E	peek() Retrieves, but does not remove, the head (first element) of this list.
E	poll() Retrieves and removes the head (first element) of this list.
E	remove() Retrieves and removes the head (first element) of this list.
E	remove(int index) Removes the element at the specified position in this list.
boolean	remove(Object o) Removes the first occurrence of the specified element in this list.
E	removeFirst() Removes and returns the first element from this list.
E	removeLast() Removes and returns the last element from this list.
E	set(int index, E element) Replaces the element at the specified position in this list with the specified element.
int	size()

Returns the number of elements in this list.

Object[] toArray()

Returns an array containing all of the elements in this list in the correct order.

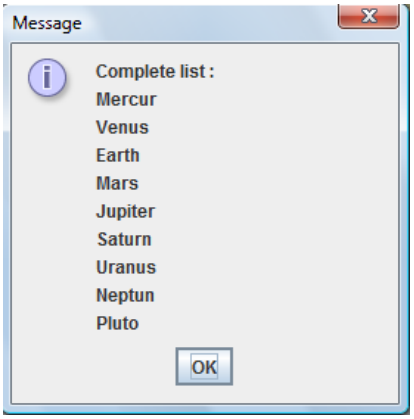
<T> T[] toArray(T[] a)

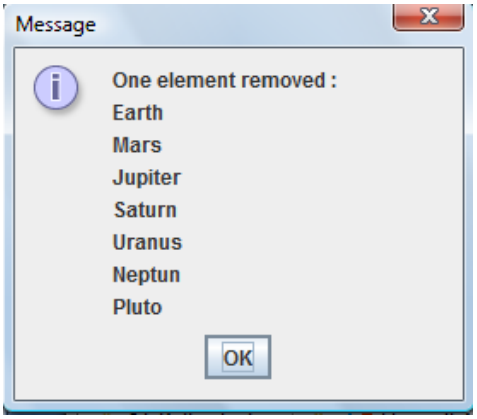
Returns an array containing all of the elements in this list in the correct order; the runtime type of the returned array is that of the specified array.

The Que dynamic constructor class **ArrayDeque** is also similar to the previous classes. Eventhough it has methods to add and remove elements to the both sides of the vector, usual interperetation of a que is to add element to the one end and remove the element from the other end, so that elements will remain in a que, and first come will be served first. Here is an example problem:

Program 14.5.9 Planets2.java an example of ArrayDeque class

```
import java.util.*;
import javax.swing.JOptionPane;

public class Planets2{
  public static String print(Collection c)
  {
    String s="";
    Iterator i=c.iterator();
    while(i.hasNext()) {s+=i.next()+"\n";}
    return s;
  }

  public static void main (String args[]) {
    String names[] = {"Mercur", "Venus", "Earth",
    "Mars", "Jupiter", "Saturn", "Uranus",
    "Neptun", "Pluto"};
    ArrayDeque<String> Planets = new ArrayDeque<String> ();
    for(int i=0;i<names.length;i++)
    { Planets.add(names[i]);}
    String ss="Complete list : \n"+print(Planets);
    JOptionPane.showMessageDialog(null,ss);
    Planets.removeFirst();
    ss="One element removed :\n"+print(Planets);
    JOptionPane.showMessageDialog(null,ss);
    Planets.removeFirst();
    ss="One element removed :\n"+print(Planets);
    JOptionPane.showMessageDialog(null,ss);
  }
}
```





Methods of the **ArrayDeque** are given in the following table:

Constructor Summary

ArrayDeque()
Constructs an empty array deque with an initial capacity sufficient to hold 16 elements.

ArrayDeque(Collection<? extends E> c)
Constructs a deque containing the elements of the specified collection, in the order they are returned by the collection's iterator.

ArrayDeque(int numElements)
Constructs an empty array deque with an initial capacity sufficient to hold the specified number of elements.

Method Summary

boolean	add(E e) Inserts the specified element at the end of this deque.
void	addFirst(E e) Inserts the specified element at the front of this deque.
void	addLast(E e) Inserts the specified element at the end of this deque.
void	clear() Removes all of the elements from this deque.
ArrayDeque<E>	clone() Returns a copy of this deque.
boolean	contains(Object o) Returns true if this deque contains the specified element.
Iterator<E>	descendingIterator() Returns an iterator over the elements in this deque in reverse sequential order.
E	element() Retrieves, but does not remove, the head of the queue represented by this deque.
E	getFirst() Retrieves, but does not remove, the first element of this deque.
E	getLast() Retrieves, but does not remove, the last element of this deque.
boolean	isEmpty() Returns true if this deque contains no elements.
Iterator<E>	iterator() Returns an iterator over the elements in this deque.
boolean	offer(E e) Inserts the specified element at the end of this deque.
boolean	offerFirst(E e)

Inserts the specified element at the front of this deque.

boolean offerLast(E e)
Inserts the specified element at the end of this deque.

E peek()
Retrieves, but does not remove, the head of the queue represented by this deque, or returns null if this deque is empty.

E peekFirst()
Retrieves, but does not remove, the first element of this deque, or returns null if this deque is empty.

E peekLast()
Retrieves, but does not remove, the last element of this deque, or returns null if this deque is empty.

E poll()
Retrieves and removes the head of the queue represented by this deque (in other words, the first element of this deque), or returns null if this deque is empty.

E pollFirst()
Retrieves and removes the first element of this deque, or returns null if this deque is empty.

E pollLast()
Retrieves and removes the last element of this deque, or returns null if this deque is empty.

E pop()
Pops an element from the stack represented by this deque.

void push(E e)
Pushes an element onto the stack represented by this deque.

E remove()
Retrieves and removes the head of the queue represented by this deque.

boolean remove(Object o)
Removes a single instance of the specified element from this deque.

E removeFirst()
Retrieves and removes the first element of this deque.

boolean removeFirstOccurrence(Object o)
Removes the first occurrence of the specified element in this deque (when traversing the deque from head to tail).

E removeLast()
Retrieves and removes the last element of this deque.

boolean removeLastOccurrence(Object o)
Removes the last occurrence of the specified element in this deque (when traversing the deque from head to tail).

int size()
Returns the number of elements in this deque.

Object[] toArray()
Returns an array containing all of the elements in this deque in proper sequence (from first to last element).

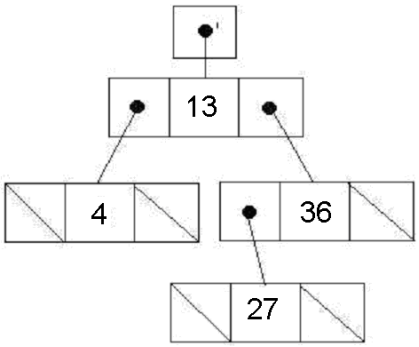
<T> T[]

toArray(T[] a)
Returns an array containing all of the elements in this deque in proper sequence (from first to last element); the runtime type of the returned array is that of the specified array.

14.6 DYNAMIC ARRAY STRUCTURES : TREES

Trees are a dynamic array structure with one datapoint and two self referencing adres. Data is entered to datapoint, the next data point is entered with a given rule, for example if the next data point is a

number and smaller then the number in the node, it will go to the left node. If it is bigger than the node value, it will go to the right node. This way, when tree elements are called back, they will be automaticaly in sorted order. Such a tree structure is shown in the figure.



In order to understand this structure better, a treenode and tree class is created here and tested.

14.6.1 TreeNode and Tree classes

```
class TreeNode
{
  TreeNode left;
  int data;
  TreeNode right;
  public TreeNode(int d)
  {
  data=d;
  left=right=null;
  }
  public synchronized void enter(int d)
  {
  // not ayni deger iki kere enterilmeye calisilirsa
  // ikinci enteris dikkate alinmiyacaktir
    if(d<data)
    { if(left==null) {left=new TreeNode(d);}
      else        {left.enter(d);}
    }
    else if(d>data)
    { if(right==null) {right=new TreeNode(d);}
      else        {right.enter(d);}
    }
  }
}
public class Tree
{
  private TreeNode kok;
  public Tree() {kok=null;}
  public synchronized void enter(int d)
  {
    if(kok==null) kok=new TreeNode(d);
    else kok.enter(d);
  }
  public void node_left_right_sort(){node_left_right_sorter(kok);}
  public void left_node_right_sort(){left_node_right_sorter(kok);}
  public void left_right_node_sort(){left_right_node_sorter(kok);}
  public void node_right_left_sort(){node_right_left_sorter(kok);}
  public void right_node_left_sort(){right_node_left_sorter(kok);}
```

```
  public void right_left_node_sort(){right_left_node_sorter(kok);}
  public void node_left_right_sorter(TreeNode node)
  {
  if(node==null) return;
  System.out.print(node.data+" ");
  node_left_right_sorter(node.left);
  node_left_right_sorter(node.right);
  }
  public void node_right_left_sorter(TreeNode node)
  {
  if(node==null) return;
  System.out.print(node.data+" ");
  node_right_left_sorter(node.right);
  node_right_left_sorter(node.left);
  }
  public void left_node_right_sorter(TreeNode node)
  {
  if(node==null) return;
  left_node_right_sorter(node.left);
  System.out.print(node.data+" ");
  left_node_right_sorter(node.right);
  }
  public void right_node_left_sorter(TreeNode node)
  {
  if(node==null) return;
  right_node_left_sorter(node.right);
  System.out.print(node.data+" ");
  right_node_left_sorter(node.left);
  }
  public void left_right_node_sorter(TreeNode node)
  {
  if(node==null) return;
  left_node_right_sorter(node.left);
  left_node_right_sorter(node.right);
  System.out.print(node.data+" ");
  }
  public void right_left_node_sorter(TreeNode node)
  {
  if(node==null) return;
  right_node_left_sorter(node.right);
  right_node_left_sorter(node.left);
  System.out.print(node.data+" ");
  }
}
```

14.6.2 TreeTest class

```
public class TreeTest
{
  public static void main(String args[])
  {
  Tree tree=new Tree();
  int number;
  System.out.println("numbers originally entered to the tree");
  for(int i=0;i<=6;i++)
  {
   number=(int)(Math.random()*45)+1;
   System.out.print(number+" ");
   tree.enter(number);
```

```
  }
  System.out.println("\nleft tree - node - right tree sort : ");
  tree.left_node_right_sort();
  System.out.println("\nnode -left tree - right tree sort : ");
  tree.node_left_right_sort();
  System.out.println("\nleft tree - right tree - node sort : ");
  tree.left_right_node_sort();
  System.out.println("\nright tree - node - left tree sort : ");
  tree.right_node_left_sort();
  System.out.println("\nnode -right tree - left tree sort : ");
  tree.node_right_left_sort();
  System.out.println("\nright tree - left tree - node sort : ");
  tree.right_left_node_sort();
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" TreeTest
numbers originally entered to the tree
32 42 24 18 39 4 39
left tree - node - right tree sort :
4 18 24 32 39 42
node -left tree - right tree sort :
32 24 18 4 42 39
left tree - right tree - node sort :
4 18 24 39 42 32
right tree - node - left tree sort :
42 39 32 24 18 4
node -right tree - left tree sort :
32 42 39 24 18 4
right tree - left tree - node sort :
42 39 24 18 4 32
> Terminated with exit code 0.
```

As a Java Library example TreeSet will be given. As a member of set this class will not take a repeated value. Note that the values entered are the same as the given figure:

14.6.3 TreeSet class example

```
import java.util.*;
import javax.swing.JOptionPane;

public class TreeTest1
{
  public static String print(Collection c)
  {
    String s="";
    Iterator i=c.iterator();
    while(i.hasNext()) {s+=i.next()+"\n";}
    return s;
  }
  public static void main(String args[])
  {
  TreeSet<Integer> tree=new TreeSet<Integer>();
  int number;
  System.out.println("numbers originally entered to the tree");
```

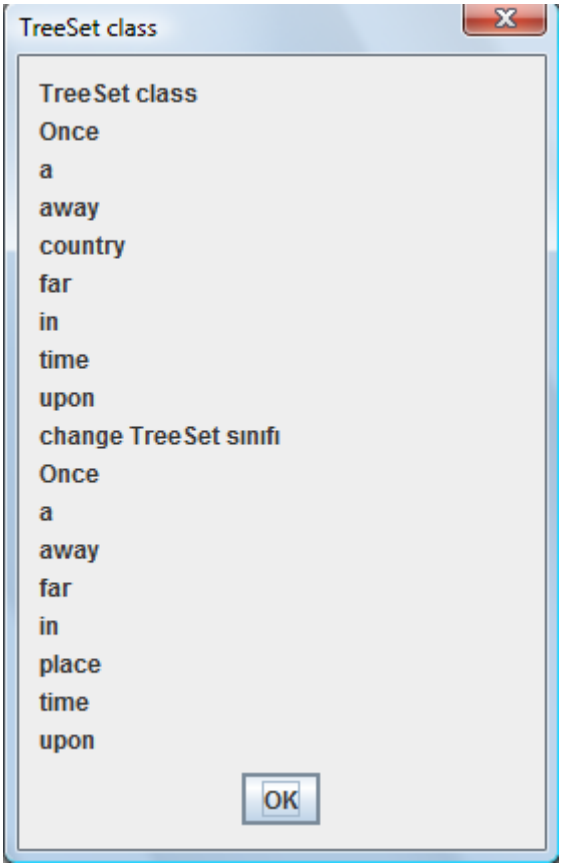
```
  tree.add(new Integer(13));
  tree.add(new Integer(4));
  tree.add(new Integer(36));
  tree.add(new Integer(27));
  String s=print(tree);
  System.out.print(s);
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" TreeTest1
numbers originally entered to the tree
4
13
27
36
> Terminated with exit code 0.
```

Another example of TreeSet class. In this class

14.6.4 TreeSet class example

```
import java.util.*;
import javax.swing.*;

public class TreeSet2
{
  public static String print(Collection c)
  {
    String s="";
    Iterator i=c.iterator();
    while(i.hasNext()) {s+=i.next()+"\n";}
    return s;
  }
  public static void main(String args[])
  {
  TreeSet<String> a=new TreeSet<String>();
  String s[]={"Once","upon","a","time","in", "a","country","far","far","away"};
  for(int i=0;i<s.length;i++) a.add(s[i]);
  String s1="TreeSet class\n";
  String s2="TreeSet class";;
  s1+=print(a);
  a.remove("country");
  a.add("place");
  s1+="change TreeSet sınıfı\n";
  s1+=print(a);
  JOptionPane.showMessageDialog(null,s1,s2,
  JOptionPane.PLAIN_MESSAGE);
  }
}
```



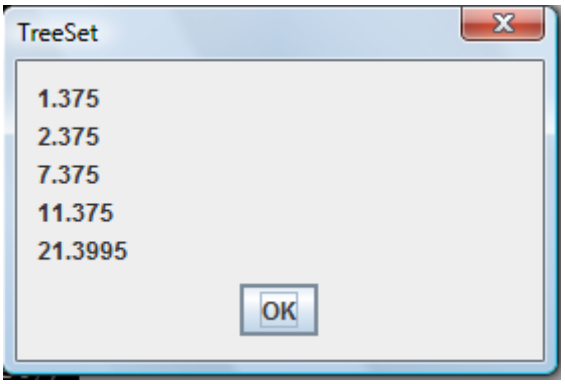
As it is seen from the table repeated values a and far is **listed only once**, not twice

Another Example of TreeSet :

14.6.5 TreeSet class example

```
class TreeSet3 {

        public static void main(String args[])
        {
        TreeSet<Double> a=new TreeSet<Double>();
        Double s1=new Double(1.375);
        Double s2=new Double(2.375);
        Double s3=new Double(21.3995);
        Double s4=new Double(1.375);
        Double s5=new Double(7.375);
        Double s6=new Double(11.375);
        a.add(s1);
        a.add(s2);
        a.add(s3);
        a.add(s4);
        a.add(s5);
        a.add(s6);
        Iterator i=a.iterator();
        String s="";
        while(i.hasNext())
        {s+=i.next()+"\n";}
        JOptionPane.showMessageDialog(null,s,"TreeSet",
        JOptionPane.PLAIN_MESSAGE);
        }
```

```
        }
```



As in the previous example continious value of 1.375 is listed only once in the TreeSet

Methods of TreeSet class is given belove

Constructor Summary

TreeSet()
Constructs a new, empty set, sorted according to the elements' natural order.

TreeSet(Collection c)
Constructs a new set containing the elements in the specified collection, sorted according to the elements' *natural order*.

TreeSet(Comparator c)
Constructs a new, empty set, sorted according to the specified comparator.

TreeSet(SortedSet s)
Constructs a new set containing the same elements as the specified sorted set, sorted according to the same ordering.

Method Summary

boolean	add(Object o) Adds the specified element to this set if it is not already present.
boolean	addAll(Collection c) Adds all of the elements in the specified collection to this set.
Void	clear() Removes all of the elements from this set.
Object	clone() Returns a shallow copy of this TreeSet instance.
Comparator	comparator() Returns the comparator used to order this sorted set, or null if this tree set uses its elements natural ordering.
boolean	contains(Object o) Returns true if this set contains the specified element.
Object	first() Returns the first (lowest) element currently in this sorted set.
SortedSet	headSet(Object toElement) Returns a view of the portion of this set whose elements are strictly less than toElement.
boolean	isEmpty() Returns true if this set contains no elements.
Iterator	iterator() Returns an iterator over the elements in this set.

Object	last()
	Returns the last (highest) element currently in this sorted set.
boolean	remove(Object o)
	Removes the specified element from this set if it is present.
İnt	size()
	Returns the number of elements in this set (its cardinality).
SortedSet	subSet(Object fromElement, Object toElement)
	Returns a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive.
SortedSet	tailSet(Object fromElement)
	Returns a view of the portion of this set whose elements are greater than or equal to fromElement.

Methods inherited from class java.util. AbstractSet

equals, hashCode, removeAll

Methods inherited from class java.util. AbstractCollection

containsAll, retainAll, toArray, toArray, toString

Methods inherited from class java.lang. Object

finalize, getClass, notify, notifyAll, wait, wait, wait

Methods inherited from interface java.util. Set

containsAll, equals, hashCode, removeAll, retainAll, toArray, toArray

The next java library class in the Tree group will be TreeMap class. In this class two argument is required as input . First a simple example is given

14.6.6 TreeMap class example

```
import java.util.*;

public class TreeMapTest{
  public static void main(String[] args) {
    System.out.println("Tree Map Example!\n");
    TreeMap <Integer, String>tMap = new TreeMap<Integer, String>();
    //Addding data to a tree map
    tMap.put(1, "Sunday");
    tMap.put(2, "Monday");
    tMap.put(3, "Tuesday");
    tMap.put(4, "Wednesday");
    tMap.put(5, "Thursday");
    tMap.put(6, "Friday");
    tMap.put(7, "Saturday");
    //Rerieving all keys
    System.out.println("Keys of tree map: " + tMap.keySet());
    //Rerieving all values
    System.out.println("Values of tree map: " + tMap.values());
    //Rerieving the value from key with key number 5
    System.out.println("Key: 5 value: " + tMap.get(5)+ "\n");
    //Rerieving the First key and its value
    System.out.println("First key: " + tMap.firstKey() + " Value: "
                          + tMap.get(tMap.firstKey()) + "\n");
    //Rerieving the Last key and value
    System.out.println("Last key: " + tMap.lastKey() + " Value: "
                            + tMap.get(tMap.lastKey()) + "\n");
    //Removing the first key and value
    System.out.println("Removing first data: " + tMap.remove(tMap.firstKey()));
```

```
    System.out.println("Now the tree map Keys: " + tMap.keySet());
    System.out.println("Now the tree map contain: " + tMap.values() + "\n");
    //Removing the last key and value
    System.out.println("Removing last data: " + tMap.remove(tMap.lastKey()));
    System.out.println("Now the tree map Keys: " + tMap.keySet());
    System.out.println("Now the tree map contain: " + tMap.values());
  }
}
```

As a second example one of the previous example af array structure is taken and changed as a TreeMap class application. Application is a simple library class wich contains books. In the previous example in arrays (chapter 8) books defined as array of books in the library class. In this example books defined as **TreeMap.** Two seperate TreeMap classes are used to arrange it book name and book author base.

14.6.7 book class example

```
public class book
{
String name;
String author;
String clasification;
String language;

public book(String kname,String kauthor,String kclasification,String klanguage)
{name=kname;
 author=kauthor;
 clasification=kclasification;
 language=klanguage;
}

public book(book k)
{name=k.name;
 author=k.author;
 clasification=k.clasification;
language=k.language;
}

public String toString()
{
String s="name = "+name+"\n";
s+="author = "+author+"\n";
s+="book clasification = "+clasification+"\n";
s+="language = "+language+"\n\n";
return s;
}
}
```

14.6.8 fiction class extended from book

```
public class fiction extends book
{
public fiction(String kisim,String kauthor,String klanguage)
{ super(kisim,kauthor,"fiction",klanguage);}
}
```

14.6.9 encyclopedia class extended from book class

```
public class encyclopedia extends book
{
public encyclopedia(String kname,String klanguage)
{ super(kname,"editing board","encyclopedia",klanguage);}
}
```

14.6.10 encyclopedia class extended from book class

```
public class poetry extends book
{
public poetry(String kname,String kauthor,String klanguage)
{ super(kname,kauthor,"poetry",klanguage);}
}
```

14.6.11 library class with TreeMap class

```
import java.util.*;

public class library
{
String name;
int number_of_books;
TreeMap <String, book> book_name = new TreeMap<String, book>();
TreeMap <String, book> author_name = new TreeMap<String, book>();

public library(String kname)
{name=kname;
 number_of_books=0;
}

public void add(book b)
{
book_name.put(b.name,b);
author_name.put(b.author,b);
number_of_books=book_name.size();
}

public void remove(book b)
{
number_of_books--;
book_name.remove(b.name);
author_name.remove(b.author);
number_of_books=book_name.size();
}

public String list_with_name()
{String s=book_name.values().toString();
return s;
}

public String list_with_author()
{String s=author_name.values().toString();
return s;
}

public String get_with_name(String name)
{String s=book_name.get(name).toString();
return s;
}

 public String toString_with_name()
 {            String s="Library name : "+name+"\n";
    s+="Total number of books : "+number_of_books+"\n\n";
    s+=list_with_name();
    return s;
 }

 public String toString_with_author()
 {            String s="Library name : "+name+"\n";
    s+="Total number of books : "+number_of_books+"\n\n";
    s+=list_with_author();
    return s;
 }}
```

14.6.12 booktest1 class with TreeMap class

```
import javax.swing.*;
class booktest1
{
public static void outputwindow(String s)
{
String s1="library class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}
public static void main(String args[])
{
library l=new library("Library of congress");
book magazine=new book("Bilim ve Teknik","TÜBİTAK","magazine","Turkish");
fiction r1=new fiction("It is all quiet in the westen front ","Erich Maria Reamarque","English");
book r2=new book("The God Delusion","Richard Dawkins","fiction","English");
poetry  s1=new poetry("Human lanscapes from my country","Nazım Hikmet","English and Turkish");
encyclopedia a1=new encyclopedia("Encyclopedia Britannica","English");
l.add(magazine);
l.add(r1);
l.add(r2);
l.add(s1));
l.add(a1);
String s=s=l.toString_with_author();
outputwindow(s);
}
}
```

The TreeMap class set methods are as follows:

Nested classes/interfaces inherited from class java.util. AbstractMap

AbstractMap.SimpleEntry<K,V>, AbstractMap.SimpleImmutableEntry<K,V>

Constructor Summary

Constructor and Description

TreeMap()
Constructs a new, empty tree map, using the natural ordering of its keys.

TreeMap(Comparator<? super K> comparator)
Constructs a new, empty tree map, ordered according to the given comparator.

TreeMap(Map<? extends K,? extends V> m)
Constructs a new tree map containing the same mappings as the given map, ordered according to the *natural ordering* of its keys.

TreeMap(SortedMap<K,? extends V> m)
Constructs a new tree map containing the same mappings and using the same ordering as the specified sorted map.

Modifier and Type	Method and Description
Map.Entry<K,V>	ceilingEntry(K key) Returns a key-value mapping associated with the least key greater than or equal to the given key, or null if there is no such key.
K	ceilingKey(K key) Returns the least key greater than or equal to the given key, or null if there is no such key.
Void	clear() Removes all of the mappings from this map.

Object	clone() Returns a shallow copy of this TreeMap instance.
Comparator<? super K>	comparator() Returns the comparator used to order the keys in this map, or null if this map uses the natural ordering of its keys.
boolean	containsKey(Object key) Returns true if this map contains a mapping for the specified key.
boolean	containsValue(Object value) Returns true if this map maps one or more keys to the specified value.
NavigableSet<K>	descendingKeySet() Returns a reverse order NavigableSet view of the keys contained in this map.
NavigableMap<K,V>	descendingMap() Returns a reverse order view of the mappings contained in this map.
Set<Map.Entry<K,V>>	entrySet() Returns a Set view of the mappings contained in this map.
Map.Entry<K,V>	firstEntry() Returns a key-value mapping associated with the least key in this map, or null if the map is empty.
K	firstKey() Returns the first (lowest) key currently in this map.
Map.Entry<K,V>	floorEntry(K key) Returns a key-value mapping associated with the greatest key less than or equal to the given key, or null if there is no such key.
K	floorKey(K key) Returns the greatest key less than or equal to the given key, or null if there is no such key.
V	get(Object key) Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
SortedMap<K,V>	headMap(K toKey) Returns a view of the portion of this map whose keys are strictly less than toKey.
NavigableMap<K,V>	headMap(K toKey, boolean inclusive) Returns a view of the portion of this map whose keys are less than (or equal to, if inclusive is true) toKey.
Map.Entry<K,V>	higherEntry(K key) Returns a key-value mapping associated with the least key strictly greater than the given key, or null if there is no such key.
K	higherKey(K key) Returns the least key strictly greater than the given key, or null if there is no such key.
Set<K>	keySet() Returns a Set view of the keys contained in this map.
Map.Entry<K,V>	lastEntry() Returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.
K	lastKey() Returns the last (highest) key currently in this map.
Map.Entry<K,V>	lowerEntry(K key) Returns a key-value mapping associated with the greatest key strictly less than the given key, or null if there is no such key.
K	lowerKey(K key) Returns the greatest key strictly less than the given key, or null if there is no such key.

NavigableSet<K>	navigableKeySet() Returns a NavigableSet view of the keys contained in this map.
Map.Entry<K,V>	pollFirstEntry() Removes and returns a key-value mapping associated with the least key in this map, or null if the map is empty.
Map.Entry<K,V>	pollLastEntry() Removes and returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.
V	put(K key, V value) Associates the specified value with the specified key in this map.
void	putAll(Map<? extends K,? extends V> map) Copies all of the mappings from the specified map to this map.
V	remove(Object key) Removes the mapping for this key from this TreeMap if present.
int	size() Returns the number of key-value mappings in this map.
NavigableMap<K,V>	subMap(K fromKey, boolean fromInclusive, K toKey, boolean toInclusive) Returns a view of the portion of this map whose keys range from fromKey to toKey.
SortedMap<K,V>	subMap(K fromKey, K toKey) Returns a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive.
SortedMap<K,V>	tailMap(K fromKey) Returns a view of the portion of this map whose keys are greater than or equal to fromKey.
NavigableMap<K,V>	tailMap(K fromKey, boolean inclusive) Returns a view of the portion of this map whose keys are greater than (or equal to, if inclusive is true) fromKey.
Collection<V>	values() Returns a Collection view of the values contained in this map.

Methods inherited from class java.util.AbstractMap

equals, hashCode, isEmpty, toString

Methods inherited from class java.lang.Object

finalize, getClass, notify, notifyAll, wait, wait, wait

Methods inherited from interface java.util.Map

equals, hashCode, isEmpty

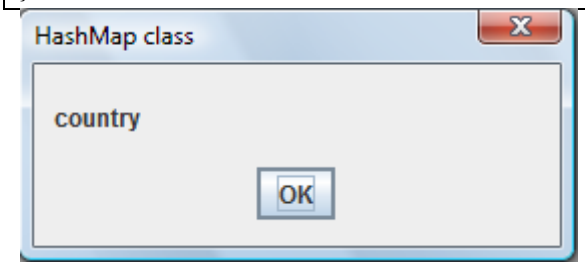
14.7 A DYNAMIC ARRAY STRUCTURES : HASHTABLES

HashTables are data structers that define as a couple of data together. When one of the couple is given the other one can be obtained from this data. As a java class example to hashtables HashMap class will be investigated in here.

14.7.1 HasMap1 class

```
import java.util.*;
import javax.swing.*;

public class HashMap1
{
```

```
 public static void main(String args[])
 {
 HashMap<String,String> a=new HashMap<String,String>();
 String s[]={"Once","upon","a","time","in", "a","country","far","far","away"};
 String w[]={"1","2","3","4","5","6","7","8","9","10"};
 for(int i=0;i<s.length;i++) a.put(w[i],s[i]);
 String s2="HashMap class\n";
 String s1=a.get("7");
 JOptionPane.showMessageDialog(null,s1,s2,
 JOptionPane.PLAIN_MESSAGE);
 }
}
```



A second example is also a derived from a previous example given in arrays (see chapter 8) a vehicle registry class

Program 14.7.2 vehicle class

```
import javax.swing.*;

public class vehicle
{
//valid vehicle types bus,truck,minitruck,minivan,auto
public String vehicletype;
public String brandName;
public double power; //KW olarak
public String color;
public double weight;          //yüksüz net ağırlık
public String registration_number;
//kurucu metodlar

public vehicle()
{vehicletype="not defined";
brandName="not defined";
power=0;
color="not defined";
weight=0;
}

public vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight,String regno)
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
  i_vehicletype.equals("minitruck") ||
  i_vehicletype.equals("bus")||
  i_vehicletype.equals("minibus")||
  i_vehicletype.equals("auto") ||
  i_vehicletype.equals("motorcycle") ||
  i_vehicletype.equals("bicycle") ||
  i_vehicletype.equals("off road vehicle")
  )
```

```
{vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
registration_number=regno;
}
public vehicle(vehicle A)
{//gerekli tanımları yap
vehicletype=A.vehicletype;
brandName=A.brandName;
power=A.power;
color=A.color;
weight=A.weight;
registration_number=A.registration_number;
}
public void define_vehicle(String i_vehicletype,String i_brandName,double i_motor,String i_color,double i_weight,String
regno)
{//vehicle type "bus","truck","mini truck","mini bus","auto","tır"
//if not define as "not defined"
if(i_vehicletype.equals("truck") ||
  i_vehicletype.equals("minitruck") ||
  i_vehicletype.equals("bus")||
  i_vehicletype.equals("minibus")||
  i_vehicletype.equals("auto") ||
  i_vehicletype.equals("motorcycle") ||
  i_vehicletype.equals("bicycle") ||
  i_vehicletype.equals("off road vehicle"))
  {vehicletype=i_vehicletype;}
else
{vehicletype="not defined";}
brandName=i_brandName;
power=i_motor;
color=i_color;
weight=i_weight;
registration_number=regno;
}
public String toString()
{String s1="";
s1+="Vehicle Type : "+vehicletype+"\n";
s1+="Brand Name : "+brandName+"\n";
s1+="Motor Power : "+power+" KW\n";
s1+="Color : "+color+"\n";
s1+="Weight : "+weight+" kg\n";
s1+="Registration number : "+registration_number+"\n\n";
return s1;//gerekli tanımları yap
}
public void outputwindow()
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,toString(),s1, JOptionPane.PLAIN_MESSAGE);
}}
```

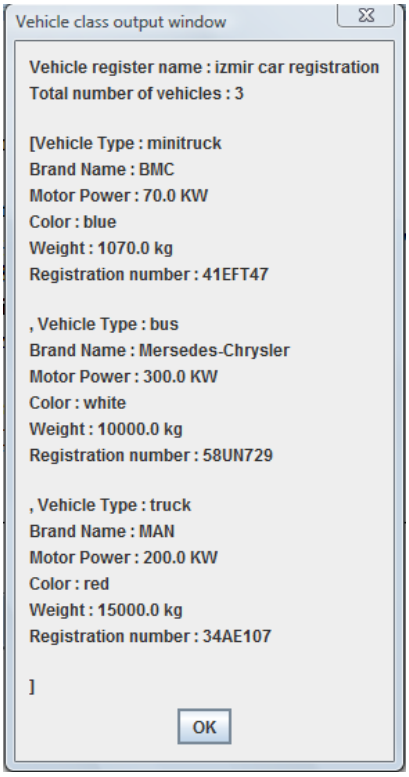
Program 14.7.3 vehicle_register class

```
İmport java.util.*;
public class vehicle_register
```

```
{
String name;
int number_of_vehicles;
HashMap<String,vehicle> v;
public vehicle_register(String kname)
{name=kname;
 number_of_vehicles=0;
 v=new HashMap<String,vehicle>();
}
 public String toString()
 {            String s="Vehicle register name : "+name+"\n";
    s+="Total number of vehicles : "+number_of_vehicles+"\n\n";
    s+=list();
    return s;
 }
public void add(vehicle b)
{
v.put(b.registration_number,b);
number_of_vehicles=v.size();
}
public void remove(vehicle b)
{v.remove(b.registration_number);
 number_of_vehicles=v.size();}
public String list()
{String s=v.values().toString();
return s;}
public String get(String regno)
{String s=v.get(regno).toString();
return s; }
}
```

14.7.4 vehicletest1 class

```
import javax.swing.*;
class vehicletest1
{
public static void outputwindow(String s)
{
String s1="Vehicle class output window";
JOptionPane.showMessageDialog(null,s,s1, JOptionPane.PLAIN_MESSAGE);
}
public static void main(String args[])
{
vehicle v1=new vehicle("truck","MAN",200.0,"red",15000,"34AE107");
vehicle v2=new vehicle("bus","Mersedes-Chrysler",300.0,"white",10000,"58UN729");
vehicle v3=new vehicle("minitruck","BMC",70,"blue",1070,"41EFT47");
vehicle_register vr=new vehicle_register("izmir car registration");
vr.add(v1);
vr.add(v2);
vr.add(v3);
String s=vr.toString()+"\n";
outputwindow(s);
}}
```



Methods of HashMap class is given belove:

Nested Class Summary

Nested classes/interfaces inherited from class java.util. AbstractMap

AbstractMap.SimpleEntry<K,V>, AbstractMap.SimpleImmutableEntry<K,V>

Constructor Summary

HashMap()
Constructs an empty HashMap with the default initial capacity (16) and the default load factor (0.75).

HashMap(int initialCapacity)
Constructs an empty HashMap with the specified initial capacity and the default load factor (0.75).

HashMap(int initialCapacity, float loadFactor)
Constructs an empty HashMap with the specified initial capacity and load factor.

HashMap(Map<? extends K,? extends V> m)
Constructs a new HashMap with the same mappings as the specified Map.

Method Summary

Void	clear() Removes all of the mappings from this map.
Object	clone() Returns a shallow copy of this HashMap instance: the keys and values themselves are not cloned.
Boolean	containsKey(Object key) Returns true if this map contains a mapping for the specified key.
Boolean	containsValue(Object value) Returns true if this map maps one or more keys to the specified value.
Set<Map.Entry<K,V>>	entrySet()

	Returns a Set view of the mappings contained in this map.
V	get(Object key) Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
Boolean	isEmpty() Returns true if this map contains no key-value mappings.
Set<K>	keySet() Returns a Set view of the keys contained in this map.
V	put(K key, V value) Associates the specified value with the specified key in this map.
Void	putAll(Map<? extends K,? extends V> m) Copies all of the mappings from the specified map to this map.
V	remove(Object key) Removes the mapping for the specified key from this map if present.
İnt	size() Returns the number of key-value mappings in this map.
Collection<V>	values() Returns a Collection view of the values contained in this map.

Methods inherited from class java.util. AbstractMap

equals, hashCode, toString

Methods inherited from class java.lang. Object

finalize, getClass, notify, notifyAll, wait, wait, wait

Methods inherited from interface java.util. Map

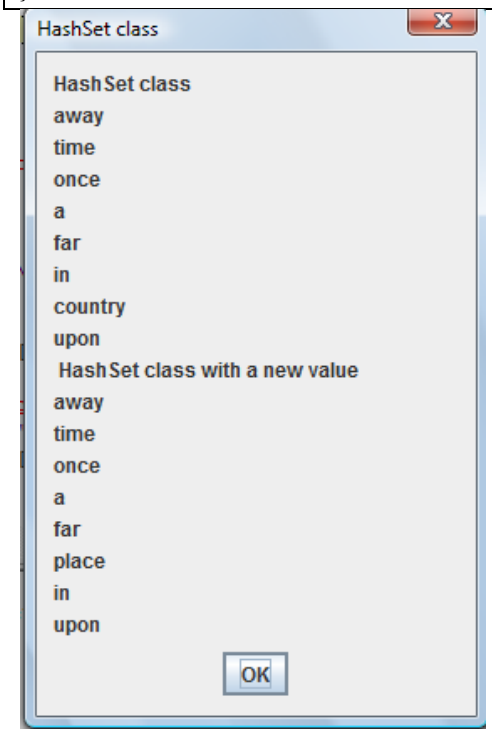
equals, hashCode

Another class to be introduced is HashSet. HashSet class is taken only one value anda a value can not be entered twice into a set (will be assumed the same value)

14.7.5 HasSet1 class

```
import java.util.*;
import javax.swing.*;
import java.util.*;
import javax.swing.*;

public class HashSet1
{
 public static String print(Collection c)
 {
  String s="";
  Iterator i=c.iterator();
  while(i.hasNext()) {s+=i.next()+"\n";}
  return s;
 }
 public static void main(String args[])
 {
 HashSet<String> a=new HashSet<String>();
 String s[]={"once","upon","a","time","in","a","country","far","far","away"};
```

```
  for(int i=0;i<s.length;i++) a.add(s[i]);

  String s1="HashSet class\n";
  String s2="HashSet class";;
  s1+=print(a);
  a.remove("country");
  a.add("place");
  s1+=" HashSet class with a new value\n";
  s1+=print(a);
  JOptionPane.showMessageDialog(null,s1,s2,
  JOptionPane.PLAIN_MESSAGE);
  }
}
```



Methods of HashSet class is given in the table belove

Constructor Summary

HashSet()
Constructs a new, empty set; the backing HashMap instance has default initial capacity (16) and load factor (0.75).

HashSet(Collection<? extends E> c)
Constructs a new set containing the elements in the specified collection.

HashSet(int initialCapacity)
Constructs a new, empty set; the backing HashMap instance has the specified initial capacity and default load factor, which is 0.75.

HashSet(int initialCapacity, float loadFactor)
Constructs a new, empty set; the backing HashMap instance has the specified initial capacity and the specified load factor.

Method Summary

boolean add(E o)
Adds the specified element to this set if it is not already present.

void clear()
Removes all of the elements from this set.

Object	clone() Returns a shallow copy of this HashSet instance: the elements themselves are not cloned.
boolean	contains(Object o) Returns true if this set contains the specified element.
boolean	isEmpty() Returns true if this set contains no elements.
Iterator<E>	iterator() Returns an iterator over the elements in this set.
boolean	remove(Object o) Removes the specified element from this set if it is present.
int	size() Returns the number of elements in this set (its cardinality).

Methods inherited from class java.util. AbstractSet

equals, hashCode, removeAll

Methods inherited from class java.util. AbstractCollection

addAll, containsAll, retainAll, toArray, toArray, toString

Methods inherited from class java.lang. Object

finalize, getClass, notify, notifyAll, wait, wait, wait

Methods inherited from interface java.util. Set

addAll, containsAll, equals, hashCode, removeAll, retainAll, toArray, toArray

14.8 DYNAMIC ARRAY STRUCTURES : NAVIGABLESET

NavigableSets are data structers that subsets can be define with easy comparisons. Methods lower, floor, ceiling, and higher return elements respectively less than, less than or equal, greater than or equal, and greater than a given element, returning null if there is no such element. A NavigableSet may be accessed and traversed in either ascending or descending order.

descendingIterator() and descendingSet()

The first interesting navigation methods are the `descendingIterator()` and `descendingSet()` methods.

The `descendingSet()` method returns a `NavigableSet` in which the order of the elements is reversed compared to this one. The returned "view" is backed by the original `NavigableSet`, so changes to the descending set are also reflected in the original set.

Here is a simple example:

NavigableSet reverse=original.descendingSet();

The `descendingIterator()` method allows you to iterate the elements of the `NavigableSet` (which is also a `SortedSet`) in reverse order, without changing the order of the elements internally.

Iterator reverse=original.descendingIterator();

headSet(), tailSet() and subSet()

The `headSet()` method returns a view of the original `NavigableSet` which only contains elements that are "less than" the given element. Here is an example:

```
NavigableSet original=new TreeSet();
Original.add("1");
Original.add("2");
Original.add("3");
//this headset will contain "1" and "2"
SortedSet headset=original.headSet("3");
//This headset will contain "1","2" and "3" because "inclusive"=true
NavigableSet headset=original.headSet("3",true);
```

The `tailSet()` method works the same way, except it returns all elements that are **higher** than the given parameter element.

The `subSet()` allows you to pass two parameters demarcating the boundaries of the view set to return. Here is an example:

```
NavigableSet original = new TreeSet();
original.add("1");
original.add("2");
original.add("3");
original.add("4");
original.add("5");

//this subset will contain "3"
SortedSet    subset  = original.subSet("2", "4");

//this subset will contain "2", "3" and "4" because
//    fromInclusive=true, and toInclusive=true
NavigableSet subset = original.subSet("2", true, "4", true);
```

ceiling(), floor(), higher(), and lower()

The `ceiling()` method returns the least (smallest) element in this set that is greater than or equal to the element passed as parameter to the `ceiling()` method. Here is an example:

```
NavigableSet original = new TreeSet();
original.add("1");
original.add("2");
original.add("3");

//ceiling will be "2".
Object ceiling = original.ceiling("2");

//floor will be "2".
Object floor = original.floor("2");
```

The `floor()` method does the opposite of `ceiling()`

The `higher()` method returns the least (smallest) element in this set that is greater than (not equal too) the element passed as parameter to the `higher()` method. Here is an example:

```
NavigableSet original = new TreeSet();
original.add("1");
original.add("2");
original.add("3");
//higher will be "3".
Object higher = original.higher("2");
//lower will be "1"
Object lower = original.lower("2");
```

The `lower()` method does the opposite of the `higher()` method.

pollFirst() and pollLast()

The `pollFirst()` method returns and removes the "first" element in the `NavigableSet` or null if the set is empty. The `pollLast()` returns and removes the "last" element in the set or null if the set is empty. "First" means smallest element according to the sort order of the set. "Last" means largest according to teh element sorting order of the set.

Here are two examples:

```
NavigableSet original = new TreeSet();
original.add("1");
original.add("2");
original.add("3");
//first is "1"
Object first = original.pollFirst();
//last is "3"
Object last = original.pollLast();
```

14.8.1 NavigableSetTest

```
import java.util.*;
import java.util.concurrent.*;

public class NavigableSetTest{
  public static void main(String[] args) {
    System.out.println("Navigable set Example!\n");
    NavigableSet <Integer>nSet = new ConcurrentSkipListSet<Integer>();
    nSet.add(10);
    nSet.add(20);
    nSet.add(50);
    nSet.add(30);
    nSet.add(100);
    nSet.add(80);
    // Returns an iterator over the elements in navigable set, in ascending order.
    Iterator iterator = nSet.iterator();
    System.out.print("Ascending order navigable set: ");
    //Ascending order list
    while (iterator.hasNext()){
      System.out.print(iterator.next() + " ");
    }
    System.out.println();
    //Descending order list
    System.out.println("Descending order navigable set: " + nSet.descendingSet() + "\n");
    //Greater than or equal to the given element
    System.out.println("Least element in Navigable set greater than or equal to 35: " + nSet.ceiling(35));
    //Less than or equal to the given element
```

```
    System.out.println("Greatest element in Navigable set less than or equal to 35: " + nSet.floor(35) + "\n");
    //Viewing the portion of navigable set whose elements are strictly less than the given element
    System.out.println("Navigable set whose elements are strictly less than '40': " + nSet.headSet(40));
    //Viewing the portion of navigable set whose elements are greater than or equal to the given element
    System.out.println("Navigable set whose elements are greater than or equal to '40': " + nSet.tailSet(40) + "\n");
    //Removing first element from navigable set
    System.out.println("Remove element: "+nSet.pollFirst());
    //After removing the first element, now get navigable set
    System.out.println("Now navigable set: " + nSet.descendingSet() + "\n");
    //Removing last element from navigable set
    System.out.println("Remove element: " + nSet.pollLast());
    //After removing the last element, now get navigable set
    System.out.println("Now navigable set: " + nSet.descendingSet());
  }
}
```

The result will be;

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" NavigableSetTest
Navigable set Example!

Ascending order navigable set: 10 20 30 50 80 100
Descending order navigable set: [100, 80, 50, 30, 20, 10]

Least element in Navigable set greater than or equal to 35: 50
Greatest element in Navigable set less than or equal to 35: 30

Navigable set whose elements are strictly less than '40': [10, 20, 30]
Navigable set whose elements are greater than or equal to '40': [50, 80, 100]

Remove element: 10
Now navigable set: [100, 80, 50, 30, 20]

Remove element: 100
Now navigable set: [80, 50, 30, 20]

> Terminated with exit code 0.
```

The methods of the navigableSet as follows

Method Summary

E	ceiling(E e) Returns the least element in this set greater than or equal to the given element, or null if there is no such element.
Iterator<E>	descendingIterator() Returns an iterator over the elements in this set, in descending order.
NavigableSet<E>	descendingSet() Returns a reverse order view of the elements contained in this set.
E	floor(E e) Returns the greatest element in this set less than or equal to the given element, or null if there is no such element.
SortedSet<E>	headSet(E toElement) Returns a view of the portion of this set whose elements are strictly less than toElement.
NavigableSet<E>	headSet(E toElement, boolean inclusive) Returns a view of the portion of this set whose elements are less than (or equal to, if inclusive is

	true) toElement.
E	higher(E e) Returns the least element in this set strictly greater than the given element, or null if there is no such element.
Iterator<E>	iterator() Returns an iterator over the elements in this set, in ascending order.
E	lower(E e) Returns the greatest element in this set strictly less than the given element, or null if there is no such element.
E	pollFirst() Retrieves and removes the first (lowest) element, or returns null if this set is empty.
E	pollLast() Retrieves and removes the last (highest) element, or returns null if this set is empty.
NavigableSet<E>	subSet(E fromElement, boolean fromInclusive, E toElement, boolean toInclusive) Returns a view of the portion of this set whose elements range from fromElement to toElement.
SortedSet<E>	subSet(E fromElement, E toElement) Returns a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive.
SortedSet<E>	tailSet(E fromElement) Returns a view of the portion of this set whose elements are greater than or equal to fromElement.
NavigableSet<E>	tailSet(E fromElement, boolean inclusive) Returns a view of the portion of this set whose elements are greater than (or equal to, if inclusive is true) fromElement.

Methods inherited from interface java.util. SortedSet

comparator, first, last

Methods inherited from interface java.util. Set

add, addAll, clear, contains, containsAll, equals, hashCode, isEmpty, remove, removeAll, retainAll, size, toArray, toArray

14.9 A DYNAMIC ARRAY STRUCTURES : AN APPLICATION PROGRAM USING ARRAYLIST TO STORE DATA

Ideal gas is a defination of the gases which obeys ideal gas law Pv=R*T, where P is the pressure(bar), T is the temperature(degree Kelvin), v is the specific volume(m^3/kg), R is the gas constant (J/(kg K). R can be calculated from universal gas constant R0 and molecular weight m(kg/kmol) R=R0/M. Where R0=8314.5 J/(kmol K). In the following table specific heat of the gases $Cp(T) = a + bT + cT^2 + dT^3$ where a,b,c and d is constants, The unit of the Cp is kJ/(kmol K).

Two other important thermodynamic properties are enthalpy (h) and entrophy (s) which defines as

As it is seen from the problem, basic data should be input into the program in order to do calculations. ArrayList dynamic data structure is used to enter data to the program. In order to establish that idealgasData class is defined as:

14.7.1 idealgasData class

```
Public class idealgasData extends Object
{
public String name;   //name of the ideal gas
public String formula; //ideal gas formula
public double M;     // ideal gaz molekular weight
public double A[];   // Cp(T)=A[0]+A[1]*T+A[2]*T*T+A[3]*T*T*T

  public idealgasData(String namei,String formulai,double Mi,double a,double b,double c,double d)
  {name=namei;;
   formula=formulai;
   M=Mi;
   A=new double[4];
   A[0]=a;
   A[1]=b;
   A[2]=c;
   A[3]=d;
  }
  public String toString()
  {
  String s="";
  s+="ideal gas name = "+name+"\n";
  s+="ideal gas formula = "+formula+"\n";
  s+="ideal gas molecular weight = "+M+" kg/kmol \n";
  s+="Cp(T)=A[0]+A[1]*T+A[2]*T*T+A[3]*T*T*T coefficients\n";
  for(int i=0;i<4;i++)
  s+="A["+i+"] = "+A[i];
  return s;
  }
}
```

Now , and idealgas class can be defined. Ideal gas data is read into idealgasData class and collected as a vector in an ArrayList (see idealgasInput method)

14.7.2 idealgas class

```
//=====================================================
// java Thermodynamic package
// Class idealgas properties of the ideal gas
// Dr. Turhan Coban
// Ege Üniversitesi Mühendislik fakultesi
```

```
// Makina Bölümü
//
//=====================================================
// File name    : idealgas.java
// This class calculates thermodynamic properties of the ideal gas
//
// ====================================================

import java.io.*;
import java.net.*;
import java.util.*;

public class idealgas
{
public idealgasData rd;
static double T0=298.15; // derece K
static double P0=1; //1 bar
public static double R=8.3145; //KJ/kgK

public idealgas(String igName)
{idealgasInput(igName);}

public String[] idealgaslist()
{ String s1=rd.name;String s2=idealgasInput("null");idealgasInput(s1);
  StringTokenizer t=new StringTokenizer(s2);
  int n=t.countTokens();
  String s3[]=new String[n];
  int i=0;
  while(t.hasMoreTokens())
  {
  s3[i]=t.nextToken();
  i++;
  }
  return s3;
}

public double Cp(double t)
{ double T=t+273.15;
  return (rd.A[0]+rd.A[1]*T+rd.A[2]*T*T+rd.A[3]*T*T*T);
  //KJ/kmol K
}

public double h(double t)
{ double T=t+273.15;
  double hi=rd.A[0]*(T-T0)+rd.A[1]/2.0*(T*T-T0*T0)+
  rd.A[2]/3.0*(T*T*T-T0*T0*T0)+rd.A[3]/4.0*(T*T*T*T-T0*T0*T0*T0);
  //KJ/kmol
  return hi;
}

public double s0(double t)
{ double T=t+273.15;
  double si=rd.A[0]*Math.log(T/T0)+rd.A[1]*(T-T0)+
  rd.A[2]/2.0*(T*T-T0*T0)+rd.A[3]/3.0*(T*T*T-T0*T0*T0);
  return si;
}

public double s(double t,double P)
```

```
{ double si=s0(t)-R*Math.log(P/P0);
  return si;
}
public String idealgasInput(String ig_name)
{
    String s="";
    ArrayList<idealgasData> gi = new ArrayList<idealgasData>();
    rd = new idealgasData("nitrogen","N2",28.013,28.90,-0.1571e-2,0.8081e-5,-2.873e-9);
    gi.add(rd);
    rd=new idealgasData("oxygen","O2",31.999,25.48,1.52e-2,-0.7155e-5,1.312e-9);
    gi.add(rd);
    rd=new idealgasData("carbondioxide","CO2",44.01,22.26,5.981e-2,-3.501e-5,7.469e-9);
    gi.add(rd);
    rd=new idealgasData("carbonmonoxide","CO",28.011,28.16,0.1675e-2,0.5372e-5,-2.222e-9);
    gi.add(rd);
    rd=new idealgasData("hydrogen","H2",2.016,29.11,-0.1916e-2,0.4003e-5,-0.8704e-9);
    gi.add(rd);
    rd=new idealgasData("air","0.21O2 0.79N2",28.97,28.11,0.1967e-2,0.4802e-5,-1.966e-9);
    gi.add(rd);
    rd=new idealgasData("steam","H2O",18.015,32.24,0.1923e-2,1.055e-5,-3.595e-9);
    gi.add(rd);
    rd=new idealgasData("nitrogenoxide","NO",30.006,29.34,-0.09395e-2,0.9747e-5,-4.187e-9);
    gi.add(rd);
    rd = new idealgasData("methane","CH4",16.043,19.89,5.024e-2,1.269e-5,-11.01e-9);
    gi.add(rd);
    rd = new idealgasData("ethane","C2H6",30.07,6.90,17.27e-2,-6.406e-5,7.285e-9);
    gi.add(rd);
    rd = new idealgasData("propane","C3H8",44.097,-4.04,30.48e-2,-15.72e-5,31.74e-9);
    gi.add(rd);
    rd = new idealgasData("n-butane","C4H10",58.124,3.96,37.15e-2,-18.34e-5,35e-9);
    gi.add(rd);
    rd = new idealgasData("n-pentane","C5H12",72.151,6.774,45.43e-2,-22.46e-5,42.29e-9);
    gi.add(rd);
    rd = new idealgasData("n-hexane","C6H14",86.178,6.938,55.22e-2,-28.65e-5,57.69e-9);
    gi.add(rd);
    rd = new idealgasData("dinitrogenoxide","N2O",44.013,22.9,5.715e-2,-3.52e-5,7.87e-9);
    gi.add(rd);
    rd = new idealgasData("ammonia","NH3",17.03,27.568,2.563e-2,0.99072e-5,-6.6909e-9);
    gi.add(rd);
    rd = new idealgasData("sulphur","S",32.064,27.21,2.218e-2,-1.628e-2,3.986e-9);
    gi.add(rd);
    rd = new idealgasData("sulphurdioxide","SO2",64.063,25.78,5.795e-2,-3.812e-5,8.612e-9);
    gi.add(rd);
    rd = new idealgasData("sulphurtrioxide","SO3",80.0625,16.40,14.58e-2,-11.20e-5,32.42e-9);
    gi.add(rd);
    rd = new idealgasData("asethylene","C2H2",26.038,21.8,9.2143e-2,-6.527e-5,18.21e-9);
    gi.add(rd);
    rd = new idealgasData("ethylene","C2H4",28.054,3.95,15.64e-2,-8.344e-5,17.67e-9);
    gi.add(rd);
    rd = new idealgasData("propylene","C3H6",42.081,3.15,23.83e-2,-12.18e-5,24.62e-9);
    gi.add(rd);
    rd = new idealgasData("benzene","C6H6",78.115,-36.22,48.475e-2,-31.57e-5,77.62e-9);
    gi.add(rd);
    rd = new idealgasData("methanol","CH3OH",32.042,19.0,9.152e-2,-1.22e-5,-8.039e-9);
    gi.add(rd);
    rd = new idealgasData("ethanol","C2H5OH",46.07,19.9,20.96e-2,-10.38e-5,20.05e-9);
    gi.add(rd);
    rd = new idealgasData("hidrogenchloride","HCl",36.461,30.33,-0.762e-2,1.327e-5,-4.338e-9);
```

```
    gi.add(rd);
    int i;
    Iterator n1=gi.iterator();
    while(n1.hasNext())
    {   rd=(idealgasData)n1.next();
    s+=rd.name+" ";
    if ((rd.name.equals(ig_name)) || (rd.formula.equals(ig_name)))   {break;}
    }
    return s;
    }
public String[][] output(double t,double P)
{double a[]=new double[6];
 String b[][]=new String[6][3];
 a[0]=+t;
 a[1]=P;
 a[2]=Cp(t)/rd.M;
 a[3]=h(t)/rd.M;
 a[4]=s0(t)/rd.M;
 a[5]=s(t,P)/rd.M;
 b[0][0]="  temperature, t ";
 b[1][0]="  pressure, P ";
 b[2][0]="  specific heat at constant pressure, Cp ";
 b[3][0]="  enthalpy h(t) ";
 b[4][0]="  entrophy s0(t) ";
 b[5][0]="  entrophy s(t,P)  ";
 b[0][2]="  degree C";
 for(int i=0;i<a.length;i++) b[i][1]=""+a[i];
 b[1][2]="  bar";
 b[2][2]="  KJ/kg K";
 b[3][2]="  KJ/kg";
 b[4][2]="  KJ/kg K";
 b[5][2]="  KJ/kg K";
  return b;
}
}
```

İdealgastest class is created to check if the class is working as it should be

14.7.3 idealgastest class

```
public class idealgastest
{
  public static void main(String arg[])
  {
  idealgas O2=new idealgas("air");
  System.out.println(O2.rd.toString());

  for(double t=(270-273);t<=100;t+=10)
  {System.out.println("t = "+t+" Cp = "+
  O2.Cp(t)+"h = "+O2.h(t)+" s = "+O2.s(t,1.0));}
  String s[]=O2.idealgaslist();

  System.out.println("Gas names in the list : ");
  for(int i=0;i<s.length;i++) System.out.println(s[i]);
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" idealgastest
```

```
ideal gas name = air
ideal gas formula = 0.21O2 0.79N2
ideal gas molecular weight = 28.97 kg/kmol
Cp(T)=A[0]+A[1]*T+A[2]*T*T+A[3]*T*T*T coefficients
A[0] = 28.11A[1] = 0.001967A[2] = 4.802E-6A[3] = -1.966E-9
t = -3.0 Cp = 28.953078611578015h = -812.3287606046259 s = -2.8610235100083043
t = 7.0 Cp = 28.99470829640096h = -522.5903559256143 s = -1.8078923370305282
t = 17.0 Cp = 29.036967916283917h = -232.43249489307325 s = -0.7902301629983612
t = 27.0 Cp = 29.079845675226867h = 58.15106286359728 s = 0.19438833312726914
t = 37.0 Cp = 29.123329777229813h = 349.1664397549974 s = 1.1481495568466065
t = 47.0 Cp = 29.167408426292766h = 640.6196402317267 s = 2.0730313357801218
t = 57.0 Cp = 29.212069826415714h = 932.516550784386 s = 2.9708285897503064
t = 67.0 Cp = 29.257302181598664h = 1224.8629399435747 s = 3.843175169374883
t = 77.0 Cp = 29.303093695841614h = 1517.6644582798926 s = 4.691562529586142
t = 87.0 Cp = 29.349432573144565h = 1810.9266384039404 s = 5.517355773063849
t = 97.0 Cp = 29.396307017507517h = 2104.6548949663174 s = 6.321807495953667
Gas names in the list :
nitrogen
oxygen
carbondioxide
carbonmonoxide
hydrogen
air
steam
nitrogenoxide
methane
ethane
propane
n-butane
n-pentane
n-hexane
dinitrogenoxide
ammonia
sulphur
sulphurdioxide
sulphurtrioxide
asethylene
ethylene
propylene
benzene
methanol
ethanol
hidrogenchloride

> Terminated with exit code 0.
```

Now it is seen that class is working properly, a human graphical user interface can be prepared for a better output. Two classes will be preapred. The first class idealGasModel is prepared to get table output format. The main output program is idealgasTable

14.7.4 idealgasModel class

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
```

```
class idealgasModel extends AbstractTableModel
{
  Object[][] veri;
  String[] baslik={"Name","Value","Units"};
  idealgas g;

  public idealgasModel()
  {
  g=new idealgas("air");
  veri=g.output(25.0,1.01325);
  }
  public idealgasModel(String idealgasname,double t,double P)
  {setValues(idealgasname,t,P);}
  public idealgasModel(idealgas gi,double t,double P)
  { setValues(gi,t,P);}
  public int getRowCount() {return veri.length;}
  public int getColumnCount() {return baslik.length;}
  public Object getValueAt(int satir,int sutun) {return veri[satir][sutun];}
  public String getColumnName(int c) {return baslik[c];}
  public void setValueAt(Object val, int row, int col)
  {
    veri[row][col] = val;
   }

  public void setValues(String idealgasname,double t,double P)
  {
  g=new idealgas(idealgasname);
  veri=g.output(t,P);
  }
  public void setValues(idealgas g1,double t,double P)
  {
  g=g1;
  veri=g.output(t,P);
  }

  public boolean isCellEditable(int row, int col) {return true;}
}
```

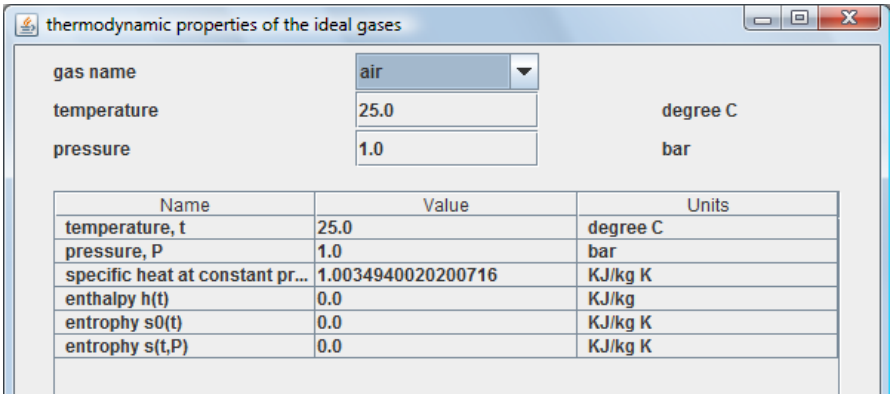
14.7.5 idealgasTable class

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;

public class idealgasTable extends JApplet implements ActionListener,ItemListener,Serializable
{
  JLabel l1[]=new JLabel[3];          // Label
  JLabel l2[]=new JLabel[3];          // Label
  JTextField t1[]=new JTextField[2];   // input fields
  JComboBox c1[]=new JComboBox[1];     // selection field
  JTextArea outputTextArea;
  idealgasModel gm;
```

```
public idealgas g1;
JTable jt;
double T;
double P;
public String gazismi;
protected StringTokenizer token;
String st[];
Color bg;
Color fg;
JPanel inputPanel;
JPanel  outputPanel;
JPanel  totalPanel;
JScrollPane panel1;
public void init()
{
  Container c=getContentPane();
  c.setLayout(new FlowLayout());
  bg=c.getBackground();
  fg=c.getForeground();
  gazismi=new String("N2");
  T=25.0;
  P=1.0;
  inputPanel=new JPanel();
  inputPanel.setLayout(new GridLayout(3,3,80,1));
  outputPanel=new JPanel();
  outputPanel.setLayout(new BorderLayout());
  totalPanel=new JPanel();
  totalPanel.setLayout(new BorderLayout());
  g1=new idealgas(gazismi);
  String s1=g1.idealgasInput("null");
  token=new StringTokenizer(s1);
  st=new String[token.countTokens()];
  int i=0;
  while(token.hasMoreTokens())
  {st[i++]=new String((String)token.nextToken());}
  c1[0]=new JComboBox(st);
  t1[0] = new JTextField();
  t1[1] = new JTextField();
  g1=new idealgas(gazismi);
  c1[0].setSelectedItem(gazismi);
  l1[0]=new JLabel("gas name        ");
  l1[1]=new JLabel("temperature            ");
  l1[2]=new JLabel("pressure               ");
  l2[0]=new JLabel("                  ");
  l2[1]=new JLabel(" degree C              ");
  l2[2]=new JLabel(" bar                   ");
  outputTextArea=new JTextArea();
  Font fn=l1[0].getFont();
  for(i=0;i<2;i++)
  {    l1[i].setBackground(Color.lightGray);
       l2[i].setBackground(Color.lightGray);
  }
  t1[0]=new JTextField(""+T);
  t1[1]=new JTextField(""+P);
  for(i=0;i<2;i++)
  {    t1[i].setBackground(bg);t1[i].setFont(fn); }
       c1[0].setBackground(bg);
  outputTextArea.setBackground(bg);
```

```
    outputTextArea.setForeground(fg);
    outputTextArea.setFont(fn);
    inputPanel.add(l1[0]);
    inputPanel.add(c1[0]);
    inputPanel.add(l2[0]);
    inputPanel.add(l1[1]);
    inputPanel.add(t1[0]);
    inputPanel.add(l2[1]);
    inputPanel.add(l1[2]);
    inputPanel.add(t1[1]);
    inputPanel.add(l2[2]);
    c1[0].addItemListener(this);
    t1[0].addActionListener(this);
    t1[1].addActionListener(this);
    double TK=T+273.0;
    gm=new idealgasModel(g1,T,P);
    jt=new JTable(gm);
    jt.setBackground(bg);
    jt.setFont(fn);
    outputPanel.add(outputTextArea,BorderLayout.NORTH);
    panel1=new JScrollPane(jt);
    outputPanel.add(panel1,BorderLayout.SOUTH);
    totalPanel.add(inputPanel,BorderLayout.NORTH);
    totalPanel.add(outputPanel,BorderLayout.SOUTH);
    c.add(totalPanel,BorderLayout.NORTH);
  }
  public void itemStateChanged(ItemEvent ev)
  {
  String st1=(String)c1[0].getSelectedItem();
  gazismi=st1;
  gm.setValues(gazismi,T,P);
  repaint();
  }
public void actionPerformed( ActionEvent e)
{
  Double valTemp=new Double(t1[0].getText());
  T=valTemp.doubleValue();
  Double valPressure=new Double(t1[1].getText());
  P=valPressure.doubleValue();
  gm.setValues(gazismi,T,P);
  repaint();
}
        public static void main(String s[]) {
        JFrame f = new JFrame("thermodynamic properties of the ideal gases ");
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {System.exit(0);}
        });
        JApplet applet = new idealgasTable();
        f.getContentPane().add("Center", applet);
        applet.init();
        f.pack();
        f.setSize(new Dimension(600,400));
        f.setVisible(true);
    }
}
```

Output of the class:



Data of the program is taken from the following tables:

TABLE A – 1

Molar mass, gas constant, and critical-point properties

Substance	Formula	Molar mass, M kg/kmol	Gas constant, R kJ/kg · K*	Critical-point properties: Temperature, K	Pressure, MPa	Volume, m³/kmol
Air	—	28.97	0.2870	132.5	3.77	0.0883
Ammonia	NH_3	17.03	0.4882	405.5	11.28	0.0724
Argon	Ar	39.948	0.2081	151	4.86	0.0749
Benzene	C_6H_6	78.115	0.1064	562	4.92	0.2603
Bromine	Br_2	159.808	0.0520	584	10.34	0.1355
n-Butane	C_4H_{10}	58.124	0.1430	425.2	3.80	0.2547
Carbon dioxide	CO_2	44.01	0.1889	304.2	7.39	0.0943
Carbon monoxide	CO	28.011	0.2968	133	3.50	0.0930
Carbon tetrachloride	CCl_4	153.82	0.05405	556.4	4.56	0.2759
Chlorine	Cl_2	70.906	0.1173	417	7.71	0.1242
Chloroform	$CHCl_3$	119.38	0.06964	536.6	5.47	0.2403
Dichlorodifluoromethane (R-12)	CCl_2F_2	120.91	0.06876	384.7	4.01	0.2179
Dichlorofluoromethane (R-21)	$CHCl_2F$	102.92	0.08078	451.7	5.17	0.1973
Ethane	C_2H_6	30.070	0.2765	305.5	4.48	0.1480
Ethyl alcohol	C_2H_5OH	46.07	0.1805	516	6.38	0.1673
Ethylene	C_2H_4	28.054	0.2964	282.4	5.12	0.1242
Helium	He	4.003	2.0769	5.3	0.23	0.0578
n-Hexane	C_6H_{14}	86.179	0.09647	507.9	3.03	0.3677
Hydrogen (normal)	H_2	2.016	4.1240	33.3	1.30	0.0649
Krypton	Kr	83.80	0.09921	209.4	5.50	0.0924
Methane	CH_4	16.043	0.5182	191.1	4.64	0.0993
Methyl alcohol	CH_3OH	32.042	0.2595	513.2	7.95	0.1180
Methyl chloride	CH_3Cl	50.488	0.1647	416.3	6.68	0.1430
Neon	Ne	20.183	0.4119	44.5	2.73	0.0417
Nitrogen	N_2	28.013	0.2968	126.2	3.39	0.0899
Nitrous oxide	N_2O	44.013	0.1889	309.7	7.27	0.0961
Oxygen	O_2	31.999	0.2598	154.8	5.08	0.0780
Propane	C_3H_8	44.097	0.1885	370	4.26	0.1998
Propylene	C_3H_6	42.081	0.1976	365	4.62	0.1810
Sulfur dioxide	SO_2	64.063	0.1298	430.7	7.88	0.1217
Tetrafluoroethane (R-134a)	CF_3CH_2F	102.03	0.08149	374.2	4.059	0.1993
Trichlorofluoromethane (R-11)	CCl_3F	137.37	0.06052	471.2	4.38	0.2478
Water	H_2O	18.015	0.4615	647.1	22.06	0.0560
Xenon	Xe	131.30	0.06332	289.8	5.88	0.1186

*The unit kJ/kg · K is equivalent to kPa · m³/kg · K. The gas constant is calculated from $R = R_u/M$, where R_u = 8.31447 kJ/kmol · K and M is the molar mass.

Source: K. A. Kobe and R. E. Lynn, Jr., *Chemical Review* 52 (1953), pp. 117–236; and ASHRAE, *Handbook of Fundamentals* (Atlanta, GA: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc., 1993), pp. 16.4 and 36.1.

TABLE A–2

Ideal-gas specific heats of various common gases (*Concluded*)

(*c*) As a function of temperature

$$\bar{c}_p = a + bT + cT^2 + dT^3$$

(T in K, c_p in kJ/kmol · K)

Substance	Formula	a	b	c	d	Temperature range, K	% error Max.	% error Avg.
Nitrogen	N_2	28.90	-0.1571×10^{-2}	0.8081×10^{-5}	-2.873×10^{-9}	273–1800	0.59	0.34
Oxygen	O_2	25.48	1.520×10^{-2}	-0.7155×10^{-5}	1.312×10^{-9}	273–1800	1.19	0.28
Air	—	28.11	0.1967×10^{-2}	0.4802×10^{-5}	-1.966×10^{-9}	273–1800	0.72	0.33
Hydrogen	H_2	29.11	-0.1916×10^{-2}	0.4003×10^{-5}	-0.8704×10^{-9}	273–1800	1.01	0.26
Carbon monoxide	CO	28.16	0.1675×10^{-2}	0.5372×10^{-5}	-2.222×10^{-9}	273–1800	0.89	0.37
Carbon dioxide	CO_2	22.26	5.981×10^{-2}	-3.501×10^{-5}	7.469×10^{-9}	273–1800	0.67	0.22
Water vapor	H_2O	32.24	0.1923×10^{-2}	1.055×10^{-5}	-3.595×10^{-9}	273–1800	0.53	0.24
Nitric oxide	NO	29.34	-0.09395×10^{-2}	0.9747×10^{-5}	-4.187×10^{-9}	273–1500	0.97	0.36
Nitrous oxide	N_2O	24.11	5.8632×10^{-2}	-3.562×10^{-5}	10.58×10^{-9}	273–1500	0.59	0.26
Nitrogen dioxide	NO_2	22.9	5.715×10^{-2}	-3.52×10^{-5}	7.87×10^{-9}	273–1500	0.46	0.18
Ammonia	NH_3	27.568	2.5630×10^{-2}	0.99072×10^{-5}	-6.6909×10^{-9}	273–1500	0.91	0.36
Sulfur	S_2	27.21	2.218×10^{-2}	-1.628×10^{-5}	3.986×10^{-9}	273–1800	0.99	0.38
Sulfur dioxide	SO_2	25.78	5.795×10^{-2}	-3.812×10^{-5}	8.612×10^{-9}	273–1800	0.45	0.24
Sulfur trioxide	SO_3	16.40	14.58×10^{-2}	-11.20×10^{-5}	32.42×10^{-9}	273–1300	0.29	0.13
Acetylene	C_2H_2	21.8	9.2143×10^{-2}	-6.527×10^{-5}	18.21×10^{-9}	273–1500	1.46	0.59
Benzene	C_6H_6	−36.22	48.475×10^{-2}	-31.57×10^{-5}	77.62×10^{-9}	273–1500	0.34	0.20
Methanol	CH_4O	19.0	9.152×10^{-2}	-1.22×10^{-5}	-8.039×10^{-9}	273–1000	0.18	0.08
Ethanol	C_2H_6O	19.9	20.96×10^{-2}	-10.38×10^{-5}	20.05×10^{-9}	273–1500	0.40	0.22
Hydrogen chloride	HCl	30.33	-0.7620×10^{-2}	1.327×10^{-5}	-4.338×10^{-9}	273–1500	0.22	0.08
Methane	CH_4	19.89	5.024×10^{-2}	1.269×10^{-5}	-11.01×10^{-9}	273–1500	1.33	0.57
Ethane	C_2H_6	6.900	17.27×10^{-2}	-6.406×10^{-5}	7.285×10^{-9}	273–1500	0.83	0.28
Propane	C_3H_8	−4.04	30.48×10^{-2}	-15.72×10^{-5}	31.74×10^{-9}	273–1500	0.40	0.12
n-Butane	C_4H_{10}	3.96	37.15×10^{-2}	-18.34×10^{-5}	35.00×10^{-9}	273–1500	0.54	0.24
i-Butane	C_4H_{10}	−7.913	41.60×10^{-2}	-23.01×10^{-5}	49.91×10^{-9}	273–1500	0.25	0.13
n-Pentane	C_5H_{12}	6.774	45.43×10^{-2}	-22.46×10^{-5}	42.29×10^{-9}	273–1500	0.56	0.21
n-Hexane	C_6H_{14}	6.938	55.22×10^{-2}	-28.65×10^{-5}	57.69×10^{-9}	273–1500	0.72	0.20
Ethylene	C_2H_4	3.95	15.64×10^{-2}	-8.344×10^{-5}	17.67×10^{-9}	273–1500	0.54	0.13
Propylene	C_3H_6	3.15	23.83×10^{-2}	-12.18×10^{-5}	24.62×10^{-9}	273–1500	0.73	0.17

Source: B. G. Kyle, *Chemical and Process Thermodynamics* (Englewood Cliffs, NJ: Prentice-Hall, 1984). Used with permission.

WEEK 2(chapter 14) CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) StringTokenizer2.java program is given belove as uncompleted. Enter a long String variable with words seperated by empty space such as : String s="Once upon a time in a country far far away";

Then separete the String by using **public static String[] enterStringarray(String s)** method. Printout the seperated String array.

```
import java.io.*;

import java.util.*;
import javax.swing.*;

public class StringTokenizerTest2
{
    public static String[] enterStringarray(String s)
    {
    String s1=JOptionPane.showInputDialog(s);
```

```
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    String a[]=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    {a[j++]=token.nextToken();}
    return a;
    }

  public static void main(String arg[])
  {
  }
}
```

EX2) One of the method of StringBuffer is replace as seen belove

StringBuffer	replace(int start, int end, String str) Replaces the characters in a substring of this sequence with characters in the specified String.

Take the String String s="Once upon a time, in a country far far away";
Write a method to change every empty space charecter in the input String with character *,

```
public static String replace_empty_with_star(String source)
{
}
```

print the output String.

HOMEWORK 2(chapter 14)

HW1) The following String is given :

String s= "Integer variables can be called as byte(8 bit), short(16 bit), int(32 bit) and long(64 bit) according to ";
s+="space they occupy in computer memory. Bits are binary structures that only contains 0 or 1. When bits ";
s+="coming together as byte groups of different sizes they represents integer numbers. For example maximum ";
s+="number can be written to 8 bit byte type of integer variable is 11111111 which represents 256. Usually one ";
s+="of the bits used as +/- sign in a number therefore byte carries numbers between -128 to 127 (total ";
s+="256 including 0) . ın order to define an integer variable, first variable type defined and a variable";
s+=" name follows. An actual value can be assigned by using an = (equal) sign.";

write a program to count the words, calculate the avarage word length, maximum word length and minimum word length.

HW2) Take the String Array created from the string given in homework 1, load the array element into ArrayList dydnamic structure, sort the array and print the result.

WEEK 3(chapter 14) CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX3.1) HashMap2.java is given list of student identification numbers and strudent names as two String arrays. Check out exercise **HashMap1.java**, and similarly load student data into HashMap class, give your student identification number and get your name as an output

EX3.2) The same list in EX1 is given in ThreeMap2.java. Check out exersize **TreeMapTes**t Add Student identification numbers and student names. Add the data as a ThreeMap, and then list the data

WEEK 3 HOMEWORK ASSIGNMENTS

HW 3.1 One of the program investigated last semester to understand array structure usage was the hotel program. This program is using the array structure **double guess[];** to store the number of guess information in each room, but as room numbers it is using the array indices 0,1,2,… Could you change to program into

HashMap<Integer,Integer> structure so that the first number will store the actual room numbers of the hotel room and the second number will be number of guess in that room

```
public class hotel
 {
 String hotel_name;
 int number_of_rooms;
 double guess[];

 public hotel(String i_hotel_name,int i_number_of_rooms)
 {   hotel_name=i_hotel_name;
             number_of_rooms=i_number_of_rooms;
             guess=new double[number_of_rooms];
 }

 public hotel(String i_hotel_name,double i_guess[])
 {   hotel_name=i_hotel_name;
             number_of_rooms=i_guess.length;
             guess=new double[number_of_rooms];
             for(int room_number=0;room_number<number_of_rooms;room_number++)
             {guess[room_number]=i_guess[room_number];}
 }

 public void room_registration(int room_number,int number_of_customers)
 { guess[room_number]=number_of_customers;}

 public void room_registration(double i_guess[])
 {           int n=i_guess.length;
             for(int room_number=0;room_number<n;room_number++)
             {guess[room_number]=i_guess[room_number];}
 }
 public int number_of_guess()
 {           int nog=0;
             for(int room_number=0;room_number<number_of_rooms;room_number++)
             {nog+=guess[room_number];}
             return nog;
 }
 public String toString()
 {           String s="Hotel name : "+hotel_name+"\n";
    s+="Total number of guess : "+number_of_guess()+"\n";
             for(int room_number=0;room_number<number_of_rooms;room_number++)
             {s+="room number = "+room_number+" number of guess = "+guess[room_number]+"\n";}
    return s;
 }

 public String name(){return hotel_name;}
}
```

```
import javax.swing.*;

class hoteltest1
{
  public static void main(String args[])
  {
  hotel h1=new hotel("long view",5);
  h1.room_registration(0,3);
  h1.room_registration(1,2);
  double a[]={1,3,4,5,1,2,3,6};
```

```
  hotel h2=new hotel("nice beach",a);
  hotel h3=new hotel("blue point",10);
  double b[]={2,1,2,3,1,2,3};
  h3.room_registration(b);
  h3.room_registration(8,3);
  String s=h1.toString()+"\n "+
          h2.toString()+"\n "+
          h3.toString()+"\n ";
  JOptionPane.showMessageDialog(null,s,
    "Hotel registry list",JOptionPane.PLAIN_MESSAGE);
    System.exit(0);}
}
```

HW3.2) NavigableSet1.java program is given. In this set student list strictly smaller than SERCAN BALTA is required. Add the needed command to the programming code to achive this.

```
import java.util.*;
import javax.swing.*;
import java.util.concurrent.*;

public class NavigatableSet1
{
public static void main(String args[])
{
String student_name[]={"MERT          KULA",
"İLYAS  BAYRAKTAR",
"SAMET              YILDIZ",
"NURİ    HARMANKAYA",
"GÜNAY              DÜRMÜŞ ",
"CAN     ORAL ",
"EMRE   ERDOĞAN ",
"ÇAĞLAR            ASARLI ",
"RAMAZAN         ORAL",
"DOVLETGELDI  ANNAGURBANOV",
"AHMET ULAŞ    BAYKAL",
"MURAT              KARAKURT",
"SELMAN            ÜNAL",
"MERT   KOÇER",
"BURAK CAN      ÇELENK",
"MEHMET AKİF  SAÇIKARA",
"AHMET              NALÇABASMAZ",
"AKİF     ÖZEN",
"TANER              IŞIK",
"DİLEK  AKTÜRK",
"MUHAMMET CANER      MARANGOZ",
"MEHMET            ATEŞ",
"HİKMET             BAYOĞLU",
"SONER              AŞIK",
"MEHMET MURAT           YARDIMCI",
"MEKAN              CHARIYEV",
"VURAL              ASLAN",
"Cihan    AŞKIN",
"Volkan  KORKUT",
"Selçuk   MUTLU",
"Serdar   DEMİR",
"Can       UYSAL",
"Gürhan  ÇOBAN",
```

"Erdal Şahin ACAR",
"Hüseyin DOĞANCAN",
"Ahmet Mert KILIÇ",
"Hüseyin ALSOY",
"Gözde BERKAY",
"Başak ŞİLEN",
"Burak MOKAN",
"Emel KIVRAK",
"Koray SARKAN",
"Uğur ÖZKAN",
"Yasin CAN",
"Bahtiyar ARSLAN",
"Mehmet SAĞIR",
"Onur GEZER",
"Kerem EREN",
"Oğuz ÇELİK",
"Hasan DEMİR",
"Yiğit Taylan YOLAÇAN",
"Mert ACAR",
"Barış AKDUR",
"Bahtiyar Gence SONYOL",
"Necip KİRAZCI",
"Anıl AVCIOĞLU",
"HASAN IŞIK",
"FERHAT ÖKMEN",
"EMİNE TÜRKECAN",
"MERVE HİLAL ÖZBELLİ",
"GÜVEN ONUR ALTIN",
"YAVUZ DURSUN",
"ALİ KIRAL",
"ALİCAN GÜLİŞLİ",
"ARDA İNÇELİ",
"SERCAN GÜRSEL",
"BUĞRAHAN ÇETİNKALE",
"MERT KARACA",
"SEVİM, SİNEM DEDELEROĞLU",
"BERAT CAN ŞENEL",
"SERKAN EMRE ERFİDAN",
"İBRAHİM CÖMERT BARTO",
"ANIL ENSARİ",
"MUSTAFA OKAN DOĞAN",
"SEMİH ŞİMŞEK",
"NURSEDA YILDIRIM",
"EMİR DAĞ",
"SERCAN BALTA",
"GÖZDE ERSAL",
"HÜSEYİN BARIŞ KÜSÜLÜ",
"AYBÜKE ORTAÇ",
"BAŞAR SUDİ KARATAŞ",
"CUMHUR ÖZEKEN",
"ANIL BALIKÇI",
"MUSTAFA KAHVECİ",
"KORHAN KÖKSAL",
"ABDULLAH BUZ",
"BURAK TEMİZ",
"YALÇIN KAÇAR",
"HAKAN KARAASMA",
"BAHADIR ÖZGİRAY",

```
};
   NavigableSet <String>nSet = new ConcurrentSkipListSet<String>();
   for(int i=0;i<student_name.length;i++)
   {nSet.add(student_name[i]);}
//add additional lines here
}
}
```

WEEK 4

4.1 In thermodynamics Van der Waals equation of state is given as:

$$\left(P+\frac{a}{v^2}\right)(v-b)=RT$$

Where a and b is the equation coefficients. This coefficients are defined as a function of critical properties.

$$a=\frac{27R^2T_{critical}^2}{64P_{critical}} \quad b=\frac{RT_{critical}}{8P_{critical}}$$

This properties are given in the table A-1. Pressure P can be taken from

$$P=\frac{RT}{(v-b)}-\frac{a}{v^2}$$

Specific volume v can not be calculated directly, therefore root finding technics are applied. In order to solve the equation

$$f(v)=\frac{RT}{(v-b)}-\frac{a}{v^2}-P=0$$ should be solved. This equation is solved by using Newton-raphson root finding. As the first value ideal gas value is used.

$$v=\frac{RT}{P}$$

vanderWaals and vanderWaalsData is developed for you.Check the written code. Also check idealgas.java code

WEEK 4(chapter 14) CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX4.1 In the program Data is defined only for gases nitrogen, oxygen and carbondioxide. Add up the remaining of the gases (all of them) in the table by using data in the table A-1 and A-2 into the method

public String vanderWaalsInput(String ig_isim) of class vanderWaals

test all gas data you enter into the program by using and changing program vanderWaalstest.java

WEEK 4(chapter 14) HOMEWORK ASSIGNMENTS

HW4.1 write a test program compare specific volume calculated by ideal gas equation of state and vanderWaals equation of state. Develop a test program. Enter the name of the gas, temperature (in degree C) and pressure (in Pa). Program will print out for you specific volume for ideal gas eqaution of state, vanderWalls equation of state and % difference between two equation. Try high pressures also

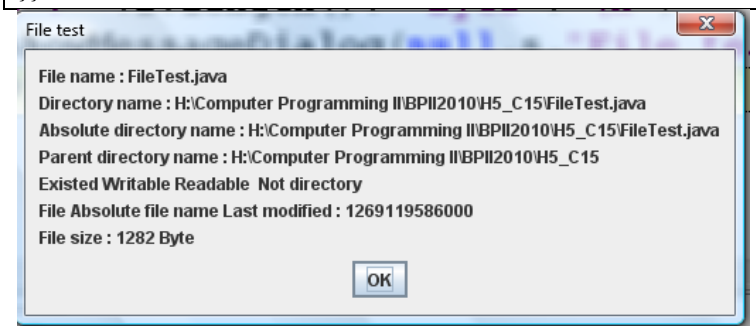
15 FILES, READING AND WRITING DATA FROM FILES

15.1 FILE CLASS

File class defines a File in order to process further in reading and writing from. Whan a file is defined through File class, you can reach detailed file information such as read- write permissions, file size etc. File class can also be used as an input class to open a channel to read and and write from this file. An example program showing some of the features of the class is given belove:

15.1-1 FileTest program

```
import java.awt.*;
import java.io.File;
import java.util.HashMap;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.filechooser.*;
import javax.swing.event.*;
public class FileTest
{
public static void main(String args[])
{
//önce JFileChooser kullanarak dosyayı seç, sonra dosya özelliklerini göster.
File f=new File("readfromfile1.java");
JFileChooser dosyasec=new JFileChooser();
int approval = dosyasec.showOpenDialog(null);
if(approval==JFileChooser.APPROVE_OPTION)
{
f=dosyasec.getSelectedFile();
}
String s="";
s+="File name : "+f.getName()+"\n";
s+="Directory name : "+f.getPath()+"\n";
s+="Absolute directory name : "+f.getAbsolutePath()+"\n";
s+="Parent directory name : "+f.getParent()+"\n";
s+=f.exists() ? "Existed " : " Not existed "+"\n";
s+=f.canWrite() ? "Writable " : " Not writable"+"\n";
s+=f.canRead() ? "Readable " : " Not Readable"+"\n";
s+=f.isDirectory() ? "Directory " : " Not directory"+"\n";
s+=f.isFile() ? "File " : " Not File"+"\n";
s+=f.isAbsolute() ? "Absolute file name " : " not an absolute file name"+"\n";
s+="Last modified : "+f.lastModified()+"\n";
s+="File size : "+f.length()+" Byte"+"\n";
JOptionPane.showMessageDialog(null,s,"File test",JOptionPane.PLAIN_MESSAGE);
System.exit(0);
}}
```



Related field and method list of the file class is given as a table

Field Summary

static String pathSeparator
The system-dependent path-separator character, represented as a string for convenience.

static char pathSeparatorChar
The system-dependent path-separator character.

static String separator
The system-dependent default name-separator character, represented as a string for convenience.

static char separatorChar
The system-dependent default name-separator character.

Constructor Summary

File(File parent, String child)
Creates a new File instance from a parent abstract pathname and a child pathname string.

File(String pathname)
Creates a new File instance by converting the given pathname string into an abstract pathname.

File(String parent, String child)
Creates a new File instance from a parent pathname string and a child pathname string.

File(URI uri)
Creates a new File instance by converting the given file: URI into an abstract pathname.

Method Summary

boolean canExecute()
Tests whether the application can execute the file denoted by this abstract pathname.

boolean canRead()
Tests whether the application can read the file denoted by this abstract pathname.

boolean canWrite()
Tests whether the application can modify the file denoted by this abstract pathname.

int compareTo(File pathname)
Compares two abstract pathnames lexicographically.

boolean createNewFile()
Atomically creates a new, empty file named by this abstract pathname if and only if a file with this name does not yet exist.

static File createTempFile(String prefix, String suffix)
Creates an empty file in the default temporary-file directory, using the given prefix and suffix to generate its name.

static File createTempFile(String prefix, String suffix, File directory)
Creates a new empty file in the specified directory, using the given prefix and suffix strings to generate its name.

boolean delete()
Deletes the file or directory denoted by this abstract pathname.

void deleteOnExit()
Requests that the file or directory denoted by this abstract pathname be deleted when the virtual machine terminates.

boolean equals(Object obj)
Tests this abstract pathname for equality with the given object.

boolean exists()
Tests whether the file or directory denoted by this abstract pathname exists.

File getAbsoluteFile()
Returns the absolute form of this abstract pathname.

String getAbsolutePath()

Returns the absolute pathname string of this abstract pathname.

File getCanonicalFile()
Returns the canonical form of this abstract pathname.

String getCanonicalPath()
Returns the canonical pathname string of this abstract pathname.

long getFreeSpace()
Returns the number of unallocated bytes in the partition named by this abstract path name.

String getName()
Returns the name of the file or directory denoted by this abstract pathname.

String getParent()
Returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory.

File getParentFile()
Returns the abstract pathname of this abstract pathname's parent, or null if this pathname does not name a parent directory.

String getPath()
Converts this abstract pathname into a pathname string.

long getTotalSpace()
Returns the size of the partition named by this abstract pathname.

long getUsableSpace()
Returns the number of bytes available to this virtual machine on the partition named by this abstract pathname.

int hashCode()
Computes a hash code for this abstract pathname.

boolean isAbsolute()
Tests whether this abstract pathname is absolute.

boolean isDirectory()
Tests whether the file denoted by this abstract pathname is a directory.

boolean isFile()
Tests whether the file denoted by this abstract pathname is a normal file.

boolean isHidden()
Tests whether the file named by this abstract pathname is a hidden file.

long lastModified()
Returns the time that the file denoted by this abstract pathname was last modified.

long length()
Returns the length of the file denoted by this abstract pathname.

String[] list()
Returns an array of strings naming the files and directories in the directory denoted by this abstract pathname.

String[] list(FilenameFilter filter)
Returns an array of strings naming the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter.

File[] listFiles()
Returns an array of abstract pathnames denoting the files in the directory denoted by this abstract pathname.

File[] listFiles(FileFilter filter)
Returns an array of abstract pathnames denoting the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter.

File[] listFiles(FilenameFilter filter)
Returns an array of abstract pathnames denoting the files and directories in the directory denoted by this

abstract pathname that satisfy the specified filter.

static File[] listRoots()
List the available filesystem roots.

boolean mkdir()
Creates the directory named by this abstract pathname.

boolean mkdirs()
Creates the directory named by this abstract pathname, including any necessary but nonexistent parent directories.

boolean renameTo(File dest)
Renames the file denoted by this abstract pathname.

boolean setExecutable(boolean executable)
A convenience method to set the owner's execute permission for this abstract pathname.

boolean setExecutable(boolean executable, boolean ownerOnly)
Sets the owner's or everybody's execute permission for this abstract pathname.

boolean setLastModified(long time)
Sets the last-modified time of the file or directory named by this abstract pathname.

boolean setReadable(boolean readable)
A convenience method to set the owner's read permission for this abstract pathname.

boolean setReadable(boolean readable, boolean ownerOnly)
Sets the owner's or everybody's read permission for this abstract pathname.

boolean setReadOnly()
Marks the file or directory named by this abstract pathname so that only read operations are allowed.

boolean setWritable(boolean writable)
A convenience method to set the owner's write permission for this abstract pathname.

boolean setWritable(boolean writable, boolean ownerOnly)
Sets the owner's or everybody's write permission for this abstract pathname.

String toString()
Returns the pathname string of this abstract pathname.

URI toURI()
Constructs a file: URI that represents this abstract pathname.

URL toURL()
Deprecated. *This method does not automatically escape characters that are illegal in URLs. It is recommended that new code convert an abstract pathname into a URL by first converting it into a URI, via the* toURI *method, and then converting the URI into a URL via the* URI.toURL *method.*

Methods inherited from class java.lang. Object

clone, finalize, getClass, notify, notifyAll, wait, wait, wait

15.2 READING SEQUENTIAL FILES THROUGH CHANNELS

Sequential files are files that information inside follows a stream logic. For example a text file is a sequential file. A text or a tape recorder operates in sequential file order. Another possible format fort he files are random data Access. In this format data is written into an adress specified by the computer randomly. Computer memory file, Datalogger file or Compact disk files are written in this order. Java has two input-output library. The first one is io library, the second one is more advanced new input output library (nio). The input output channels in the io library is as follows :

InputStream

FilterInputStream
LineNumberInputStream

BufferedInputStream
DataInputStream
PushbackInputStream
ByteArrayInputStream
FileInputStream
ObjectInputStream
PipedInputStream
SequenceInputStream
StringBufferInputStream
OutputStream

ByteArrayOutputStream

FileOutputStream
FilterOutputStream
BufferedOutputStream

DataOutputStream
PrintStream
ObjectOutputStream
PipedOutputStream
ObjectStreamClass
StreamTokenizer
Reader
FileReader

BufferedReader
LineNumberReader

CharArrayReader
FilterReader
PushbackReader

InputStreamReader

FileReader
PipedReader
StringReader
Writer

BufferedWriter
CharArrayWriter
FilterWriter
OutputStreamWriter
PipedWriter
PrintWriter
StringWriter

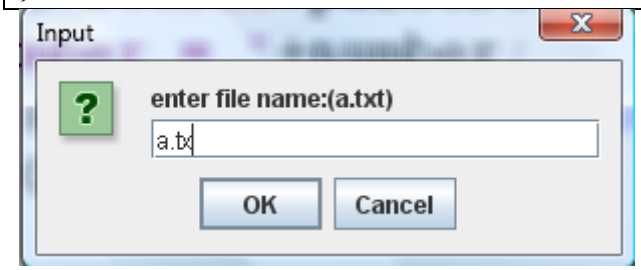
Out of these input output classes BufferedReader and FileReader will be used to open a data read channel from a sequential file as

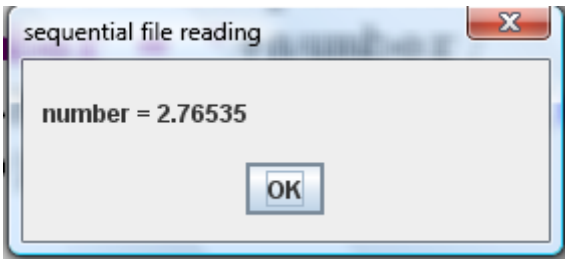
BufferedReader fin=new BufferedReader(new FileReader(f));

The first example is reading double data from a file, and print out the result.

15.2-1 readfromfile class example

```
 import java.io.*;
import java.util.*;
 import javax.swing.*;
 import javax.swing.table.*;
 import java.awt.*;

public class readfromfile
{
public static void main(String args[]) throws IOException
{
String isim=JOptionPane.showInputDialog("enter file name:(a.txt)");
File f=new File(isim);
BufferedReader fin=new BufferedReader(new FileReader(f));
double number=Double.parseDouble(fin.readLine());
String s="number = "+number;
JOptionPane.showMessageDialog(null,s,"sequential file reading",JOptionPane.PLAIN_MESSAGE);
System.exit(0);
}
}
```



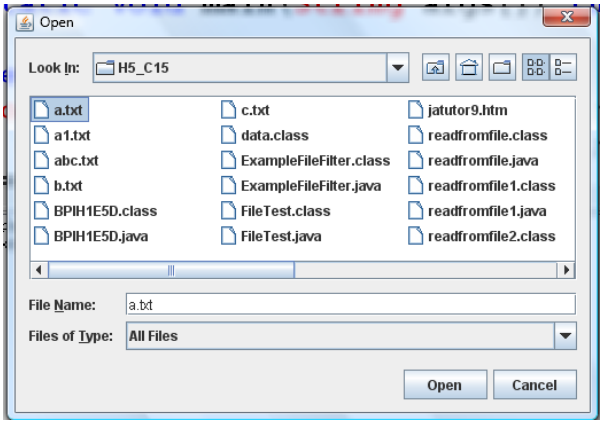


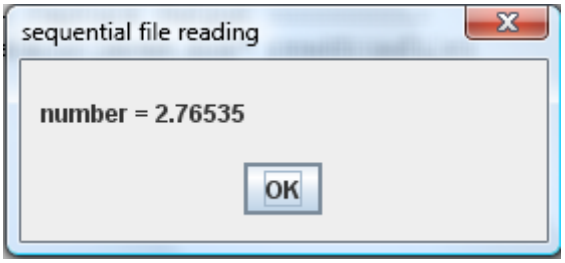
The second example is the same as the first example, but file name is read through JFileChooser graphic chooser.

15.2-2 readfromfile1 class example

```
            import java.io.*;
            import java.util.*;
            import javax.swing.*;
      import javax.swing.table.*;
      import java.awt.*;

public class readfromfile1
{public static void main(String args[]) throws IOException
{
File f=new File("a.txt");
JFileChooser dosyasec=new JFileChooser();
int onay = dosyasec.showOpenDialog(null);
```

```
if(onay==JFileChooser.APPROVE_OPTION)
{f=dosyasec.getSelectedFile();}
BufferedReader fin=new BufferedReader(new FileReader(f));
double number=Double.parseDouble(fin.readLine());
String s="number = "+number;
JOptionPane.showMessageDialog(null,s,"sequential file reading",JOptionPane.PLAIN_MESSAGE);
System.exit(0);
}}
```





In the next example whilw using JFileChooser, only selected type of files, text (txt) type in this case will be shown in the graphic chooser window. In order to establish this, a public non-java program ExampleFileFilter is used. The code is given belove. In the following example, JFileChooser is used together with ExampleFileFilter.

15.2-4 ExampleFileFilter class

```
/*
 *
 *A file filter when displaying files in a file dialog.  The specific use for this class
 *is to only display files with the .gal extension.
 *
 * @(#)ExampleFileFilter.java            1.16 04/07/26
 *
 * Copyright (c) 2004 Sun Microsystems, Inc. All Rights Reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *
 * -Redistribution of source code must retain the above copyright notice, this
 *  list of conditions and the following disclaimer.
 *
 * -Redistribution in binary form must reproduce the above copyright notice,
 *  this list of conditions and the following disclaimer in the documentation
```

```
 *  and/or other materials provided with the distribution.
 *
 * Neither the name of Sun Microsystems, Inc. or the names of contributors may
 * be used to endorse or promote products derived from this software without
 * specific prior written permission.
 *
 * This software is provided "AS IS," without a warranty of any kind. ALL
 * EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING
 * ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE
 * OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN MIDROSYSTEMS, INC. ("SUN")
 * AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE
 * AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS
 * DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST
 * REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL,
 * INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY
 * OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE,
 * EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
 *
 * You acknowledge that this software is not designed, licensed or intended
 * for use in the design, construction, operation or maintenance of any
 * nuclear facility.
 */

/*
 * @(#)ExampleFileFilter.java            1.16 04/07/26
 */


import java.io.File;
import java.util.Hashtable;
import java.util.Enumeration;
import javax.swing.*;
import javax.swing.filechooser.*;

/**
 * A convenience implementation of FileFilter that filters out
 * all files except for those type extensions that it knows about.
 *
 * Extensions are of the type ".foo", which is typically found on
 * Windows and Unix boxes, but not on Macinthosh. Case is ignored.
 *
 * Example - create a new filter that filters out all files
 * but gif and jpg image files:
 *
 *     JFileChooser chooser = new JFileChooser();
 *     ExampleFileFilter filter = new ExampleFileFilter(
 *                 new String{"gif", "jpg"}, "JPEG & GIF Images")
 *     chooser.addChoosableFileFilter(filter);
 *     chooser.showOpenDialog(this);
 *
 * @version 1.16 07/26/04
 * @author Jeff Dinkins
 */
public class ExampleFileFilter extends FileFilter {

    private static String TYPE_UNKNOWN = "Type Unknown";
    private static String HIDDEN_FILE = "Hidden File";
```

```
private Hashtable filters = null;
private String description = null;
private String fullDescription = null;
private boolean useExtensionsInDescription = true;

/**
 * Creates a file filter. If no filters are added, then all
 * files are accepted.
 *
 * @see #addExtension
 */
public ExampleFileFilter() {
	this.filters = new Hashtable();
}

/**
 * Creates a file filter that accepts files with the given extension.
 * Example: new ExampleFileFilter("jpg");
 *
 * @see #addExtension
 */
public ExampleFileFilter(String extension) {
	this(extension,null);
}

/**
 * Creates a file filter that accepts the given file type.
 * Example: new ExampleFileFilter("jpg", "JPEG Image Images");
 *
 * Note that the "." before the extension is not needed. If
 * provided, it will be ignored.
 *
 * @see #addExtension
 */
public ExampleFileFilter(String extension, String description) {
	this();
	if(extension!=null) addExtension(extension);
	if(description!=null) setDescription(description);
}

/**
 * Creates a file filter from the given string array.
 * Example: new ExampleFileFilter(String {"gif", "jpg"});
 *
 * Note that the "." before the extension is not needed and
 * will be ignored.
 *
 * @see #addExtension
 */
public ExampleFileFilter(String[] filters) {
	this(filters, null);
}

/**
 * Creates a file filter from the given string array and description.
 * Example: new ExampleFileFilter(String {"gif", "jpg"}, "Gif and JPG Images");
 *
 * Note that the "." before the extension is not needed and will be ignored.
```

```
 *
 * @see #addExtension
 */
public ExampleFileFilter(String[] filters, String description) {
        this();
        for (int i = 0; i < filters.length; i++) {
            // add filters one by one
            addExtension(filters[i]);
        }
        if(description!=null) setDescription(description);
}

/**
 * Return true if this file should be shown in the directory pane,
 * false if it shouldn't.
 *
 * Files that begin with "." are ignored.
 *
 * @see #getExtension
 * @see FileFilter#accepts
 */
public boolean accept(File f) {
        if(f != null) {
            if(f.isDirectory()) {
                    return true;
            }
            String extension = getExtension(f);
            if(extension != null && filters.get(getExtension(f)) != null) {
                    return true;
            };
        }
        return false;
}

/**
 * Return the extension portion of the file's name .
 *
 * @see #getExtension
 * @see FileFilter#accept
 */
 public String getExtension(File f) {
        if(f != null) {
            String filename = f.getName();
            int i = filename.lastIndexOf('.');
            if(i>0 && i<filename.length()-1) {
                    return filename.substring(i+1).toLowerCase();
            };
        }
        return null;
}

/**
 * Adds a filetype "dot" extension to filter against.
 *
 * For example: the following code will create a filter that filters
 * out all files except those that end in ".jpg" and ".tif":
 *
 *   ExampleFileFilter filter = new ExampleFileFilter();
```

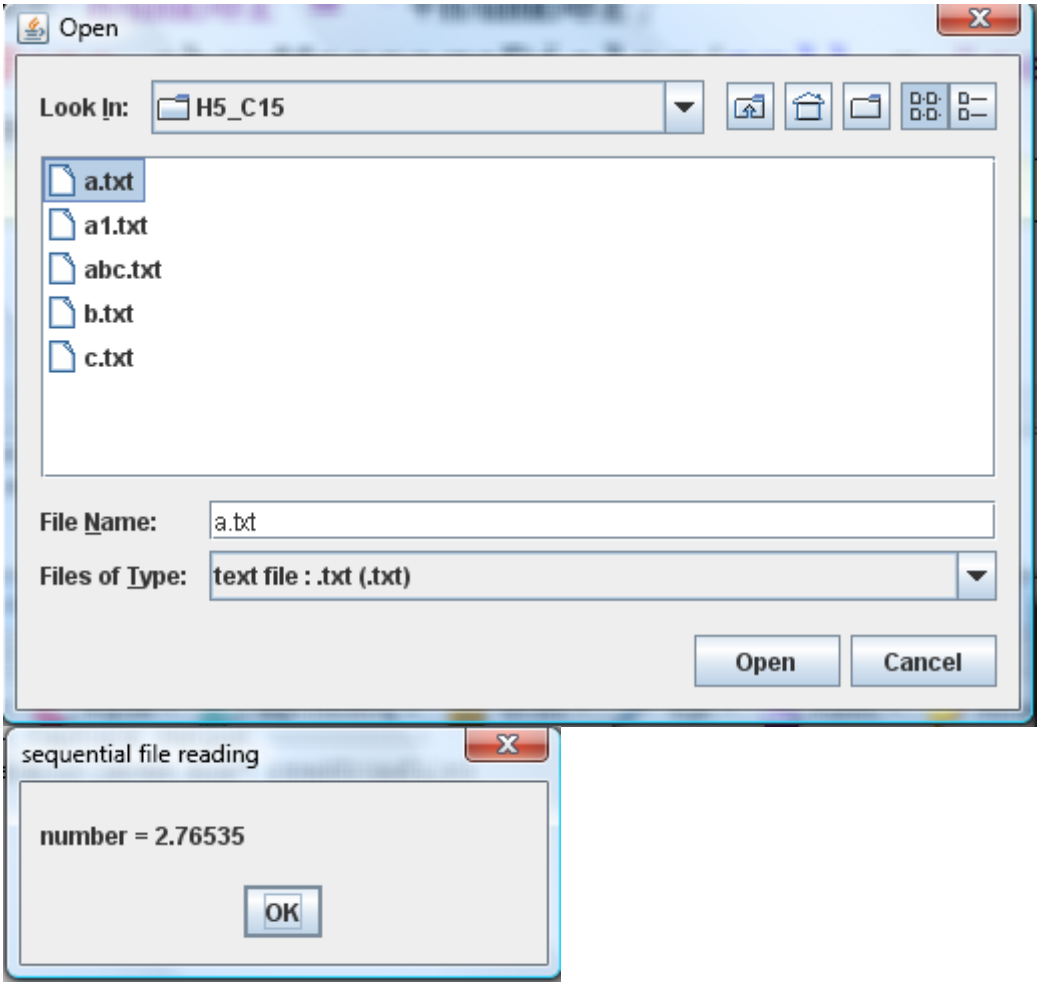
```
 *   filter.addExtension("jpg");
 *   filter.addExtension("tif");
 *
 * Note that the "." before the extension is not needed and will be ignored.
 */
public void addExtension(String extension) {
        if(filters == null) {
            filters = new Hashtable(5);
        }
        filters.put(extension.toLowerCase(), this);
        fullDescription = null;
}


/**
 * Returns the human readable description of this filter. For
 * example: "JPEG and GIF Image Files (*.jpg, *.gif)"
 *
 * @see setDescription
 * @see setExtensionListInDescription
 * @see isExtensionListInDescription
 * @see FileFilter#getDescription
 */
public String getDescription() {
        if(fullDescription == null) {
            if(description == null || isExtensionListInDescription()) {
                    fullDescription = description==null ? "(" : description + " (";
                    // build the description from the extension list
                    Enumeration extensions = filters.keys();
                    if(extensions != null) {
                        fullDescription += "." + (String) extensions.nextElement();
                        while (extensions.hasMoreElements()) {
                                fullDescription += ", ." + (String) extensions.nextElement();
                        }
                    }
                    fullDescription += ")";
            } else {
                    fullDescription = description;
            }
        }
        return fullDescription;
}

/**
 * Sets the human readable description of this filter. For
 * example: filter.setDescription("Gif and JPG Images");
 *
 * @see setDescription
 * @see setExtensionListInDescription
 * @see isExtensionListInDescription
 */
public void setDescription(String description) {
        this.description = description;
        fullDescription = null;
}

/**
 * Determines whether the extension list (.jpg, .gif, etc) should
```

```
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see isExtensionListInDescription
     */
    public void setExtensionListInDescription(boolean b) {
            useExtensionsInDescription = b;
            fullDescription = null;
    }

    /**
     * Returns whether the extension list (.jpg, .gif, etc) should
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see setExtensionListInDescription
     */
    public boolean isExtensionListInDescription() {
            return useExtensionsInDescription;
    }
}
```

15.2-5 readfromfile2 class example

```
            import java.io.*;
            import java.util.*;
            import javax.swing.*;
      import javax.swing.table.*;
      import java.awt.*;

public class readfromfile2
{


public static void main(String args[]) throws IOException
{
  String s1="a.txt";
  JFileChooser fc=new JFileChooser();
  ExampleFileFilter filter = new ExampleFileFilter();
  filter.addExtension("txt");
  filter.setDescription("text file : .txt");
  fc.setFileFilter(filter);
  if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {File file = fc.getSelectedFile();s1=file.getName(); }
  File f=new File(s1);
BufferedReader fin=new BufferedReader(new FileReader(f));
double number=Double.parseDouble(fin.readLine());
String s="number = "+number;
JOptionPane.showMessageDialog(null,s,"sequential file reading",JOptionPane.PLAIN_MESSAGE);
System.exit(0);
```

```
}
}
```

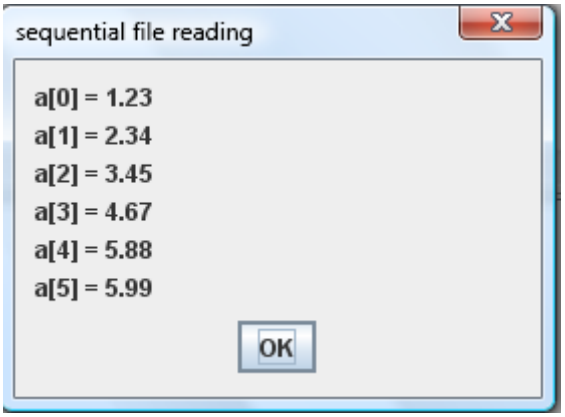


In the next example, StringTokenizer class is used to tokenize a line input stream to a set of double values.

15.2-6 readfromfile3 class example

```
            import java.io.*;
            import java.util.*;
            import javax.swing.*;
    import javax.swing.table.*;
    import java.awt.*;

public class readfromfile3
{

    public static double[] inputdouble(String filename) throws IOException
    {
            //reading double values from one line of data
            BufferedReader fin=new BufferedReader(new FileReader(filename));
    double a[]=inputdouble(fin);
    return a;
    }
```

```
    public static double[] inputdouble(BufferedReader fin) throws IOException
    {
            //reading double values from one line of data
            String s1=fin.readLine();
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }


    public static String toString(double a[])
    {
            String s="";
    for(int i=0;i<a.length;i++)
    {s+="a["+i+"] = "+a[i]+"\n";}
    return s;
            }


public static void main(String args[]) throws IOException
{
  String s1="a.txt";
  JFileChooser fc=new JFileChooser();
  ExampleFileFilter filter = new ExampleFileFilter();
  filter.addExtension("txt");
  filter.setDescription("text file : .txt");
  fc.setFileFilter(filter);
  if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {File file = fc.getSelectedFile();s1=file.getName(); }
  File f=new File(s1);
  BufferedReader fin=new BufferedReader(new FileReader(f));
  double number[]=inputdouble(fin);
  JOptionPane.showMessageDialog(null,toString(number),"sequential file reading",JOptionPane.PLAIN_MESSAGE);
  System.exit(0);
}
}
```

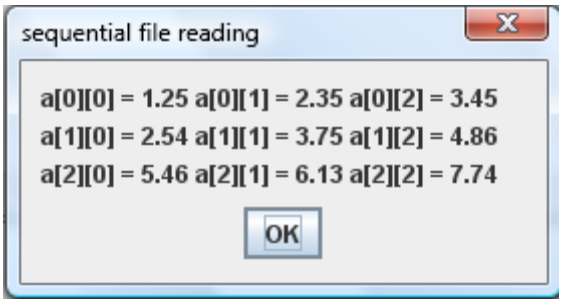


In the next example, StringTokenizer class is used to tokenize a line input stream to a set of double values. Togetger with End Of File Exception class **EOFException,** to read an unknown size data of double matrix.

15.2-7 readfromfile4 class example

```
                import java.io.*;
                import java.util.*;
                import javax.swing.*;
        import javax.swing.table.*;
        import java.awt.*;

class data
                {
                public double x[];
                public data(double[] xi)
                {
                int n=xi.length;
                x=new double[n];
                for(int i=0;i<n;i++) {x[i]=xi[i];}
                }
                public String toString() {return ""+x[0]+" "+x[1];}
                }

public class readfromfile4
{
     public static double[][] inputdouble(String filename) throws IOException
     {
                //reading double values from one line of data
                BufferedReader fin=new BufferedReader(new FileReader(filename));
     double a[][]=inputdouble(fin);
     return a;
     }

    public static double[][] inputdouble(BufferedReader fin) throws IOException
    {
                ArrayList<data> a=new ArrayList<data>();
                int mmax=0;
                double b[][];
                int n=0,m=0;
                String s1="";
                try{
       while(fin!=null)
       { s1=fin.readLine();
       if(s1==null)break;
         StringTokenizer token=new StringTokenizer(s1);
         m=token.countTokens();
         if(m>mmax) mmax=m;
         double aa[]=new double[m];
         int j=0;
         while(token.hasMoreTokens())
         { Double ax=new Double(token.nextToken());
           aa[j++]=ax.doubleValue();
         }
         data vx=new data(aa);
         a.add(vx);
       n++;
                }
```

```
      } catch(EOFException e_eof)
        {
        //close ffile
        try{fin.close();
      }
      catch(NullPointerException e) {}
      catch(IOException e)
            {
            System.err.println("Error Closing File\n"+e.toString());
            System.exit(1);
            }
        } //End of EOFException
    b=new double[n][mmax];
    int k=0;
    Iterator i=a.iterator();
    while(i.hasNext()) {data vx=(data)i.next();b[k++]=vx.x;}
    return b;
    }

    public static String toString(double a[][])
    {
            String s="";
    for(int i=0;i<a.length;i++)
    {
            for(int j=0;j<a[0].length;j++)
    {s+="a["+i+"]["+j+"] = "+a[i][j]+" ";}
    s+="\n";
    }
    return s;
            }

public static void main(String args[]) throws IOException
{
  String s1="a.txt";
  JFileChooser fc=new JFileChooser();
  ExampleFileFilter filter = new ExampleFileFilter();
  filter.addExtension("txt");
  filter.setDescription("text file : .txt");
  fc.setFileFilter(filter);
  if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {File file = fc.getSelectedFile();s1=file.getName(); }
  File f=new File(s1);
  BufferedReader fin=new BufferedReader(new FileReader(f));
  double number[][]=inputdouble(fin);
  JOptionPane.showMessageDialog(null,toString(number),"sequential file reading",JOptionPane.PLAIN_MESSAGE);
  System.exit(0);
}
}
```



Details of constructor and other methods of BufferReader and FileReader classes are given belove:

Field Summary

Fields inherited from class java.io. Reader

lock

Constructor Summary

BufferedReader(Reader in)
Create a buffering character-input stream that uses a default-sized input buffer.

BufferedReader(Reader in, int sz)
Create a buffering character-input stream that uses an input buffer of the specified size.

Method Summary

void	close() Close the stream.
void	mark(int readAheadLimit) Mark the present position in the stream.
boolean	markSupported() Tell whether this stream supports the mark() operation, which it does.
int	read() Read a single character.
int	read(char[] cbuf, int off, int len) Read characters into a portion of an array.
String	readLine() Read a line of text.
boolean	ready() Tell whether this stream is ready to be read.
Void	reset() Reset the stream to the most recent mark.
Long	skip(long n) Skip characters.

Methods inherited from class java.io. Reader

read

Methods inherited from class java.lang. Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

FileReader class;

Field Summary

Fields inherited from class java.io. Reader

lock

Constructor Summary

FileReader(File file)
Creates a new FileReader, given the File to read from.

FileReader(FileDescriptor fd)
Creates a new FileReader, given the FileDescriptor to read from.

FileReader(String fileName)
Creates a new FileReader, given the name of the file to read from.

Methods inherited from class java.io. InputStreamReader

close, getEncoding, read, read, ready

Methods inherited from class java.io. Reader

mark, markSupported, read, reset, skip

Methods inherited from class java.lang. Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

15.2.2 SCANNER CLASS

Scanner class is a realatively new form of ınput channel for reading data from sequential file. In addition to the role of above example, ie reading data, this class has build in StringTokenizer class check so it can read the data in tokenized form directly. The first example of this clas was given at the beginning of the first semester, now we can look at it again

15.2-8 BPIH1E5D class example

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;  // international format library Locale
import javax.swing.*;     //Java swing graphic library
import java.io.*;         //Java input library for File class

public class BPIH1E5D
{
    // main method begins execution of Java application
    public static void main( String args[] ) throws FileNotFoundException
    {
       // create Scanner to obtain input from a file
      String filename=JOptionPane.showInputDialog("Enter file name (a.txt): ");
      Scanner input = new Scanner(new File(filename));
      Locale us=new Locale("US");
      input.useLocale(us);
      double number1; // first number to add
      double number2; // second number to add
      double sum; // sum of number1 and number2
      number1 = input.nextDouble(); // read first number from user
      number2 = input.nextDouble(); // read second number from user
      sum = number1 + number2; // add numbers
      String s=String.format(us,"Sum is %10.5f\n", sum );
      System.out.println(s); // display sum
    } // end method main
} // end class BPIH1E5D
```

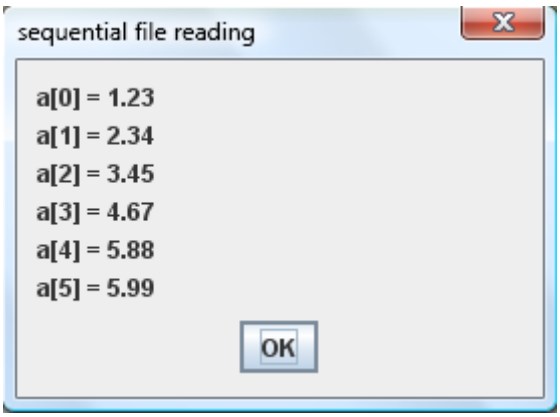
```
---------- Capture Output ----------

> "D:\java\bin\javaw.exe" BPIH1E5D

Sum is    3.57000

> Terminated with exit code 0.
```

In the next example, A vector of data is read by using Scanner class. The advantage of this vector reading class compare to previous BufferedReader version is that, this program can read the vector even if it is not in one line only.

15.2-9 BPIH1E5D class example

```
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
```

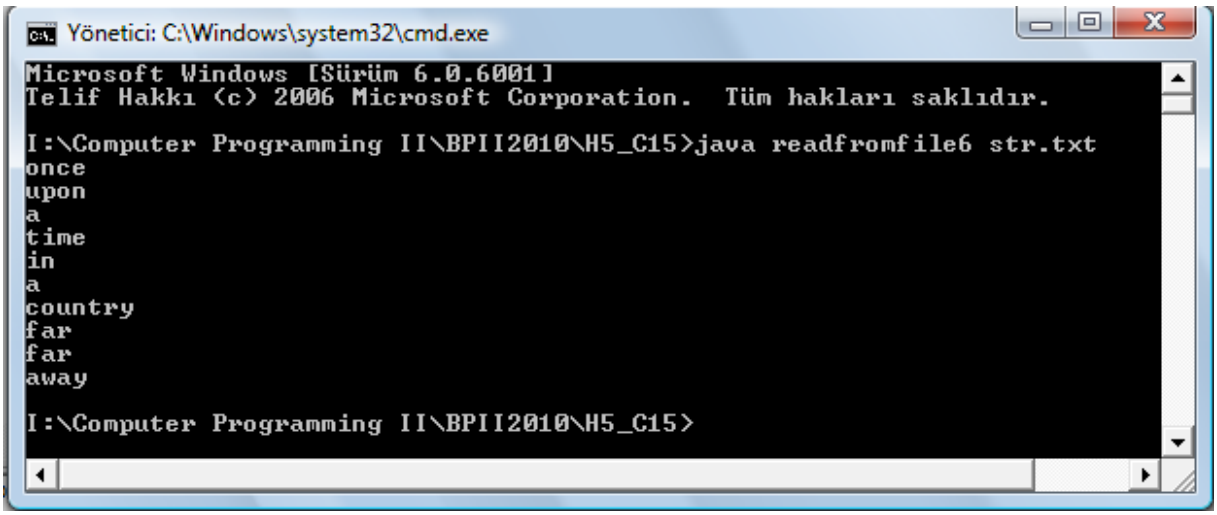
```
public class readfromfile5
{
    public static double[] inputdouble(String filename) throws IOException
    {
   ArrayList<String> b=new ArrayList<String>();
   Scanner fin = new Scanner(new File(filename));
    int n=0;
    int k=0;
    while(fin.hasNext())
    {b.add((String)fin.next());}
    double a[]=new double[b.size()];
    Iterator i=b.iterator();
    while(i.hasNext())
    {Double ax=new Double((String)i.next());
     a[k++]=ax.doubleValue();
    }
     return a;
    }

    public static String toString(double a[])
    {
            String s="";
    for(int i=0;i<a.length;i++)
    { s+="a["+i+"] = "+a[i]+"\n";}
    return s;
            }


public static void main(String args[]) throws IOException
{
  String s1="b.txt";
  double number[]=inputdouble(s1);
  JOptionPane.showMessageDialog(null,toString(number),"sequential file reading",JOptionPane.PLAIN_MESSAGE);
  System.exit(0);
}
}
```



Another similar example to read string array. In this example file name should be given in the run statement, so the output and java statement is run in console(dos) environment

15.2-10 readfromfile6 class example

```
import java.io.File;

 import java.io.FileNotFoundException;

import java.util.Scanner;

public class readfromfile6 {

  private static void readFile(String fileName) {
    try {
      File file = new File(fileName);
      Scanner scanner = new Scanner(file);
      while (scanner.hasNext()) {
        System.out.println(scanner.next());
      }
      scanner.close();
    } catch (FileNotFoundException e) {
      e.printStackTrace();
    }
  }

  public static void main(String[] args) {
    if (args.length != 1) {
      System.err.println("usage: java readfromfile6"+ "file location");
      System.exit(0);
    }
    readFile(args[0]);
  }
}
```



Another advantage of the scanner class would be ability of reading a mixed data from the file

15.2-11 readfromfile7 class example

```
import java.io.FileReader;
import java.io.FileWriter;
```

```
import java.io.IOException;
import java.util.Scanner;

public class readfromfile7 {
  public static void main(String args[]) throws IOException {

    int i;
    double d;
    boolean b;
    String str;
    FileReader fin = new FileReader("mixed.txt");
    Scanner src = new Scanner(fin);

    while (src.hasNext()) {
      if (src.hasNextInt()) {
        i = src.nextInt();
        System.out.println("int: " + i);
      } else if (src.hasNextDouble()) {
        d = src.nextDouble();
        System.out.println("double: " + d);
      } else if (src.hasNextBoolean()) {
        b = src.nextBoolean();
        System.out.println("boolean: " + b);
      } else {
        str = src.next();
        System.out.println("String: " + str);
      }
    }

    fin.close();
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" readfromfile7
String: Testing
String: Scanner
int: 10
String: 12.2
String: one
boolean: true
String: two
boolean: false
> Terminated with exit code 0.
```

Constructor methods and other methods of the scanner class is given belove:

Constructor Summary

Scanner(File source)
Constructs a new Scanner that produces values scanned from the specified file.

Scanner(File source, String charsetName)
Constructs a new Scanner that produces values scanned from the specified file.

Scanner(InputStream source)
Constructs a new Scanner that produces values scanned from the specified input stream.

Scanner(InputStream source, String charsetName)
Constructs a new Scanner that produces values scanned from the specified input stream.

Scanner(Readable source)
Constructs a new Scanner that produces values scanned from the specified source.

Scanner(ReadableByteChannel source)
Constructs a new Scanner that produces values scanned from the specified channel.

Scanner(ReadableByteChannel source, String charsetName)
Constructs a new Scanner that produces values scanned from the specified channel.

Scanner(String source)
Constructs a new Scanner that produces values scanned from the specified string.

Method Summary

void	close() Closes this scanner.
Pattern	delimiter() Returns the Pattern this Scanner is currently using to match delimiters.
String	findInLine(Pattern pattern) Attempts to find the next occurrence of the specified pattern ignoring delimiters.
String	findInLine(String pattern) Attempts to find the next occurrence of a pattern constructed from the specified string, ignoring delimiters.
String	findWithinHorizon(Pattern pattern, int horizon) Attempts to find the next occurrence of the specified pattern.
String	findWithinHorizon(String pattern, int horizon) Attempts to find the next occurrence of a pattern constructed from the specified string, ignoring delimiters.
boolean	hasNext() Returns true if this scanner has another token in its input.
boolean	hasNext(Pattern pattern) Returns true if the next complete token matches the specified pattern.
boolean	hasNext(String pattern) Returns true if the next token matches the pattern constructed from the specified string.
boolean	hasNextBigDecimal() Returns true if the next token in this scanner's input can be interpreted as a BigDecimal using the nextBigDecimal() method.
boolean	hasNextBigInteger() Returns true if the next token in this scanner's input can be interpreted as a BigInteger in the default radix using the nextBigInteger() method.
boolean	hasNextBigInteger(int radix) Returns true if the next token in this scanner's input can be interpreted as a BigInteger in the specified radix using the nextBigInteger() method.
boolean	hasNextBoolean() Returns true if the next token in this scanner's input can be interpreted as a boolean value using a case insensitive pattern created from the string "true|false".
boolean	hasNextByte() Returns true if the next token in this scanner's input can be interpreted as a byte value in the default radix using the nextByte() method.
boolean	hasNextByte(int radix) Returns true if the next token in this scanner's input can be interpreted as a byte value in the specified radix using the nextByte() method.

boolean	hasNextDouble() Returns true if the next token in this scanner's input can be interpreted as a double value using the nextDouble() method.
boolean	hasNextFloat() Returns true if the next token in this scanner's input can be interpreted as a float value using the nextFloat() method.
boolean	hasNextInt() Returns true if the next token in this scanner's input can be interpreted as an int value in the default radix using the nextInt() method.
boolean	hasNextInt(int radix) Returns true if the next token in this scanner's input can be interpreted as an int value in the specified radix using the nextInt() method.
boolean	hasNextLine() Returns true if there is another line in the input of this scanner.
boolean	hasNextLong() Returns true if the next token in this scanner's input can be interpreted as a long value in the default radix using the nextLong() method.
boolean	hasNextLong(int radix) Returns true if the next token in this scanner's input can be interpreted as a long value in the specified radix using the nextLong() method.
boolean	hasNextShort() Returns true if the next token in this scanner's input can be interpreted as a short value in the default radix using the nextShort() method.
boolean	hasNextShort(int radix) Returns true if the next token in this scanner's input can be interpreted as a short value in the specified radix using the nextShort() method.
IOException	ioException() Returns the IOException last thrown by this Scanner's underlying Readable.
Locale	locale() Returns this scanner's locale.
MatchResult	match() Returns the match result of the last scanning operation performed by this scanner.
String	next() Finds and returns the next complete token from this scanner.
String	next(Pattern pattern) Returns the next token if it matches the specified pattern.
String	next(String pattern) Returns the next token if it matches the pattern constructed from the specified string.
BigDecimal	nextBigDecimal() Scans the next token of the input as a BigDecimal.
BigInteger	nextBigInteger() Scans the next token of the input as a BigInteger.
BigInteger	nextBigInteger(int radix) Scans the next token of the input as a BigInteger.
boolean	nextBoolean() Scans the next token of the input into a boolean value and returns that value.
byte	nextByte() Scans the next token of the input as a byte.
byte	nextByte(int radix) Scans the next token of the input as a byte.

double	nextDouble() Scans the next token of the input as a double.
float	nextFloat() Scans the next token of the input as a float.
int	nextInt() Scans the next token of the input as an int.
int	nextInt(int radix) Scans the next token of the input as an int.
String	nextLine() Advances this scanner past the current line and returns the input that was skipped.
long	nextLong() Scans the next token of the input as a long.
long	nextLong(int radix) Scans the next token of the input as a long.
short	nextShort() Scans the next token of the input as a short.
short	nextShort(int radix) Scans the next token of the input as a short.
int	radix() Returns this scanner's default radix.
void	remove() The remove operation is not supported by this implementation of Iterator.
Scanner	skip(Pattern pattern) Skips input that matches the specified pattern, ignoring delimiters.
Scanner	skip(String pattern) Skips input that matches a pattern constructed from the specified string.
String	toString() Returns the string representation of this Scanner.
Scanner	useDelimiter(Pattern pattern) Sets this scanner's delimiting pattern to the specified pattern.
Scanner	useDelimiter(String pattern) Sets this scanner's delimiting pattern to a pattern constructed from the specified String.
Scanner	useLocale(Locale locale) Sets this scanner's locale to the specified locale.
Scanner	useRadix(int radix) Sets this scanner's default radix to the specified radix.

Methods inherited from class java.lang. Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

15.3 WRITING INTO SEQUENTIAL FILES THROUGH CHANNELS

Writing the data is similar to readinng the data. Again, there are several ways of opening a data channel in java for writing. In the first example PrintWriter and BufferedWriter combination will be used.

15.3-1 writeandread class

```
import java.io.*;
class writeandread
{
    public static void main (String args[]) throws IOException
    {
```

```
            String s1="first";
            String s2="second";
            String s3="third";
            PrintWriter cfout=new PrintWriter(new BufferedWriter(new FileWriter("abc.txt")));
            cfout.println("Hello");
            cfout.println("What is your name?");
            cfout.println("nice to meeting you");
            cfout.close();
            BufferedReader cfin=new BufferedReader(new FileReader("abc.txt"));
            s1=cfin.readLine();
            s2=cfin.readLine();
            s3=cfin.readLine();
            cfin.close();
            System.out.println("s1 = "+s1);
            System.out.println("s2 = "+s2);
            System.out.println("s3 = "+s3);
    }
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" writeandread
s1 = Hello
s2 = What is your name?
s3 = nice to meeting you

> Terminated with exit code 0.
```

As a second channel example file writer class will be used alone

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class FileWriterTest {
  public static void main(String args[]) throws IOException {

    int i;
    double d;
    boolean b;
    String str;

    FileWriter fout = new FileWriter("mixed1.txt");
    fout.write("Testing Scanner 10 12.2 one true two false");
    fout.close();

    FileReader fin = new FileReader("mixed1.txt");
    Scanner src = new Scanner(fin);

    while (src.hasNext()) {
      if (src.hasNextInt()) {
        i = src.nextInt();
        System.out.println("int: " + i);
      } else if (src.hasNextDouble()) {
        d = src.nextDouble();
        System.out.println("double: " + d);
```

```
      } else if (src.hasNextBoolean()) {
        b = src.nextBoolean();
        System.out.println("boolean: " + b);
      } else {
        str = src.next();
        System.out.println("String: " + str);
      }
    }

    fin.close();
  }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" FileWriterTest
String: Testing
String: Scanner
int: 10
String: 12.2
String: one
boolean: true
String: two
boolean: false

> Terminated with exit code 0.
```

The constructor and other methods of the PrintWriter Class :

Field Summary

protected Writer	out	The underlying character-output stream of this PrintWriter.

Fields inherited from class java.io.Writer

lock

Constructor Summary

PrintWriter(File file)
Creates a new PrintWriter, without automatic line flushing, with the specified file.

PrintWriter(File file, String csn)
Creates a new PrintWriter, without automatic line flushing, with the specified file and charset.

PrintWriter(OutputStream out)
Creates a new PrintWriter, without automatic line flushing, from an existing OutputStream.

PrintWriter(OutputStream out, boolean autoFlush)
Creates a new PrintWriter from an existing OutputStream.

PrintWriter(String fileName)
Creates a new PrintWriter, without automatic line flushing, with the specified file name.

PrintWriter(String fileName, String csn)
Creates a new PrintWriter, without automatic line flushing, with the specified file name and charset.

PrintWriter(Writer out)
Creates a new PrintWriter, without automatic line flushing.

PrintWriter(Writer out, boolean autoFlush)
Creates a new PrintWriter.

Method Summary

PrintWriter	append(char c) Appends the specified character to this writer.
PrintWriter	append(CharSequence csq) Appends the specified character sequence to this writer.
PrintWriter	append(CharSequence csq, int start, int end) Appends a subsequence of the specified character sequence to this writer.
boolean	checkError() Flushes the stream if it's not closed and checks its error state.
protected void	clearError() Clears the error state of this stream.
void	close() Closes the stream and releases any system resources associated with it.
void	flush() Flushes the stream.
PrintWriter	format(Locale l, String format, Object... args) Writes a formatted string to this writer using the specified format string and arguments.
PrintWriter	format(String format, Object... args) Writes a formatted string to this writer using the specified format string and arguments.
void	print(boolean b) Prints a boolean value.
void	print(char c) Prints a character.
void	print(char[] s) Prints an array of characters.
void	print(double d) Prints a double-precision floating-point number.
void	print(float f) Prints a floating-point number.
void	print(int i) Prints an integer.
void	print(long l) Prints a long integer.
void	print(Object obj) Prints an object.
void	print(String s) Prints a string.
PrintWriter	printf(Locale l, String format, Object... args) A convenience method to write a formatted string to this writer using the specified format string and arguments.
PrintWriter	printf(String format, Object... args) A convenience method to write a formatted string to this writer using the specified format string and arguments.
void	println() Terminates the current line by writing the line separator string.
void	println(boolean x)

Prints a boolean value and then terminates the line.

void println(char x)
Prints a character and then terminates the line.

void println(char[] x)
Prints an array of characters and then terminates the line.

void println(double x)
Prints a double-precision floating-point number and then terminates the line.

void println(float x)
Prints a floating-point number and then terminates the line.

void println(int x)
Prints an integer and then terminates the line.

void println(long x)
Prints a long integer and then terminates the line.

void println(Object x)
Prints an Object and then terminates the line.

void println(String x)
Prints a String and then terminates the line.

protected void setError()
Indicates that an error has occurred.

void write(char[] buf)
Writes an array of characters.

void write(char[] buf, int off, int len)
Writes A Portion of an array of characters.

void write(int c)
Writes a single character.

void write(String s)
Writes a string.

void write(String s, int off, int len)
Writes a portion of a string.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor and other classes of BufferedWriter class:

Field Summary

Fields inherited from class java.io.Writer

lock

Constructor Summary

BufferedWriter(Writer out)
Creates a buffered character-output stream that uses a default-sized output buffer.

BufferedWriter(Writer out, int sz)
Creates a new buffered character-output stream that uses an output buffer of the given size.

Method Summary

void close()
Closes the stream, flushing it first.

void flush()
Flushes the stream.

void newLine()
Writes a line separator.

void write(char[] cbuf, int off, int len)

Writes a portion of an array of characters.

void write(int c)
Writes a single character.

void write(String s, int off, int len)
Writes a portion of a String.

Methods inherited from class java.io.Writer

append, append, append, write, write

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor and other methods of FileWriter class:

Field Summary

Fields inherited from class java.io.Writer

lock

Constructor Summary

FileWriter(File file)
Constructs a FileWriter object given a File object.

FileWriter(File file, boolean append)
Constructs a FileWriter object given a File object.

FileWriter(FileDescriptor fd)
Constructs a FileWriter object associated with a file descriptor.

FileWriter(String fileName)
Constructs a FileWriter object given a file name.

FileWriter(String fileName, boolean append)
Constructs a FileWriter object given a file name with a boolean indicating whether or not to append the data written.

Method Summary

Methods inherited from class java.io.OutputStreamWriter

close, flush, getEncoding, write, write, write

Methods inherited from class java.io.Writer

append, append, append, write, write

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

There are more possibilities for writing sequential files, for our purposes this much is enough for now. Interested students will look nio library also for further efficient buffered definitions.

15.4 ZIP,UNZIP CLASSES

To compress data zip format is one of the most used format. **ZipOutputStream** defined in java io can do zipping data for us. Our example class can work either from console(dos) environment or from graphic input screens toz ip any given file or directory content.

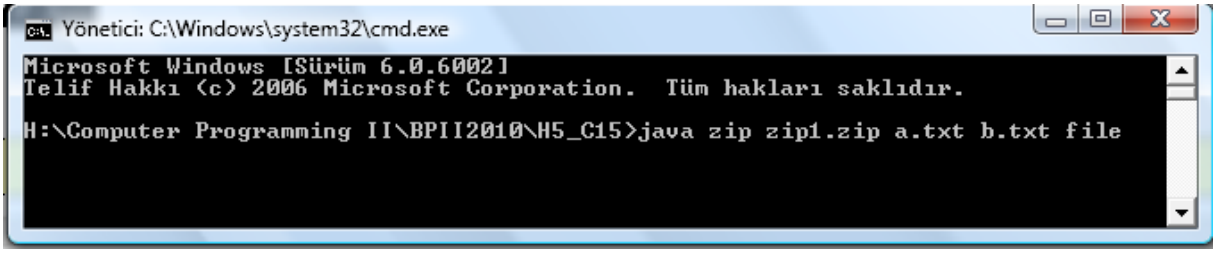
Program 15.4-1 class zip.java

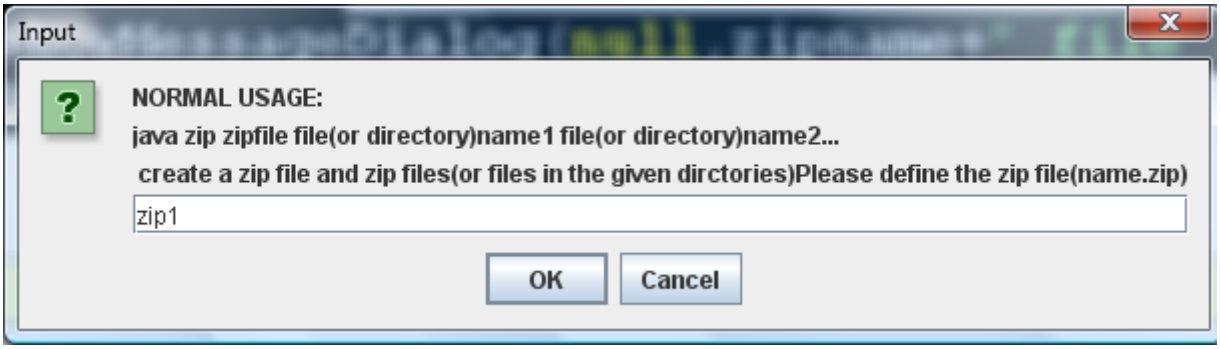
```
//file : zip.java
//Turhan Çoban
import java.io.*;
import java.util.*;
import java.util.zip.*;
import javax.swing.JOptionPane;
```

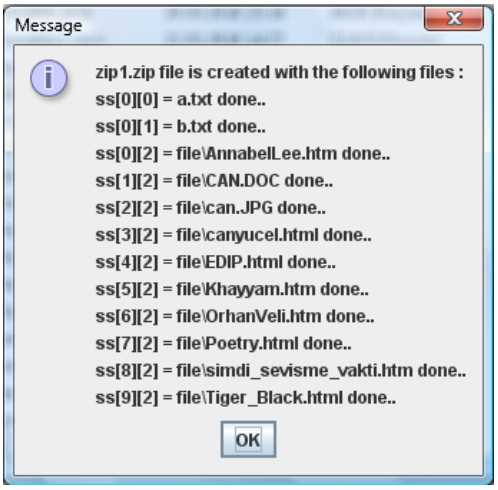
```
public class zip {
   public static int sChunk = 8192;

   public static String[] dosyalistesi(String p)
   {
    File path=new File(p);
    if(path.isDirectory())
    {
    ArrayList<String> x=new ArrayList<String>();
    File files[];
    files=path.listFiles();
    int n=files.length;
    int n1=0;
    for(int i=0;i<n;i++)
    {
     if(!files[i].isDirectory())
        {
        x.add(files[i].toString());
        n1++;
        }
    }
    String s1[];
    s1=new String[n1];
    Iterator nn=x.iterator();
    int k=0;
    while(nn.hasNext())
    {s1[k++]=(String)nn.next();}
    return s1;
    }
    else
    {
    String[] s1 = new String[1];
    s1[0]=p;
    return s1;
    }
   }


   public static void main(String[] args) {
   String str="";
   FileOutputStream out;
   ZipOutputStream zout;
   String zipname, source;
   int m,mmax=1;
   String s="NORMAL USAGE: \n"+
     "java zip zipfile file(or directory)name1 file(or directory)name2...\n"+
     " create a zip file and zip files(or files in the given dirctories)"+
     "Please define the zip file(name.zip)";
//  anahtarı kontrol et
    String zipismi="";
    String dosyaismi[];
    if(args.length==0 )
    {
    String s1="zip file name.zip is undefined please enter the name)\n"+s;
    zipismi=JOptionPane.showInputDialog(null,s);

                String s2="file(or directory) names are undefined\n"+
```

```
 " please enter names\n";
 String s3=JOptionPane.showInputDialog(null,s2);
 StringTokenizer token=new StringTokenizer(s3);
    m=token.countTokens();
    if(m>mmax) mmax=m;
    dosyaismi=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    { dosyaismi[j++]=token.nextToken();}

 } // girdi dosyasını incele
 else if(args.length==1)
 {zipismi=args[0];
  String s1="file(or directory) to be zipped is undefined please enter the name)\n"+s;
 dosyaismi=new String[1];
 dosyaismi[0]=JOptionPane.showInputDialog(null,s1);
 }
 else
 {   zipismi=args[0];
             int n=args.length-1;
             dosyaismi=new String[n];
    for(int i=0;i<n;i++)
    {dosyaismi[i]=args[i+1];}
 }
 if (zipismi.endsWith(".zip")) {
  zipname = zipismi;
  source = zipismi.substring(0, zipismi.length(  ) - 3);
}
else {
  zipname = zipismi + ".zip";
  source = zipismi;
     }
//zip kanalını tanımla
try {
out = new FileOutputStream(zipname);
zout=new ZipOutputStream(out);
    }
catch (IOException e) {
JOptionPane.showMessageDialog(null,"Dosya oluşturma hatası : " + zipname + ".");
  return;
}
int k=0;
while(k<dosyaismi.length)
{
try
{
   String ss[];
   ss=dosyalistesi(dosyaismi[k]);
   int j=0;
   byte[] buffer=new byte[sChunk];
   while(j<ss.length)
   {
   ZipEntry giris=new ZipEntry(ss[j]);
   zout.putNextEntry(giris);
   try {
      FileInputStream in=new FileInputStream(ss[j]);
      int length;
      while((length=in.read(buffer,0,sChunk))!=-1)
```

```
            zout.write(buffer,0,length);
        in.close();
        } //try sonu
        catch (IOException e)
        {
        JOptionPane.showMessageDialog(null,"file zip error : " +ss[j] );
        } //catch sonu
      str+="ss["+j+"]["+k+"] = "+giris+" done..\n";
      j++;
      }//while sonu
    } //try sonu
    catch(IOException io) {}
    k++;
    } //while(k< sonu
    try{
      zout.close();
      out.close();
      }
    catch(IOException e){}
    JOptionPane.showMessageDialog(null,zipname+" file is created with the following files : \n"+str);
    System.exit(0);
    }
}
```





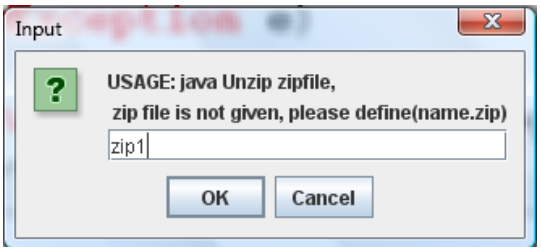


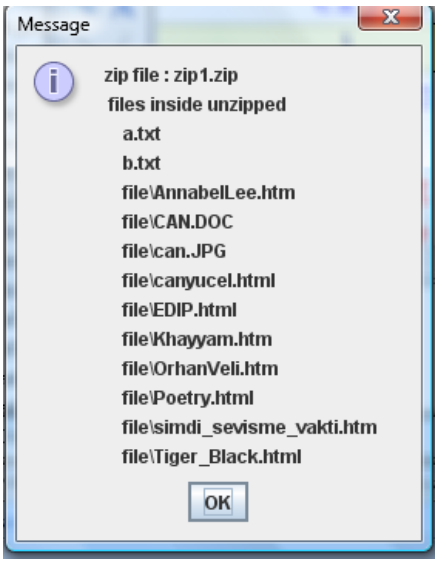
Of course if you zip the data, you should unzip it to recover the data in the original form **ZipInputStream** an be used for this purpose.

Program 15.4-2 class unzpzip.java

```
// dosya : unzip.java
import java.io.*;
import java.util.zip.*;
import javax.swing.JOptionPane;

public class unzip {
   public static int sChunk = 8192;

   public static void main(String[] args) {
   String zipname, source;
   String s="";
   if (args.length != 1) {
     zipname=JOptionPane.showInputDialog(null,"USAGE: java Unzip zipfile, \n zip file is not given, please
define(name.zip)");
                        }
   else{zipname=args[0];}
   // girdi dosyasını incele
   if (zipname.endsWith(".zip"))
   {source = zipname.substring(0, zipname.length(  ) - 3);
}
   else
   {
     zipname = zipname + ".zip";
     source = zipname;
   }
   s+="zip file : "+zipname+"\n files inside unzipped \n";
   try {
         FileInputStream in = new FileInputStream(zipname);
         ZipInputStream zin=new ZipInputStream(in);
         byte[] buffer = new byte[sChunk];
         // dosyayı aç (decompress
         try{
         ZipEntry z;
         do
         {
         z=zin.getNextEntry();
         try {
```

```
            FileOutputStream out = new FileOutputStream(z.getName());
            int length;
            while ((length = zin.read(buffer, 0, sChunk)) != -1)
            out.write(buffer, 0, length);
            out.close( );
            }
            catch (IOException e) {
            JOptionPane.showMessageDialog(null,
            "File zip error : " + zipname + ".");
            System.exit(0);
                        }
          s+="    "+z.getName()+" \n";
          }while(z!=null);
           }  catch(NullPointerException npe)
             { JOptionPane.showMessageDialog(null,s);
              System.exit(0);
             }
         try { zin.close(); } catch (IOException e) {}
          }
          catch(IOException e)
          {
          JOptionPane.showMessageDialog(null,"File open error : " + zipname);
          System.exit(0);
          }
  }
}
```





Constructor and other methods for **ZipOutputStream**

Fields inherited from class java.util.zip. DeflaterOutputStream

buf, def

Fields inherited from class java.io. FilterOutputStream

out

Constructor Summary

ZipOutputStream(OutputStream out)
Creates a new ZIP output stream.

Method Summary

void close()
Closes the ZIP output stream as well as the stream being filtered.

void closeEntry()
Closes the current ZIP entry and positions the stream for writing the next entry.

void finish()
Finishes writing the contents of the ZIP output stream without closing the underlying stream.

void putNextEntry(ZipEntry e)
Begins writing a new ZIP file entry and positions the stream to the start of the entry data.

void setComment(String comment)
Sets the ZIP file comment.

void setLevel(int level)
Sets the compression level for subsequent entries which are DEFLATED.

void setMethod(int method)
Sets the default compression method for subsequent entries.

void write(byte[] b, int off, int len)
Writes an array of bytes to the current ZIP entry data.

Methods inherited from class java.util.zip. DeflaterOutputStream

deflate, write

Methods inherited from class java.io. FilterOutputStream

flush, write

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors and other methods for **ZipInputStream**

Fields inherited from class java.util.zip. InflaterInputStream

buf, inf, len

Fields inherited from class java.io. FilterInputStream

in

Constructor Summary

ZipInputStream(InputStream in)
Creates a new ZIP input stream.

Method Summary

int available()
Returns 0 after EOF has reached for the current entry data, otherwise always return 1.

void close()
Closes the ZIP input stream.

void closeEntry()
Closes the current ZIP entry and positions the stream for reading the next entry.

protected ZipEntry createZipEntry(String name)
Creates a new ZipEntry object for the specified entry name.

ZipEntry getNextEntry()
Reads the next ZIP file entry and positions stream at the beginning of the entry data.

int read(byte[] b, int off, int len)
Reads from the current ZIP entry into an array of bytes.

long skip(long n)
Skips specified number of bytes in the current ZIP entry.

Methods inherited from class java.util.zip. InflaterInputStream

fill, read

Methods inherited from class java.io. FilterInputStream

mark, markSupported, read, reset

Methods inherited from class java.lang. Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

WEEK 5(chapter 15) CLASS EXERSIZES (Wil be shown at the and of Lab class)

EX1) Temperature and specific heat of CO_2 gas is given in file co2_cv.txt . Read the data, from the file and output to the screen. Use

BufferedReader fin=new BufferedReader(new FileReader("co2_cv.txt")); To open the channel

EX 2) : Read x and y values from file e.txt. Write x and $\sqrt{y}$ values into file f.txt file.

EX3) by using zip.java compress readfromfile.java into readfromfile.zip file

HOMEWORKS

HW 1 : Read contents of readfromfile.java file as Strings, count total number of words, list each word seperately. Write a program to do this.

HW 2 : Temperature and specific heat of CO_2 gas is given in file co2_cv.txt . Read the data, from the file, calculate the avarage temperature and Cp value and print out the result.

16 DATA BASE PROGRAMMING, ACCESSING DATABASES THROUGH JAVA PROGRAMMING

Data base is an organised collection of data. There are several ways to organise data. We have seen some of them while investigating java programming. Today's most popular Data base systems are relational databases. A language called Sequential Querry Language, SQL, sequel or es-qu-el is standart language used almost universally with relational databases to perform queries (request information that satisfy given condition(s)). Individual SQL languages changes a little bit from company to company that created the database program. Some of the most popular ones are Microsoft SQL server, Microsoft Office Access, Oracle, Sybase, IBM DB2, MySQL SQLite, Java DB. In this chapter, SQLite will be used as the basic database program. The main reason to use these program is that they are free and some of the most common free databases in use.

16.1 SQLite DATABASE

SQLite is quite versatile. It is a database, a programming library, and a command-line tool, as well an excellent learning tool that provides a good introduction to relational databases. There are many ways to use it—in embedded environments, websites, operating system services, scripts, and applications. For programmers, SQLite is like "data duct tape," providing an easy way to bind applications and their data.

Like duct tape, there is no end to its potential uses. In a web environment, SQLite can help with managing complex session information. Rather than serializing session data into one big blob, individual pieces can be selectively written to and read from individual session databases. SQLite also serves as a good stand-in relational database for development and testing: there are no external RDBMSs or networking to configure or usernames and passwords to hinder the programmer's focus. SQLite can also serve as a cache, hold configuration data, or, by leveraging its binary compatibility across platforms, even work as an application file format. The basic SQLite can be obtained from http://www.sqlite.org/download.html. In this internet site documentation is given as

sqlite-doc-3080401.zip (4.39 MiB)	This ZIP archive contains most of the static HTML files that comprise this website, including all of the SQL Syntax and the C/C++ interface specs and other miscellaneous documentation.

Precompiled binaries for Microsoft computers are given as:

sqlite-shell-win32-x86-3080401.zip	A command-line shell for accessing and modifying SQLite databases. This program is compatible with all versions of and beyond.

(289.44 KiB)	(sha1: 73559b6b007cee17e531784640203f99dfca1472)
sqlite-dll-win32-x86-3080401.zip (323.28 KiB)	This ZIP archive contains a DLL for the SQLite library version 3.8.4.1 for 32-bit x86 processors using the Win32 API using SQLITE_ENABLE_COLUMN_METADATA so that it is suitable for use with Ruby on Rails. (sha1: d266bc1551f99222cbdd2d3e642aa83e7c4fd1a6)
sqlite-analyzer-win32-x86-3080401.zip (679.86 KiB)	An analysis program for database files compatible with all SQLite versions through 3.8.4.1 and beyond. (sha1: b01c0cac66421bf751b2ea06a57c91e331277cd8)

The site also contains precompiled binaries for different machine operating systems such as Linux, Mac. Please refer to the site for the details.

When you opened **sqlite-shell-win32-x86-3080401.zip** an executable program **sqlite3.exe** will be available, by running this program in dos Shell , SQLite database can be directly used.

There are also some freeware programs available to use SQLite commands inside a graphics programming environment. One of this programs is **SQLite2009Pro.exe**. This program can be obtained from **http://osenxpsuite.net/download.htm**

Java language can be connected to SQLite two different way. The first method uses **Java Database Connectivity (JDBC)**. The advantages of using this method is utilising standart java class **java.sql** so that standart input-output comments commn for ecery type of database programming can be utilised. . A recent adress for a JDBC driver is **https://bitbucket.org/xerial/sqlite-jdbc/downloads** Different JDBC - SQLite drivers might existed in internet. The site will give you few options, newest jar file **sqlite-jdbc4-3.8.2-SNAPSHOT.jar** is selected .Copy this file into JAVA_HOME/jre/lib/ext directory to program SQLite in java environment. It should be note that internet adreses of the freware programs can be changed quickly, but a quick search on JDBC SQLite will usually give a new version of JDBC driver for you. Since JDBC is a standard library under java more documentation can be found related to JDBC library.

The second way of connecting java file into the SQLite database is called Native language (C++) library utilisaton programs, they are also called wrappers. Java compiler is written in C++ language, therefore it can communucate directly with C++ native code. Utilisation of this type of environment is faster to process compare to JDBC environment. A wrapper usually has its own command structure, therfore JDBC and wrapper programs should be investigated seperately. One of the best wrapper program could be found at **https://code.google.com/p/sqlite4java/** adress. Latest version (for Windows environment) when this text was written was **sqlite4java-282.** When the jar package openedjar file sqlite4java.jar will be found. This file should be copied into JAVA_HOME/jre/lib/ext directory to program SQLite in java environment. Addition to this two dll files, **sqlite4java-win32-x64.dll** and **sqlite4java-win32-x86.dll** found in this directory. This files should be copied into **Windows/System** or **Windows/System32** directories.

Now let us look at some of the problems defined in SQLite database in this new database environment. Let us assume that a database named **temp.db** will be created in console environment. Under the database temp a table COFFEE will be created. For this porpose SQL command "**create table"** can be used.

```
sqlite>
sqlite> create table COFFEE(COFFEE_NAME String, CODE int,PRICE double,AMOUNT int,TOTAL double);
sqlite> .separator ,
sqlite> .mode column
sqlite> .headers on
sqlite> .import a.txt COFFEE
sqlite> select * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
colombian    101         7.99        959         959.0
sqlite>
```

Please note that in this database, variable names are similar to the Java equivalents. In order to read data from a text file into the database table **.separator ,** comment is given to define text data seperator as comma and then by using command **.import a.txt COFFEE** data is read from text file a.txt into the database table. **".headers on**" is format statement to list headers with select statement and **".mode column**" list select statement output in column(table) form. Note that the number of items should be equal in TABLE and text file or database will give us an error statement. Text file used in this example has the following form:

SQLight_Table.java | data.java | RTGraph.java | cubicE

```
1 French,49,8.99,130,1169
2 decafeiniated french,49,9.99,105,1069
3 colombian,101,7.99,959,959
4
```

Output from the database table is taken by using **select * from COFFEE** SQL statement. The data can also be inserted into the database table by **using insert into COFFEE values('espresso',150,9.99,100,0);** SQL comment.

```
sqlite> insert into COFFEE values('espresso',150,9.99,110,0);
sqlite> select * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
colombian    101         7.99        959         959.0
espresso     150         9.99        110         0.0
sqlite>
```

The general way of the term SELECT is as follows:

SELECT [DISTINCT|ALL] {*| liste}
FROM [tablo ismi][, ikinci tablo ismi]….
[WHERE condition term]
[GROUP BY colomn list]
[HAVING condition term]
[ORDER BY colomn list]

In the following example a second case of espresso is added to the table then select statement is used with distinct and all keywords.

```
sqlite> insert into COFFEE values('espresso',150,9.99,110,0);
sqlite> select * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
colombian    101         7.99        959         959.0
espresso     150         9.99        110         0.0
espresso     150         9.99        110         0.0
sqlite> select distinct * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
colombian    101         7.99        959         959.0
espresso     150         9.99        110         0.0
sqlite> select all * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
colombian    101         7.99        959         959.0
espresso     150         9.99        110         0.0
espresso     150         9.99        110         0.0
```

The effect of ascent and where key words is shown in the following example

```
sqlite> select distinct * from COFFEE where COFFEE_NAME like '%French%';
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
decafeiniat  49          9.99        105         1069.0
sqlite> select distinct * from COFFEE order by COFFEE_NAME asc;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1169.0
colombian    101         7.99        959         959.0
decafeiniat  49          9.99        105         1069.0
espresso     150         9.99        110         0.0
sqlite> select distinct * from COFFEE order by COFFEE_NAME des;
```

The following commands used together with WHERE statement to create a boolean condition (comparisons)

AND

OR

NOT

BETWEEN (used usually together with AND , OR ;NOT type of statement for example :BETWEEN 20 AND 30)

LIKE

LIKE '%French%'

NULL (empty)

ANY

ALL

EXIST

Number comparisons:

= :equal

<> :not equal

> : greater than

< : smaller than

>= : greater than or equal

<= smaller than or equal

ORDER BY term sort the table according to fiven column

In order to change data **update** key Word can be used.

```
sqlite> update COFFEE set TOTAL=PRICE*AMOUNT;
sqlite> select distinct * from COFFEE;
COFFEE_NAME  CODE        PRICE       AMOUNT      TOTAL
-----------  ----------  ----------  ----------  ----------
French       49          8.99        130         1168.7
decafeiniat  49          9.99        105         1048.95
colombian    101         7.99        959         7662.41
espresso     150         9.99        110         1098.9
sqlite>
```

By using drop table statement, a table can be deleted.

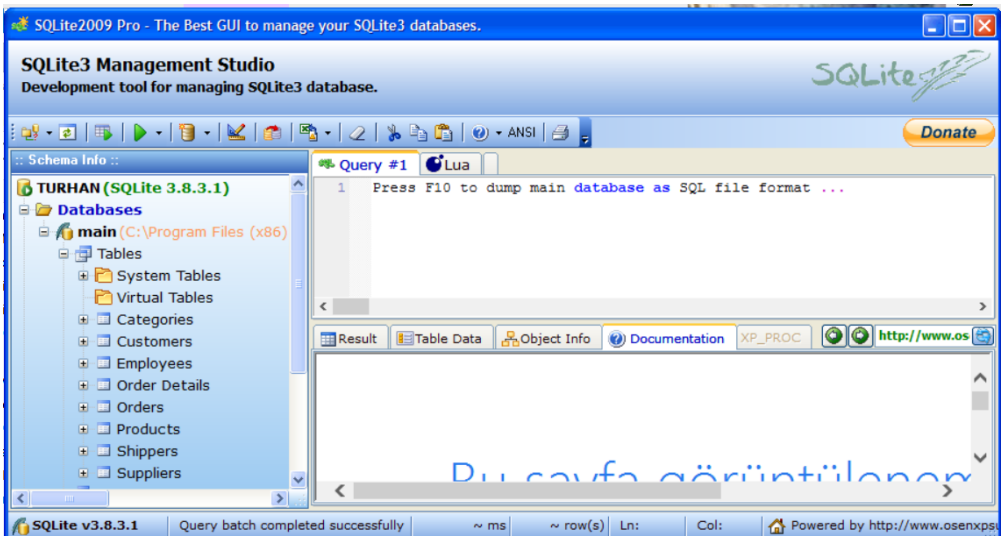
```
sqlite> drop table COFFEE;
sqlite> select * from COFFEE;
Error: no such table: COFFEE
```

Some of the SQLite dos environment commants are listed below:

```
.backup ?DB? FILE      Backup DB (default "main") to FILE
.bail ON|OFF           Stop after hitting an error.  Default OFF
.databases             List names and files of attached databases
.dump ?TABLE? ...      Dump the database in an SQL text format
                       If TABLE specified, only dump tables matching
                       LIKE pattern TABLE.
.echo ON|OFF           Turn command echo on or off
.exit                  Exit this program
.explain ?ON|OFF?      Turn output mode suitable for EXPLAIN on or off.
                       With no args, it turns EXPLAIN on.
.header(s) ON|OFF      Turn display of headers on or off
.help                  Show this message
.import FILE TABLE     Import data from FILE into TABLE
.indices ?TABLE?       Show names of all indices
                       If TABLE specified, only show indices for tables
                       matching LIKE pattern TABLE.
.load FILE ?ENTRY?     Load an extension library
.log FILE|off          Turn logging on or off.  FILE can be stderr/stdout
.mode MODE ?TABLE?     Set output mode where MODE is one of:
                       csv      Comma-separated values
                       column   Left-aligned columns.  (See .width)
                       html     HTML <table> code
                       insert   SQL insert statements for TABLE
                       line     One value per line
                       list     Values delimited by .separator string
                       tabs     Tab-separated values
                       tcl      TCL list elements
.nullvalue STRING      Print STRING in place of NULL values
.output FILENAME       Send output to FILENAME
.output stdout         Send output to the screen
.prompt MAIN CONTINUE  Replace the standard prompts
.quit                  Exit this program
.read FILENAME         Execute SQL in FILENAME
.restore ?DB? FILE     Restore content of DB (default "main") from FILE
.schema ?TABLE?        Show the CREATE statements
                       If TABLE specified, only show tables matching
                       LIKE pattern TABLE.
.separator STRING      Change separator used by output mode and .import
.show                  Show the current values for various settings
.stats ON|OFF          Turn stats on or off
.tables ?TABLE?        List names of tables
                       If TABLE specified, only list tables matching
                       LIKE pattern TABLE.
.timeout MS            Try opening locked tables for MS milliseconds
.width NUM1 NUM2 ...   Set column widths for "column" mode
.timer ON|OFF          Turn the CPU timer measurement on or off
sqlite> .help > a.txt;
.backup ?DB? FILE      Backup DB (default "main") to FILE
.bail ON|OFF           Stop after hitting an error.  Default OFF
.databases             List names and files of attached databases
.dump ?TABLE? ...      Dump the database in an SQL text format
                       If TABLE specified, only dump tables matching
                       LIKE pattern TABLE.
.echo ON|OFF           Turn command echo on or off
.exit                  Exit this program
.explain ?ON|OFF?      Turn output mode suitable for EXPLAIN on or off.
                       With no args, it turns EXPLAIN on.
```

```
.header(s) ON|OFF      Turn display of headers on or off
.help                  Show this message
.import FILE TABLE     Import data from FILE into TABLE
.indices ?TABLE?       Show names of all indices
                       If TABLE specified, only show indices for tables
                       matching LIKE pattern TABLE.
.load FILE ?ENTRY?     Load an extension library
.log FILE|off          Turn logging on or off.  FILE can be stderr/stdout
.mode MODE ?TABLE?     Set output mode where MODE is one of:
                       csv      Comma-separated values
                       column   Left-aligned columns.  (See .width)
                       html     HTML <table> code
                       insert   SQL insert statements for TABLE
                       line     One value per line
                       list     Values delimited by .separator string
                       tabs     Tab-separated values
                       tcl      TCL list elements
.nullvalue STRING      Print STRING in place of NULL values
.output FILENAME       Send output to FILENAME
.output stdout         Send output to the screen
.prompt MAIN CONTINUE  Replace the standard prompts
.quit                  Exit this program
.read FILENAME         Execute SQL in FILENAME
.restore ?DB? FILE     Restore content of DB (default "main") from FILE
.schema ?TABLE?        Show the CREATE statements
                       If TABLE specified, only show tables matching
                       LIKE pattern TABLE.
.separator STRING      Change separator used by output mode and .import
.show                  Show the current values for various settings
.stats ON|OFF          Turn stats on or off
.tables ?TABLE?        List names of tables
                       If TABLE specified, only list tables matching
                       LIKE pattern TABLE.
.timeout MS            Try opening locked tables for MS milliseconds
.width NUM1 NUM2 ...   Set column widths for "column" mode
.timer ON|OFF          Turn the CPU timer measurement on or off
```

Now Let us try the same process by using **SQLite200Pro** program. When this program executed, a graphic program page will apear. Any database and table can be created and executed in this environment also.



16.2 SQLite DATABASE WITH JAVA JAVA DATABASE CONNECTIVITY (JDBC)

It is assumed that **JDBC** is connected to java as its defined in the previous section. A recent adress for a JDBC driver is **https://bitbucket.org/xerial/sqlite-jdbc/downloads**
Different JDBC - SQLite drivers might existed in internet. The site will give you few options, newest jar file **sqlite-jdbc4-3.8.2-SNAPSHOT.jar** is selected .Copy this file into JAVA_HOME/jre/lib/ext directory to program SQLite in java environment

JDBC packages utilises standart Java package **java.sql**, in this package 3 important interface classes are used in each program, Connection, Statement and ResultSet. Detailed description of methods and field variables of these classes are given in the tables below:

```
public interface Connection
extends Wrapper, AutoCloseable
```

Field Summary

Fields

Modifier and Type	Field and Description
static int	TRANSACTION_NONE A constant indicating that transactions are not supported.
static int	TRANSACTION_READ_COMMITTED A constant indicating that dirty reads are prevented; non-repeatable reads and phantom reads can occur.
static int	TRANSACTION_READ_UNCOMMITTED A constant indicating that dirty reads, non-repeatable reads and phantom reads can occur.
static int	TRANSACTION_REPEATABLE_READ A constant indicating that dirty reads and non-repeatable reads are prevented; phantom reads
static int	TRANSACTION_SERIALIZABLE A constant indicating that dirty reads, non-repeatable reads and phantom reads are prevented.

Method Summary

Methods

Modifier and Type	Method and Description
void	abort(Executor executor) Terminates an open connection.
void	clearWarnings() Clears all warnings reported for this Connection object.
void	close() Releases this Connection object's database and JDBC resources immediately instead of waiting for them to be automatically released.
void	commit() Makes all changes made since the previous commit/rollback permanent and releases any database locks currently held by thisConnection object.
Array	createArrayOf(String typeName, Object[] elements) Factory method for creating Array objects.
Blob	createBlob() Constructs an object that implements the Blob interface.
Clob	createClob() Constructs an object that implements the Clob interface.
NClob	createNClob() Constructs an object that implements the NClob interface.
SQLXML	createSQLXML() Constructs an object that implements the SQLXML interface.
Statement	createStatement() Creates a Statement object for sending SQL statements to the database.
Statement	createStatement(int resultSetType, int resultSetConcurrency) Creates a Statement object that will generate ResultSet objects with the given type and concurrency.
Statement	createStatement(int resultSetType, int resultSetConcurrency, int resultSetHoldability) Creates a Statement object that will generate ResultSet objects with the given type, concurrency, and holdability.
Struct	createStruct(String typeName, Object[] attributes)

	Factory method for creating Struct objects.
boolean	getAutoCommit() Retrieves the current auto-commit mode for this Connection object.
String	getCatalog() Retrieves this Connection object's current catalog name.
Properties	getClientInfo() Returns a list containing the name and current value of each client info property supported by the driver.
String	getClientInfo(String name) Returns the value of the client info property specified by name.
int	getHoldability() Retrieves the current holdability of ResultSet objects created using this Connection object.
DatabaseMetaData	getMetaData() Retrieves a DatabaseMetaData object that contains metadata about the database to which this Connection object repre
int	getNetworkTimeout() Retrieves the number of milliseconds the driver will wait for a database request to complete.
String	getSchema() Retrieves this Connection object's current schema name.
int	getTransactionIsolation() Retrieves this Connection object's current transaction isolation level.
Map<String,Class<?>>	getTypeMap() Retrieves the Map object associated with this Connection object.
SQLWarning	getWarnings() Retrieves the first warning reported by calls on this Connection object.
boolean	isClosed() Retrieves whether this Connection object has been closed.
boolean	isReadOnly() Retrieves whether this Connection object is in read-only mode.
boolean	isValid(int timeout) Returns true if the connection has not been closed and is still valid.
String	nativeSQL(String sql) Converts the given SQL statement into the system's native SQL grammar.
CallableStatement	prepareCall(String sql) Creates a CallableStatement object for calling database stored procedures.
CallableStatement	prepareCall(String sql, int resultSetType, int resultSetConcurrency) Creates a CallableStatement object that will generate ResultSet objects with the given type and concurrency.
CallableStatement	prepareCall(String sql, int resultSetType, int resultSetConcurrency, int resultSetHolda Creates a CallableStatement object that will generate ResultSet objects with the given type and concurrency.
PreparedStatement	prepareStatement(String sql) Creates a PreparedStatement object for sending parameterized SQL statements to the database.
PreparedStatement	prepareStatement(String sql, int autoGeneratedKeys) Creates a default PreparedStatement object that has the capability to retrieve auto-generated keys.
PreparedStatement	prepareStatement(String sql, int[] columnIndexes) Creates a default PreparedStatement object capable of returning the auto-generated keys designated by the given array.
PreparedStatement	prepareStatement(String sql, int resultSetType, int resultSetConcurrency) Creates a PreparedStatement object that will generate ResultSet objects with the given type and concurrency.
PreparedStatement	prepareStatement(String sql, int resultSetType, int resultSetConcurrency, int resultSet Creates a PreparedStatement object that will generate ResultSet objects with the given type, concurrency, and holdab
PreparedStatement	prepareStatement(String sql, String[] columnNames) Creates a default PreparedStatement object capable of returning the auto-generated keys designated by the given array.
void	releaseSavepoint(Savepoint savepoint) Removes the specified Savepoint and subsequent Savepoint objects from the current transaction.
void	rollback()

	Undoes all changes made in the current transaction and releases any database locks currently held by this `Connection` object
void	**rollback**(**Savepoint** savepoint) Undoes all changes made after the given `Savepoint` object was set.
void	**setAutoCommit**(boolean autoCommit) Sets this connection's auto-commit mode to the given state.
void	**setCatalog**(**String** catalog) Sets the given catalog name in order to select a subspace of this `Connection` object's database in which to work.
void	**setClientInfo**(**Properties** properties) Sets the value of the connection's client info properties.
void	**setClientInfo**(**String** name, **String** value) Sets the value of the client info property specified by name to the value specified by value.
void	**setHoldability**(int holdability) Changes the default holdability of `ResultSet` objects created using this `Connection` object to the given holdability.
void	**setNetworkTimeout**(**Executor** executor, int milliseconds) Sets the maximum period a `Connection` or objects created from the `Connection` will wait for the database to reply to any
void	**setReadOnly**(boolean readOnly) Puts this connection in read-only mode as a hint to the driver to enable database optimizations.
Savepoint	**setSavepoint**() Creates an unnamed savepoint in the current transaction and returns the new `Savepoint` object that represents it.
Savepoint	**setSavepoint**(**String** name) Creates a savepoint with the given name in the current transaction and returns the new `Savepoint` object that represents it.
void	**setSchema**(**String** schema) Sets the given schema name to access.
void	**setTransactionIsolation**(int level) Attempts to change the transaction isolation level for this `Connection` object to the one given.
void	**setTypeMap**(**Map**<**String**,**Class**<?>> map) Installs the given `TypeMap` object as the type map for this `Connection` object.

```
public interface Statement
extends Wrapper, AutoCloseable
```

Field Summary

Fields

Modifier and Type	Field and Description
static int	**CLOSE_ALL_RESULTS** The constant indicating that all `ResultSet` objects that have previously been kept open should be closed when calling `getMoreResults`.
static int	**CLOSE_CURRENT_RESULT** The constant indicating that the current `ResultSet` object should be closed when calling `getMoreResults`.
static int	**EXECUTE_FAILED** The constant indicating that an error occured while executing a batch statement.
static int	**KEEP_CURRENT_RESULT** The constant indicating that the current `ResultSet` object should not be closed when calling `getMoreResults`.
static int	**NO_GENERATED_KEYS** The constant indicating that generated keys should not be made available for retrieval.
static int	**RETURN_GENERATED_KEYS** The constant indicating that generated keys should be made available for retrieval.
static int	**SUCCESS_NO_INFO** The constant indicating that a batch statement executed successfully but that no count of the number of rows it affected is available.

Method Summary

Methods	
Modifier and Type	**Method and Description**
void	addBatch(String sql) Adds the given SQL command to the current list of commmands for this Statement object.
void	cancel() Cancels this Statement object if both the DBMS and driver support aborting an SQL statement.
void	clearBatch() Empties this Statement object's current list of SQL commands.
void	clearWarnings() Clears all the warnings reported on this Statement object.
void	close() Releases this Statement object's database and JDBC resources immediately instead of waiting for this
void	closeOnCompletion() Specifies that this Statement will be closed when all its dependent result sets are closed.
boolean	execute(String sql) Executes the given SQL statement, which may return multiple results.
boolean	execute(String sql, int autoGeneratedKeys) Executes the given SQL statement, which may return multiple results, and signals the driver that any au
boolean	execute(String sql, int[] columnIndexes) Executes the given SQL statement, which may return multiple results, and signals the driver that the aut for retrieval.
boolean	execute(String sql, String[] columnNames) Executes the given SQL statement, which may return multiple results, and signals the driver that the aut for retrieval.
int[]	executeBatch() Submits a batch of commands to the database for execution and if all commands execute successfully, r
ResultSet	executeQuery(String sql) Executes the given SQL statement, which returns a single ResultSet object.
int	executeUpdate(String sql) Executes the given SQL statement, which may be an INSERT, UPDATE, or DELETE statement or an SQ
int	executeUpdate(String sql, int autoGeneratedKeys) Executes the given SQL statement and signals the driver with the given flag about whether the auto-gen available for retrieval.
int	executeUpdate(String sql, int[] columnIndexes) Executes the given SQL statement and signals the driver that the auto-generated keys indicated in the g
int	executeUpdate(String sql, String[] columnNames) Executes the given SQL statement and signals the driver that the auto-generated keys indicated in the g
Connection	getConnection() Retrieves the Connection object that produced this Statement object.
int	getFetchDirection() Retrieves the direction for fetching rows from database tables that is the default for result sets generate
int	getFetchSize() Retrieves the number of result set rows that is the default fetch size for ResultSet objects generated
ResultSet	getGeneratedKeys() Retrieves any auto-generated keys created as a result of executing this Statement object.
int	getMaxFieldSize() Retrieves the maximum number of bytes that can be returned for character and binary column values in
int	getMaxRows() Retrieves the maximum number of rows that a ResultSet object produced by this Statement obje
boolean	getMoreResults()

	Moves to this `Statement` object's next result, returns `true` if it is a `ResultSet` object, and implicit method `getResultSet`.
`boolean`	**`getMoreResults`**`(int current)` Moves to this `Statement` object's next result, deals with any current `ResultSet` object(s) according next result is a `ResultSet` object.
`int`	**`getQueryTimeout`**`()` Retrieves the number of seconds the driver will wait for a `Statement` object to execute.
`ResultSet`	**`getResultSet`**`()` Retrieves the current result as a `ResultSet` object.
`int`	**`getResultSetConcurrency`**`()` Retrieves the result set concurrency for `ResultSet` objects generated by this `Statement` object.
`int`	**`getResultSetHoldability`**`()` Retrieves the result set holdability for `ResultSet` objects generated by this `Statement` object.
`int`	**`getResultSetType`**`()` Retrieves the result set type for `ResultSet` objects generated by this `Statement` object.
`int`	**`getUpdateCount`**`()` Retrieves the current result as an update count; if the result is a `ResultSet` object or there are no mo
`SQLWarning`	**`getWarnings`**`()` Retrieves the first warning reported by calls on this `Statement` object.
`boolean`	**`isClosed`**`()` Retrieves whether this `Statement` object has been closed.
`boolean`	**`isCloseOnCompletion`**`()` Returns a value indicating whether this `Statement` will be closed when all its dependent result sets ar
`boolean`	**`isPoolable`**`()` Returns a value indicating whether the `Statement` is poolable or not.
`void`	**`setCursorName`**`(String name)` Sets the SQL cursor name to the given `String`, which will be used by subsequent `Statement` object
`void`	**`setEscapeProcessing`**`(boolean enable)` Sets escape processing on or off.
`void`	**`setFetchDirection`**`(int direction)` Gives the driver a hint as to the direction in which rows will be processed in `ResultSet` objects create
`void`	**`setFetchSize`**`(int rows)` Gives the JDBC driver a hint as to the number of rows that should be fetched from the database when m this `Statement`.
`void`	**`setMaxFieldSize`**`(int max)` Sets the limit for the maximum number of bytes that can be returned for character and binary column v
`void`	**`setMaxRows`**`(int max)` Sets the limit for the maximum number of rows that any `ResultSet` object generated by this `Stater`
`void`	**`setPoolable`**`(boolean poolable)` Requests that a `Statement` be pooled or not pooled.
`void`	**`setQueryTimeout`**`(int seconds)` Sets the number of seconds the driver will wait for a `Statement` object to execute to the given numbe

```
public interface ResultSet
extends Wrapper, AutoCloseable
```

Field Summary

Fields

Modifier and Type	Field and Description
`static int`	**`CLOSE_CURSORS_AT_COMMIT`** The constant indicating that open `ResultSet` objects with this holdability will be closed when the current transaction is commited.

static int	**CONCUR_READ_ONLY** The constant indicating the concurrency mode for a ResultSet object that may NOT be updated.
static int	**CONCUR_UPDATABLE** The constant indicating the concurrency mode for a ResultSet object that may be updated.
static int	**FETCH_FORWARD** The constant indicating that the rows in a result set will be processed in a forward direction; first-to-last.
static int	**FETCH_REVERSE** The constant indicating that the rows in a result set will be processed in a reverse direction; last-to-first.
static int	**FETCH_UNKNOWN** The constant indicating that the order in which rows in a result set will be processed is unknown.
static int	**HOLD_CURSORS_OVER_COMMIT** The constant indicating that open ResultSet objects with this holdability will remain open when the current transaction is commited.
static int	**TYPE_FORWARD_ONLY** The constant indicating the type for a ResultSet object whose cursor may move only forward.
static int	**TYPE_SCROLL_INSENSITIVE** The constant indicating the type for a ResultSet object that is scrollable but generally not sensitive to changes to the data that underlies the ResultSet.
static int	**TYPE_SCROLL_SENSITIVE** The constant indicating the type for a ResultSet object that is scrollable and generally sensitive to changes to the data that underlies theResultSet.

Method Summary

Methods

Modifier and Type	Method and Description
boolean	**absolute**(int row) Moves the cursor to the given row number in this ResultSet object.
void	**afterLast**() Moves the cursor to the end of this ResultSet object, just after the last row.
void	**beforeFirst**() Moves the cursor to the front of this ResultSet object, just before the first row.
void	**cancelRowUpdates**() Cancels the updates made to the current row in this ResultSet object.
void	**clearWarnings**() Clears all warnings reported on this ResultSet object.
void	**close**() Releases this ResultSet object's database and JDBC resources immediately instead of waiting for this to happen when it is automatically closed.
void	**deleteRow**() Deletes the current row from this ResultSet object and from the underlying database.
int	**findColumn**(String columnLabel) Maps the given ResultSet column label to its ResultSet column index.
boolean	**first**() Moves the cursor to the first row in this ResultSet object.
Array	**getArray**(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as an Array object in the Java programming language.
Array	**getArray**(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as an Array object in the Java program
InputStream	**getAsciiStream**(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a stream of ASCII characters.

InputStream	getAsciiStream(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a stream of ASCII characters.
BigDecimal	getBigDecimal(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.math.BigDecimal with full precision.
BigDecimal	getBigDecimal(int columnIndex, int scale) **Deprecated.**
BigDecimal	getBigDecimal(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.math.BigDecimal with full precision.
BigDecimal	getBigDecimal(String columnLabel, int scale) **Deprecated.**
InputStream	getBinaryStream(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a stream of uninterpreted bytes.
InputStream	getBinaryStream(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a stream of uninterpreted bytes.
Blob	getBlob(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a Blob object in the Java programming language.
Blob	getBlob(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a Blob object in the Java programming language.
boolean	getBoolean(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a boolean in the Java programming language.
boolean	getBoolean(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a boolean in the Java programming language.
byte	getByte(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a byte in the Java programming language.
byte	getByte(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a byte in the Java programming language.
byte[]	getBytes(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a byte array in the Java programmin
byte[]	getBytes(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a byte array in the Java programming language.
Reader	getCharacterStream(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.io.Reader object.
Reader	getCharacterStream(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.io.Reader object.
Clob	getClob(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a Clob object in the Java programming language.
Clob	getClob(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a Clob object in the Java programming language.
int	getConcurrency() Retrieves the concurrency mode of this ResultSet object.
String	getCursorName()

	Retrieves the name of the SQL cursor used by this `ResultSet` object.
Date	**getDate**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.sql.Date` object in the Java programming language.
Date	**getDate**(int columnIndex, **Calendar** cal) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.sql.Date` object n the Java programming language.
Date	**getDate**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.sql.Date` object in the Java programming language.
Date	**getDate**(**String** columnLabel, **Calendar** cal) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.sql.Date` object in the Jav
double	**getDouble**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `double` in the Java programming language.
double	**getDouble**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `double` in the Java programming language.
int	**getFetchDirection**() Retrieves the fetch direction for this `ResultSet` object.
int	**getFetchSize**() Retrieves the fetch size for this `ResultSet` object.
float	**getFloat**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `float` in the Java programming language.
float	**getFloat**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `float` in the Java programming language.
int	**getHoldability**() Retrieves the holdability of this `ResultSet` object
int	**getInt**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as an `int` in the Java programming language.
int	**getInt**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as an `int` in the Java programming language.
long	**getLong**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `long` in the Java programming language.
long	**getLong**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `long` in the Java programming language.
ResultSetMetaData	**getMetaData**() Retrieves the number, types and properties of this `ResultSet` object's columns.
Reader	**getNCharacterStream**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.io.Reader` object.
Reader	**getNCharacterStream**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `java.io.Reader` object.
NClob	**getNClob**(int columnIndex) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `NClob` object in the Java programming language.
NClob	**getNClob**(**String** columnLabel) Retrieves the value of the designated column in the current row of this `ResultSet` object as a `NClob` object in the Java programming language.

String	getNString(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java programming language.
String	getNString(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java programming language.
Object	getObject(int columnIndex) Gets the value of the designated column in the current row of this ResultSet object as an Object in the Java programming language.
<T> T	getObject(int columnIndex, Class<T> type) Retrieves the value of the designated column in the current row of this ResultSet object and will convert from the SQL type of the column to the requested Java data type, if the conversion is supported.
Object	getObject(int columnIndex, Map<String,Class<?>> map) Retrieves the value of the designated column in the current row of this ResultSet object as an Object in the Java programming language.
Object	getObject(String columnLabel) Gets the value of the designated column in the current row of this ResultSet object as an Object in the Java programming language.
<T> T	getObject(String columnLabel, Class<T> type) Retrieves the value of the designated column in the current row of this ResultSet object and will convert from the SQL type of the column to the requested Java data type, if the conversion is supported.
Object	getObject(String columnLabel, Map<String,Class<?>> map) Retrieves the value of the designated column in the current row of this ResultSet object as an Object in the Java programming language.
Ref	getRef(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a Ref object in the Java programming language.
Ref	getRef(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a Ref object in the Java programmin
int	getRow() Retrieves the current row number.
RowId	getRowId(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.RowId object in the Java programming language.
RowId	getRowId(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.RowId object in the Java programming language.
short	getShort(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a short in the Java programming language.
short	getShort(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a short in the Java programming lar
SQLXML	getSQLXML(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet as a java.sql.SQLXML object in the Java programming language.
SQLXML	getSQLXML(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet as a java.sql.SQLXML object in the Java programming language.
Statement	getStatement() Retrieves the Statement object that produced this ResultSet object.
String	getString(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java programming language.
String	getString(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a String in the Java

	programming language.
Time	getTime(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Time object in the Java programming language.
Time	getTime(int columnIndex, Calendar cal) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Time object in the Java programming language.
Time	getTime(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Time object in the Java programming language.
Time	getTime(String columnLabel, Calendar cal) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Time object in the Java programming language.
Timestamp	getTimestamp(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Timestamp object in the Java programming language.
Timestamp	getTimestamp(int columnIndex, Calendar cal) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Timestamp object in the Java programming language.
Timestamp	getTimestamp(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Timestamp object in the Java programming language.
Timestamp	getTimestamp(String columnLabel, Calendar cal) Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Timestamp object in the Java programming language.
int	getType() Retrieves the type of this ResultSet object.
InputStream	getUnicodeStream(int columnIndex) **Deprecated.** *use getCharacterStream in place of getUnicodeStream*
InputStream	getUnicodeStream(String columnLabel) **Deprecated.** *use getCharacterStream instead*
URL	getURL(int columnIndex) Retrieves the value of the designated column in the current row of this ResultSet object as a java.net.URL object in the Java programming language.
URL	getURL(String columnLabel) Retrieves the value of the designated column in the current row of this ResultSet object as a java.net.URL object in the Java programming language.
SQLWarning	getWarnings() Retrieves the first warning reported by calls on this ResultSet object.
void	insertRow() Inserts the contents of the insert row into this ResultSet object and into the database.
boolean	isAfterLast() Retrieves whether the cursor is after the last row in this ResultSet object.
boolean	isBeforeFirst() Retrieves whether the cursor is before the first row in this ResultSet object.
boolean	isClosed() Retrieves whether this ResultSet object has been closed.
boolean	isFirst() Retrieves whether the cursor is on the first row of this ResultSet object.
boolean	isLast() Retrieves whether the cursor is on the last row of this ResultSet object.
boolean	last()

	Moves the cursor to the last row in this `ResultSet` object.
void	**moveToCurrentRow**() Moves the cursor to the remembered cursor position, usually the current row.
void	**moveToInsertRow**() Moves the cursor to the insert row.
boolean	**next**() Moves the cursor froward one row from its current position.
boolean	**previous**() Moves the cursor to the previous row in this `ResultSet` object.
void	**refreshRow**() Refreshes the current row with its most recent value in the database.
boolean	**relative**(int rows) Moves the cursor a relative number of rows, either positive or negative.
boolean	**rowDeleted**() Retrieves whether a row has been deleted.
boolean	**rowInserted**() Retrieves whether the current row has had an insertion.
boolean	**rowUpdated**() Retrieves whether the current row has been updated.
void	**setFetchDirection**(int direction) Gives a hint as to the direction in which the rows in this `ResultSet` object will be processed.
void	**setFetchSize**(int rows) Gives the JDBC driver a hint as to the number of rows that should be fetched from the database when more rows are needed for this
void	**updateArray**(int columnIndex, **Array** x) Updates the designated column with a `java.sql.Array` value.
void	**updateArray**(**String** columnLabel, **Array** x) Updates the designated column with a `java.sql.Array` value.
void	**updateAsciiStream**(int columnIndex, **InputStream** x) Updates the designated column with an ascii stream value.
void	**updateAsciiStream**(int columnIndex, **InputStream** x, int length) Updates the designated column with an ascii stream value, which will have the specified number of bytes.
void	**updateAsciiStream**(int columnIndex, **InputStream** x, long length) Updates the designated column with an ascii stream value, which will have the specified number of bytes.
void	**updateAsciiStream**(**String** columnLabel, **InputStream** x) Updates the designated column with an ascii stream value.
void	**updateAsciiStream**(**String** columnLabel, **InputStream** x, int length) Updates the designated column with an ascii stream value, which will have the specified number of bytes.
void	**updateAsciiStream**(**String** columnLabel, **InputStream** x, long length) Updates the designated column with an ascii stream value, which will have the specified number of bytes.
void	**updateBigDecimal**(int columnIndex, **BigDecimal** x) Updates the designated column with a `java.math.BigDecimal` value.
void	**updateBigDecimal**(**String** columnLabel, **BigDecimal** x) Updates the designated column with a `java.sql.BigDecimal` value.
void	**updateBinaryStream**(int columnIndex, **InputStream** x) Updates the designated column with a binary stream value.
void	**updateBinaryStream**(int columnIndex, **InputStream** x, int length) Updates the designated column with a binary stream value, which will have the specified number of bytes.
void	**updateBinaryStream**(int columnIndex, **InputStream** x, long length) Updates the designated column with a binary stream value, which will have the specified number of bytes.
void	**updateBinaryStream**(**String** columnLabel, **InputStream** x) Updates the designated column with a binary stream value.
void	**updateBinaryStream**(**String** columnLabel, **InputStream** x, int length) Updates the designated column with a binary stream value, which will have the specified number of bytes.

void	updateBinaryStream(String columnLabel, InputStream x, long length) Updates the designated column with a binary stream value, which will have the specified number of bytes.
void	updateBlob(int columnIndex, Blob x) Updates the designated column with a java.sql.Blob value.
void	updateBlob(int columnIndex, InputStream inputStream) Updates the designated column using the given input stream.
void	updateBlob(int columnIndex, InputStream inputStream, long length) Updates the designated column using the given input stream, which will have the specified number of bytes.
void	updateBlob(String columnLabel, Blob x) Updates the designated column with a java.sql.Blob value.
void	updateBlob(String columnLabel, InputStream inputStream) Updates the designated column using the given input stream.
void	updateBlob(String columnLabel, InputStream inputStream, long length) Updates the designated column using the given input stream, which will have the specified number of bytes.
void	updateBoolean(int columnIndex, boolean x) Updates the designated column with a boolean value.
void	updateBoolean(String columnLabel, boolean x) Updates the designated column with a boolean value.
void	updateByte(int columnIndex, byte x) Updates the designated column with a byte value.
void	updateByte(String columnLabel, byte x) Updates the designated column with a byte value.
void	updateBytes(int columnIndex, byte[] x) Updates the designated column with a byte array value.
void	updateBytes(String columnLabel, byte[] x) Updates the designated column with a byte array value.
void	updateCharacterStream(int columnIndex, Reader x) Updates the designated column with a character stream value.
void	updateCharacterStream(int columnIndex, Reader x, int length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	updateCharacterStream(int columnIndex, Reader x, long length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	updateCharacterStream(String columnLabel, Reader reader) Updates the designated column with a character stream value.
void	updateCharacterStream(String columnLabel, Reader reader, int length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	updateCharacterStream(String columnLabel, Reader reader, long length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	updateClob(int columnIndex, Clob x) Updates the designated column with a java.sql.Clob value.
void	updateClob(int columnIndex, Reader reader) Updates the designated column using the given Reader object.
void	updateClob(int columnIndex, Reader reader, long length) Updates the designated column using the given Reader object, which is the given number of characters long.
void	updateClob(String columnLabel, Clob x) Updates the designated column with a java.sql.Clob value.
void	updateClob(String columnLabel, Reader reader) Updates the designated column using the given Reader object.
void	updateClob(String columnLabel, Reader reader, long length) Updates the designated column using the given Reader object, which is the given number of characters long.
void	updateDate(int columnIndex, Date x) Updates the designated column with a java.sql.Date value.

void	**updateDate**(String columnLabel, Date x) Updates the designated column with a `java.sql.Date` value.
void	**updateDouble**(int columnIndex, double x) Updates the designated column with a `double` value.
void	**updateDouble**(String columnLabel, double x) Updates the designated column with a `double` value.
void	**updateFloat**(int columnIndex, float x) Updates the designated column with a `float` value.
void	**updateFloat**(String columnLabel, float x) Updates the designated column with a `float` value.
void	**updateInt**(int columnIndex, int x) Updates the designated column with an `int` value.
void	**updateInt**(String columnLabel, int x) Updates the designated column with an `int` value.
void	**updateLong**(int columnIndex, long x) Updates the designated column with a `long` value.
void	**updateLong**(String columnLabel, long x) Updates the designated column with a `long` value.
void	**updateNCharacterStream**(int columnIndex, Reader x) Updates the designated column with a character stream value.
void	**updateNCharacterStream**(int columnIndex, Reader x, long length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	**updateNCharacterStream**(String columnLabel, Reader reader) Updates the designated column with a character stream value.
void	**updateNCharacterStream**(String columnLabel, Reader reader, long length) Updates the designated column with a character stream value, which will have the specified number of bytes.
void	**updateNClob**(int columnIndex, NClob nClob) Updates the designated column with a `java.sql.NClob` value.
void	**updateNClob**(int columnIndex, Reader reader) Updates the designated column using the given `Reader` The data will be read from the stream as needed until end-of-stream is rea
void	**updateNClob**(int columnIndex, Reader reader, long length) Updates the designated column using the given `Reader` object, which is the given number of characters long.
void	**updateNClob**(String columnLabel, NClob nClob) Updates the designated column with a `java.sql.NClob` value.
void	**updateNClob**(String columnLabel, Reader reader) Updates the designated column using the given `Reader` object.
void	**updateNClob**(String columnLabel, Reader reader, long length) Updates the designated column using the given `Reader` object, which is the given number of characters long.
void	**updateNString**(int columnIndex, String nString) Updates the designated column with a `String` value.
void	**updateNString**(String columnLabel, String nString) Updates the designated column with a `String` value.
void	**updateNull**(int columnIndex) Updates the designated column with a `null` value.
void	**updateNull**(String columnLabel) Updates the designated column with a `null` value.
void	**updateObject**(int columnIndex, Object x) Updates the designated column with an `Object` value.
void	**updateObject**(int columnIndex, Object x, int scaleOrLength) Updates the designated column with an `Object` value.
void	**updateObject**(String columnLabel, Object x) Updates the designated column with an `Object` value.

void	updateObject(String columnLabel, Object x, int scaleOrLength) Updates the designated column with an Object value.
void	updateRef(int columnIndex, Ref x) Updates the designated column with a java.sql.Ref value.
void	updateRef(String columnLabel, Ref x) Updates the designated column with a java.sql.Ref value.
void	updateRow() Updates the underlying database with the new contents of the current row of this ResultSet object.
void	updateRowId(int columnIndex, RowId x) Updates the designated column with a RowId value.
void	updateRowId(String columnLabel, RowId x) Updates the designated column with a RowId value.
void	updateShort(int columnIndex, short x) Updates the designated column with a short value.
void	updateShort(String columnLabel, short x) Updates the designated column with a short value.
void	updateSQLXML(int columnIndex, SQLXML xmlObject) Updates the designated column with a java.sql.SQLXML value.
void	updateSQLXML(String columnLabel, SQLXML xmlObject) Updates the designated column with a java.sql.SQLXML value.
void	updateString(int columnIndex, String x) Updates the designated column with a String value.
void	updateString(String columnLabel, String x) Updates the designated column with a String value.
void	updateTime(int columnIndex, Time x) Updates the designated column with a java.sql.Time value.
void	updateTime(String columnLabel, Time x) Updates the designated column with a java.sql.Time value.
void	updateTimestamp(int columnIndex, Timestamp x) Updates the designated column with a java.sql.Timestamp value.
void	updateTimestamp(String columnLabel, Timestamp x) Updates the designated column with a java.sql.Timestamp value.
boolean	wasNull() Reports whether the last column read had a value of SQL NULL.

Example programs given to explain the case better. In the example programs a database table COFFEE will be created and utilised.

Program 16.1-1 COFFEE.**java**

```
 import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class COFFEE
{
  public static void main(String[] args) throws ClassNotFoundException
  {
    // load the sqlite-JDBC driver using the current class loader
    Class.forName("org.sqlite.JDBC");
    Connection connection = null;
    try
    {
      // create a database connection
      connection = DriverManager.getConnection("jdbc:sqlite:temp.db");
```

```
      Statement statement = connection.createStatement();
      statement.setQueryTimeout(30);  // set timeout to 30 sec.

      statement.executeUpdate("drop table if exists COFFEE");
      statement.executeUpdate("create table COFFEE(COFFEE_NAME String, CODE int, PRICE double, AMOUNT
int,TOTAL double)");
      statement.executeUpdate("insert into COFFEE values('espresso',150,9.99,110,0);");
      statement.executeUpdate("insert into COFFEE values('Venezuella',173,12.99,57,0);");
      statement.executeUpdate("insert into COFFEE values('Brasilian',123,11.56,39,0);");
      statement.executeUpdate("insert into COFFEE values('Colombian',111,15.99,20,0);");
      statement.executeUpdate("update COFFEE set TOTAL=PRICE*AMOUNT;");
      ResultSet rs = statement.executeQuery("select * from COFFEE");
      while(rs.next())
      {
        // read the result set
        System.out.print("COFFEE_NAME = " + rs.getString("COFFEE_NAME"));
        System.out.print(" CODE = " + rs.getInt("CODE"));
        System.out.print(" PRICE = " + rs.getInt("PRICE"));
        System.out.print(" AMOUNT = " + rs.getInt("AMOUNT"));
        System.out.println(" TOTAL = " + rs.getString("TOTAL"));
      }
    }
    catch(SQLException e)
    {
      // if the error message is "out of memory",
      // it probably means no database file is found
      System.err.println(e.getMessage());
    }
    finally
    {
      try
      {
        if(connection != null)
          connection.close();
      }
      catch(SQLException e)
      {
        // connection close failed.
        System.err.println(e);
      }
    }
  }
}
```

```
 ---------- Capture Output ----------
> "C:\java\bin\java.exe" COFFEE
COFFEE_NAME = espresso CODE = 150 PRICE = 9 AMOUNT = 110 TOTAL = 1098.9
COFFEE_NAME = Venezuella CODE = 173 PRICE = 12 AMOUNT = 57 TOTAL = 740.43
COFFEE_NAME = Brasilian CODE = 123 PRICE = 11 AMOUNT = 39 TOTAL = 450.84
COFFEE_NAME = Colombian CODE = 111 PRICE = 15 AMOUNT = 20 TOTAL = 319.8

> Terminated with exit code 0.
```

As it is seen from the program it connects to SQLite database by using
Class.forName("org.sqlite.JDBC");
Connection connection == DriverManager.getConnection("jdbc:sqlite:temp.db");
temp.db is the database to be connected and the name of the example table is also given as COFFEE.

As a second example a graphic database table is created by using java program

Program 16.2-2 SQLite_Table.java Program 16.2-7 SQLite_Table.java, for graphic SQLite table

```
import java.sql.SQLException;
import java.sql.Statement;

public class SQLite_Table extends JFrame
```

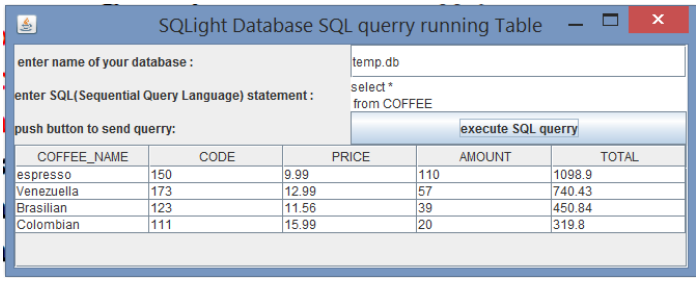
```
{
  JTextField hostField;
  JTextField tableField;
  JTextArea queryField;
  QueryTableModel qtm;
  String s="jdbc:sqlite:";

  static final long serialVersionUID = 8597941330711875390L;
  public SQLite_Table() throws ClassNotFoundException
  {
                    super("SQLight Database SQL querry running Table");
                    Class.forName("org.sqlite.JDBC");
                    Connection connection = null;
    //addWindowListener(new BasicWindowMonitor());
        setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       setSize(500,300);
    qtm=new QueryTableModel();
    JTable table=new JTable(qtm);
    JScrollPane scrollpane=new JScrollPane(table);
    JPanel p1=new JPanel();
    p1.setLayout(new GridLayout(3,2));
    p1.add(new JLabel(" enter name of your database :"));
    p1.add(hostField=new JTextField("temp.db"));
    p1.add(new JLabel("enter SQL(Sequential Query Language) statement :"));
    p1.add(queryField=new JTextArea("select *\n from COFFEE"));
    p1.add(new JLabel("push button to send querry:"));
    JButton jb=new JButton("execute SQL querry");
    jb.addActionListener(new ActionListener()
      {
      public void actionPerformed(ActionEvent e)
        {
        qtm.setHostURL(s+hostField.getText().trim());
        qtm.setQuery(queryField.getText().trim());
        }
      }); // jb.addActionlistener
  p1.add(jb);
  getContentPane().add(p1,BorderLayout.NORTH);
  getContentPane().add(scrollpane,BorderLayout.CENTER);
  } //DatabaseTest()

  public static void main(String args[])
  {
    try
    {
                SQLite_Table tt=new SQLite_Table();
     tt.setVisible(true);
    } catch(ClassNotFoundException e)
    {
      // if the error message is "out of memory",
      // it probably means no database file is found
      System.err.println(e.getMessage());
    }
  }
}
```

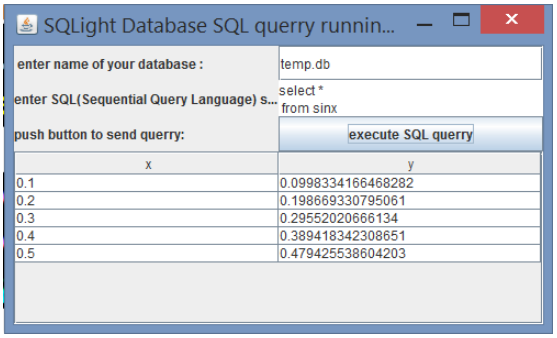
Program 16.2-3 QueryTableModel.java

```
import java.sql.*;
import java.io.*;
import java.util.Vector;
import javax.swing.*;
import javax.swing.table.*;
```

```
public class QueryTableModel extends AbstractTableModel
{
 Vector cache;
 int colCount;
 String[] headers;
 Connection db;
 Statement statement;
 String currentURL;

 public QueryTableModel()
 {
  cache=new Vector();
 }

 public String getColumnName(int i) {return headers[i];}
 public int getColumnCount() {return colCount;}
 public int getRowCount() {return cache.size();}

 public Object getValueAt(int row,int col)
 {
  return ((String[])cache.elementAt(row))[col];
 }

 public void setHostURL(String url)
 {
   if(url.equals(currentURL))
   {return;}
   closeDB();
   initDB(url);
   currentURL=url;
 }

 public void setQuery(String q)
 {
 cache= new Vector();
 try{
    ResultSet rs=statement.executeQuery(q);
    ResultSetMetaData meta=rs.getMetaData();
    colCount=meta.getColumnCount();
    headers=new String[colCount];
    for (int h=1;h<=colCount;h++)
      {
      headers[h-1]=meta.getColumnName(h);
      }
    while(rs.next())
      {
      String[] record=new String[colCount];
      for(int i=0;i<colCount;i++)
        {record[i]=rs.getString(i+1);}
      cache.add(record);
      } //while'in sonu
    fireTableChanged(null);
    } //try'in sonu
    catch(Exception e)
    {
    cache=new Vector();
    e.printStackTrace();
    }
```

```
} // setQuery sonu

public void initDB(String url)
{
try {
   db=DriverManager.getConnection(url);
   statement=db.createStatement();
   }
   catch(Exception e)
   {
    System.out.println("DataBase can not be started");
    e.printStackTrace();
   }
} //initDB sonu

public void closeDB()
{
  try {
     if(statement!= null) {statement.close();}
     if(db != null) {db.close();}
     }
     catch(Exception e)
     {
     System.out.println("database is not closed");
     e.printStackTrace();
     }
}  //closeDB sonu

}
```





As a last program a general input/output class ro read and write double data into a database table is given. This class can also be used to plot data from the database.

Program 16.2-4 SQLite.java, a general utility program read and write data to/from a database table

```
import java.sql.Statement;
import java.sql.*;
import java.util.Vector;
// This class reads, writes and plot data into/from a given SQL Database table
//
public class SQLite
{
// Reads given data column from a specified Lite database table
public static double[][] read_double(String databasename, String tablei,String column[]) throws
```

```
IOException,ClassNotFoundException
{
   // load the sqlite-JDBC driver using the current class loader
   double a[][];
   int n=1,m=1;
   Class.forName("org.sqlite.JDBC");
   Connection connection = null;
   try
   {
     // create a database connection
     connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
     Statement statement = connection.createStatement();
     statement.setQueryTimeout(30);  // set timeout to 30 sec.
     //statement.executeUpdate("drop table if exists H9Ex3");
     ResultSet rs = statement.executeQuery("select * from "+tablei);
     int i=0;
     n=column.length;
     double x;
     while(rs.next())
     {
       x=rs.getDouble(column[0]);
       i++;
     }
     m=i;
   }
   catch(SQLException e)
   {
     // if the error message is "out of memory",
     // it probably means no database file is found
     System.err.println(e.getMessage());
   }
   finally
   {
     try
     {
       if(connection != null)
         connection.close();
     }
     catch(SQLException e)
     {
       // connection close failed.
       System.err.println(e);
     }
   }
   a=new double[n][m];
   try
   {
     // create a database connection
     connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
     Statement statement = connection.createStatement();
     statement.setQueryTimeout(30);  // set timeout to 30 sec.
     //statement.executeUpdate("drop table if exists H9Ex3");
     String s="select ";
     ResultSet rs = statement.executeQuery("select * from "+tablei);
     for(int i=0;i<m;i++)
     {rs.next();
      for(int j=0;j<n;j++)
       {
```

```
                    a[j][i]=rs.getDouble(column[j]);
        }
      }
    }
    catch(SQLException e)
    {
      // if the error message is "out of memory",
      // it probably means no database file is found
      System.err.println(e.getMessage());
    }
    finally
    {
      try
      {
        if(connection != null)
          connection.close();
      }
      catch(SQLException e)
      {
        // connection close failed.
        System.err.println(e);
      }
    }
    return a;
}
//Reads all data columns from a specified Lite database table
public static double[][] read_double(String databasename, String tablei) throws IOException,ClassNotFoundException
{
            // load the sqlite-JDBC driver using the current class loader
    double a[][];
    int n=1,m=1;
    Class.forName("org.sqlite.JDBC");
    Connection connection = null;
    try
    {
      // create a database connection
      connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
      Statement statement = connection.createStatement();
      statement.setQueryTimeout(30);  // set timeout to 30 sec.
      //statement.executeUpdate("drop table if exists H9Ex3");
      ResultSet rs = statement.executeQuery("select * from "+tablei);
      ResultSetMetaData meta=rs.getMetaData();
      int colCount=meta.getColumnCount();
      String column[]=new String[colCount];
      for (int h=0;h<colCount;h++)
      {
      column[h]=meta.getColumnName(h+1);
      }
      int i=0;
      n=colCount;
      double x;
      while(rs.next())
      {
        x=rs.getDouble(column[0]);
        i++;
      }
      m=i;
    }
```

```
catch(SQLException e)
{
  // if the error message is "out of memory",
  // it probably means no database file is found
  System.err.println(e.getMessage());
}
finally
{
  try
  {
    if(connection != null)
      connection.close();
  }
  catch(SQLException e)
  {
    // connection close failed.
    System.err.println(e);
  }
}
a=new double[n][m];
try
{
  // create a database connection
  connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
  Statement statement = connection.createStatement();
  statement.setQueryTimeout(30);  // set timeout to 30 sec.
  //statement.executeUpdate("drop table if exists H9Ex3");
  String s="select ";
  ResultSet rs = statement.executeQuery("select * from "+tablei);
  ResultSetMetaData meta=rs.getMetaData();
  int colCount=meta.getColumnCount();
  String column[]=new String[colCount];
  for (int h=0;h<colCount;h++)
  {
  column[h]=meta.getColumnName(h+1);
  }
  for(int i=0;i<m;i++)
  {rs.next();
   for(int j=0;j<n;j++)
    {
                a[j][i]=rs.getDouble(column[j]);
    }
  }
}
catch(SQLException e)
{
  // if the error message is "out of memory",
  // it probably means no database file is found
  System.err.println(e.getMessage());
}
finally
{
  try
  {
    if(connection != null)
      connection.close();
  }
  catch(SQLException e)
```

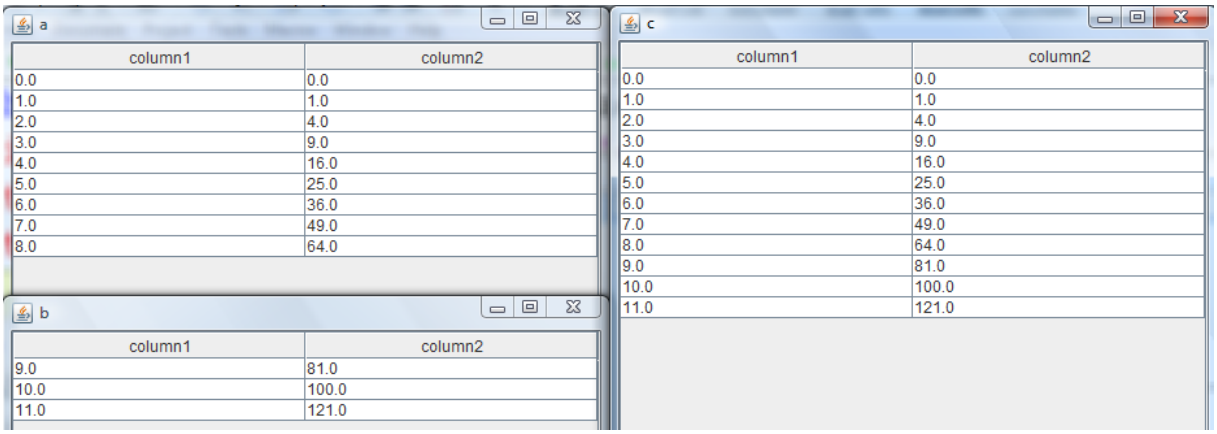
```
      {
        // connection close failed.
        System.err.println(e);
      }
    }
    return a;
}

public static void write_double(String databasename, String tablei,String column[],double a[][]) throws
IOException,ClassNotFoundException
{
    // load the sqlite-JDBC driver using the current class loader
    int n=1,m=1;
    int i=0,j=0;
    Class.forName("org.sqlite.JDBC");
    Connection connection = null;
    try
    {
      // create a database connection
      connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
      Statement statement = connection.createStatement();
      String s="create table "+tablei+"(";
      n=column.length;
      m=a[0].length;
      for(i=0;i<n-1;i++)
      {s+=column[i]+" double, ";}
      s+=column[i]+" double)";
      statement.setQueryTimeout(30);  // set timeout to 30 sec.
      statement.executeUpdate("drop table if exists "+tablei);
      statement.executeUpdate(s);
      String s2="";
      for(j=0;j<a[0].length;j++)
      {s2="";
       for(i=0;i<a.length-1;i++)
       {s2+=""+a[i][j]+",";}
       s2+=""+a[i][j];
       String s3="insert into "+tablei+" values("+s2+");";
       statement.executeUpdate(s3);
      }
    }
    catch(SQLException e)
    {
      // if the error message is "out of memory",
      // it probably means no database file is found
      System.err.println(e.getMessage());
    }
    finally
    {
      try
      {
        if(connection != null)
          connection.close();
      }
      catch(SQLException e)
      {
        // connection close failed.
        System.err.println(e);
      }
```

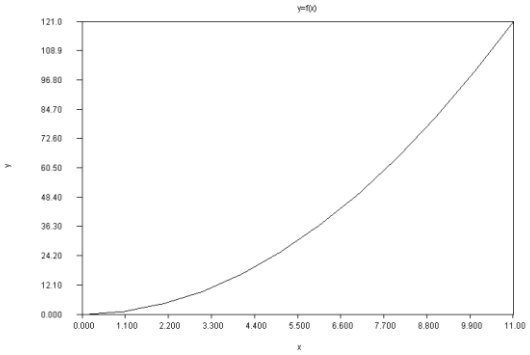
```
    }
}
public static void add_double(String databasename, String tablei,String column[],double a[][]) throws
IOException,ClassNotFoundException
{
    // load the sqlite-JDBC driver using the current class loader
    int n=1,m=1;
    int i=0,j=0;
    Class.forName("org.sqlite.JDBC");
    Connection connection = null;
    try
    {
      // create a database connection
      connection = DriverManager.getConnection("jdbc:sqlite:"+databasename+".db");
      Statement statement = connection.createStatement();
      /*
      String s="create table "+tablei+"(";
      n=column.length;
      m=a[0].length;
      for(i=0;i<n-1;i++)
      {s+=column[i]+" double, ";}
      s+=column[i]+" double)";
      statement.setQueryTimeout(30);  // set timeout to 30 sec.
      statement.executeUpdate("drop table if exists "+tablei);
      statement.executeUpdate(s);
      */
      String s2="";
      for(j=0;j<a[0].length;j++)
      {s2="";
       for(i=0;i<a.length-1;i++)
       {s2+=""+a[i][j]+",";}
       s2+=""+a[i][j];
       String s3="insert into "+tablei+" values("+s2+");";
       statement.executeUpdate(s3);
      }
    }
    catch(SQLException e)
    {
      // if the error message is "out of memory",
      // it probably means no database file is found
      System.err.println(e.getMessage());
    }
    finally
    {
      try
      {
        if(connection != null)
          connection.close();
      }
      catch(SQLException e)
      {
        // connection close failed.
        System.err.println(e);
      }
    }
}
public static void plot(String databasename, String tablei) throws IOException,ClassNotFoundException
{
```

```
double a[][]=read_double(databasename,tablei);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot(String databasename, String tablei,String column[]) throws IOException,ClassNotFoundException
{
double a[][]=read_double(databasename,tablei,column);
Plot pp=new Plot(a);
pp.plot();
}
}
```

Program 16.2-5 SQLitetest.java, a test program for SQLite class given above

```
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class SQLitetest
{
public static void main(String arg[]) throws IOException,ClassNotFoundException
{
double a[][]=Text.T(Text.readDouble("a.txt"));
double b[][]=Text.T(Text.readDouble("b.txt"));
String column[]={"column1","column2"};
Text.print(Text.T(a),column,"a");
Text.print(Text.T(b),column,"b");
int n=a.length;
int m=a[0].length;
SQLite.write_double("DB1","tableX",column,a);
SQLite.add_double("DB1","tableX",column,b);
double c[][]=SQLite.read_double("DB1","tableX");
Text.print(Text.T(c),column,"c");
SQLite.plot("DB1","tableX");
}
}
```





16.3 SQLite DATABASE WITH WRAPPER CONNECTION

SQLLite wrapper connections are programs written directly in Java native language (C++), therefore execute faster than java JDBC package connected programs. When management of big databases are required, it is better(faster) to use wrapper type programs as well. A wrapper usually has its own command structure, therefore JDBC and wrapper programs should be investigated seperately. One of the best wrapper program could be found at **https://code.google.com/p/sqlite4java/** adress. Latest version (for Windows environment) when this text was written was **sqlite4java-282.** When the jar package openedjar file sqlite4java.jar will be found. This file should be copied into JAVA_HOME/jre/lib/ext directory to program SQLite in java environment. Addition to this two dll files, **sqlite4java-win32-x64.dll** and **sqlite4java-win32-x86.dll** found in this directory. This files should be copied into **Windows/System** or **Windows/System32** directories. Basic classes of the package and the methods of SQLiteConnection class is given below:

Package com.almworks.sqlite4java

Interface Summary	
SQLiteConstants	This interface lists SQLite constants that may be used with sqlite4java.

Class Summary	
SQLite	SQLite class has several utility methods that are applicable to the whole instance of SQLite library within the current process.
SQLiteBackup	SQLiteBackup wraps an instance of SQLite database backup, represented as **`sqlite3_backup*`** in SQLite C API.
SQLiteBlob	SQLiteBlob encapsulates **`sqlite3_blob*`** handle, which represents an open BLOB (binary large object), stored in a single cell of a table.
SQLiteConnection	SQLiteConnection is a single connection to sqlite database.
SQLiteJob<T>	SQLiteJob is a unit of work accepted by `SQLiteQueue`.
SQLiteLongArray	SQLiteLongArray wraps a virtual table handle, created with`SQLiteConnection.createArray(java.lang.String, boolean)`.
SQLiteProfiler	SQLiteProfiler measures and accumulates statistics for various SQLite methods.
SQLiteQueue	SQLiteQueue is a basic implementation of job queue for an SQLite connection.
SQLiteStatement	SQLiteStatement wraps an instance of compiled SQL statement, represented as **`sqlite3_stmt*`** handle in SQLite C Interface.
SQLParts	SQLParts is a means to avoid excessive garbage production during String concatenation when SQL is constructed.

Exception Summary	
SQLiteBusyException	SQLiteBusyException is a special exception that is thrown whenever SQLite returns SQLITE_BUSY or SQLITE_IOERR_BLOCKED error code.

SQLiteException	SQLiteException is thrown whenever SQLite cannot execute an operation and returns an error code.
SQLiteInterruptedException	SQLiteInterruptedException is a special exception that is thrown whenever SQLite returns SQLITE_INTERRUPT following a call to `SQLiteConnection.interrupt()`.

Field Summary	
`static java.lang.String`	**`DEFAULT_DB_NAME`**

Constructor Summary
`SQLiteConnection``()` Creates a connection to an in-memory temporary database.
`SQLiteConnection``(java.io.File dbfile)` Creates a connection to the database located in the specified file.

Method Summary	
`SQLiteBlob`	**`blob`**`(java.lang.String table, java.lang.String column, long rowid, boolean writeAccess)` Convenience method for calling blob() on the currently selected database.
`SQLiteBlob`	**`blob`**`(java.lang.String dbname, java.lang.String table, java.lang.String column, long rowid, boolean writeAccess)` Opens a BLOB for reading or writing.
`SQLiteLongArray`	**`createArray`**`()` Creates a virtual table within the current session, to represent an array of long values (functionality provided by test_intarray module from SQLite sources).
`SQLiteLongArray`	**`createArray`**`(java.lang.String name, boolean cached)` Creates a virtual table within the current session, to represent an array of long values (functionality provided by test_intarray module from SQLite sources).
`java.lang.String`	**`debug`**`(java.lang.String sql)` Runs SQL and returns formatted result.
`void`	**`dispose`**`()` Closes this connection and disposes all related resources.
`SQLiteConnection`	**`exec`**`(java.lang.String sql)` Executes SQL.
`protected void`	**`finalize`**`()` The finalize() method is used to warn about a non-closed connection being forgotten.
`boolean`	**`getAutoCommit`**`()` Checks if the database is in the auto-commit mode.
`int`	**`getChanges`**`()` This method returns the number of database rows that were changed or inserted or deleted by the most recently completed SQL statement in this connection.
`java.io.File`	**`getDatabaseFile`**`()` Returns the database file.
`int`	**`getErrorCode`**`()` This method returns the error code of the most recently failed operation.
`java.lang.String`	**`getErrorMessage`**`()` This method returns the English error message that describes the error returned by `getErrorCode()`.
`long`	**`getLastInsertId`**`()` Returns the ROWID of the row, last inserted in this connection (regardless of which statement was used).
`int`	**`getOpenFlags`**`()`

	Returns the flags that were used to open this connection.
int	**getTotalChanges**() This method returns the total number of database rows that were changed or inserted or deleted since this connection was opened.
SQLiteBackup	**initializeBackup**(java.io.File destinationDbFile) Initializes backup of the database from this connection to the specified file.
SQLiteBackup	**initializeBackup**(java.lang.String sourceDbName, java.io.File destinationDbFile, int flags) Initializes backup of the database with the given name from the current connection to the specified file.
void	**interrupt**() This method can be called to interrupt a currently long-running SQL statement, causing it to fail with an exception.
boolean	**isDisposed**() Checks if the connection has been disposed.
boolean	**isMemoryDatabase**() Checks whether this connection is to an in-memory database.
boolean	**isOpen**() Tells whether connection is open.
void	**loadExtension**(java.io.File extensionFile) Loads an SQLite extension library using default extension entry point.
void	**loadExtension**(java.io.File extensionFile, java.lang.String entryPoint) Loads an SQLite extension library.
SQLiteConnection	**open**() Opens the connection, creating database if needed.
SQLiteConnection	**open**(boolean allowCreate) Opens the connection, optionally creating the database.
SQLiteConnection	**openReadonly**() Opens the connection is read-only mode.
SQLiteConnection	**openV2**(int flags) Opens the connection with the specified flags for the sqlite3_open_v2 method.
SQLiteStatement	**prepare**(SQLParts sql) Convenience method that prepares a cached statement for the given SQL.
SQLiteStatement	**prepare**(SQLParts sql, boolean cached) Prepares an SQL statement.
SQLiteStatement	**prepare**(java.lang.String sql) Convenience method that prepares a cached statement for the given SQL.
SQLiteStatement	**prepare**(java.lang.String sql, boolean cached) Convenience method that prepares a statement for the given String-based SQL.
SQLiteProfiler	**profile**() Starts SQL profiling and returns the profiler class.
SQLiteConnection	**setBusyTimeout**(long millis) Sets "busy timeout" for this connection.
void	**setExtensionLoadingEnabled**(boolean enabled) Enables or disables SQLite extension loading for this connection.
void	**setStepsPerCallback**(int stepsPerCallback) Sets the frequency of database callbacks during long-running SQL statements.
SQLiteProfiler	**stopProfiling**() Stops the profiling and returns the profiler instance with data.

java.lang.String	**toString**()

Methods inherited from class java.lang.Object
clone, equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Summary	
static SQLiteStatement	**DISPOSED** Public instance of initially disposed, dummy statement.

Method Summary	
SQLiteStatement	**bind**(int index, byte[] value) Binds SQL parameter to a BLOB value, represented by a byte array.
SQLiteStatement	**bind**(int index, byte[] value, int offset, int length) Binds SQL parameter to a BLOB value, represented by a range within byte array.
SQLiteStatement	**bind**(int index, double value) Binds SQL parameter to a value of type double.
SQLiteStatement	**bind**(int index, int value) Binds SQL parameter to a value of type int.
SQLiteStatement	**bind**(int index, long value) Binds SQL parameter to a value of type long.
SQLiteStatement	**bind**(int index, java.lang.String value) Binds SQL parameter to a value of type String.
SQLiteStatement	**bindNull**(int index) Binds SQL parameter to a NULL value.
java.io.OutputStream	**bindStream**(int index) Binds SQL parameter to a BLOB value, represented by a stream.
java.io.OutputStream	**bindStream**(int index, int bufferSize) Binds SQL parameter to a BLOB value, represented by a stream.
SQLiteStatement	**bindZeroBlob**(int index, int length) Binds SQL parameter to a BLOB value, consiting of a given number of zero bytes.
void	**cancel**() Cancels the currently running statement.
SQLiteStatement	**clearBindings**() Clears parameter bindings, if there are any.
byte[]	**columnBlob**(int column) Gets a column value after step has returned a row of the result set.
int	**columnCount**() Gets the number of columns in the result set.
double	**columnDouble**(int column) Gets a column value after step has returned a row of the result set.
int	**columnInt**(int column) Gets a column value after step has returned a row of the result set.
long	**columnLong**(int column) Gets a column value after step has returned a row of the result set.
boolean	**columnNull**(int column) Checks if the value returned in the given column is null.

java.io.InputStream	**columnStream**(int column) Gets an InputStream for reading a BLOB column value after step has returned a row of the result set.
java.lang.String	**columnString**(int column) Gets a column value after step has returned a row of the result set.
int	**columnType**(int column) Gets a type of a column after step() has returned a row.
java.lang.Object	**columnValue**(int column) Gets a column value after step has returned a row of the result set.
void	**dispose**() Disposes this statement and frees allocated resources.
int	**getBindParameterCount**() Returns the number of parameters that can be bound.
int	**getBindParameterIndex**(java.lang.String name) Returns the index of a bind parameter with a given name, as defined in the SQL.
java.lang.String	**getBindParameterName**(int index) Returns the name of a given bind parameter, as defined in the SQL.
java.lang.String	**getColumnDatabaseName**(int column) Gets a name of the column's table's database in the result set.
java.lang.String	**getColumnName**(int column) Gets a name of the column in the result set.
java.lang.String	**getColumnOriginName**(int column) Gets the original name of the column that is behind the given column in the result set.
java.lang.String	**getColumnTableName**(int column) Gets a name of the column's table in the result set.
SQLParts	**getSqlParts**() Returns the immutable SQLParts object that was used to create this instance.
boolean	**hasBindings**() Checks if some parameters were bound
boolean	**hasRow**() Checks whether there's data to be read with columnXYZ methods.
boolean	**hasStepped**() Checks if the statement has been evaluated
boolean	**isDisposed**()
boolean	**isReadOnly**() Check if the underlying statement is a SELECT.
int	**loadInts**(int column, int[] buffer, int offset, int length) Loads int values returned from a query into a buffer.
int	**loadLongs**(int column, long[] buffer, int offset, int length) Loads long values returned from a query into a buffer.
SQLiteStatement	**reset**() Convenience method that resets the statement and clears bindings.
SQLiteStatement	**reset**(boolean clearBindings) Resets the statement if it has been stepped, allowing SQL to be run again.
boolean	**step**() Evaluates SQL statement until either there's data to be read, an error occurs, or the statement completes.
SQLiteStatement	**stepThrough**()

	Convenience method that ignores the available data and steps through the SQL statement until evaluation is completed.
`java.lang.String`	**`toString`**`()`

An example program written in this wrapper version is given below:

Program 16.3-1 COFFEE1.java

```
import com.almworks.sqlite4java.*;
import java.io.*;

public class COFFEE1
{
public static void main(String arg[]) throws SQLiteException
{
SQLiteConnection db = new SQLiteConnection(new File("temp"));
db.open(true);
db.exec("drop table if exists COFFEE1");
db.exec("create table COFFEE1(NAME String, CODE int, PRICE double, AMOUNT int,TOTAL double)");
db.exec("insert into COFFEE1 values('espresso',150,9.99,110,0);");
db.exec("insert into COFFEE1 values('Venezuella',173,12.99,57,0);");
db.exec("insert into COFFEE1 values('Brasilian',123,11.56,39,0);");
db.exec("insert into COFFEE1 values('Colombian',111,15.99,20,0);");
db.exec("update COFFEE1 set TOTAL=PRICE*AMOUNT;");
SQLiteStatement statement = db.prepare("select * from COFFEE1");
String NAME;
int CODE;
double PRICE;
int AMOUNT;
double TOTAL;
try {
     while (statement.step()) {
       NAME=statement.columnString(0);
       CODE=statement.columnInt(1);
       PRICE=statement.columnDouble(2);
       AMOUNT=statement.columnInt(3);
       TOTAL=statement.columnDouble(4);
       System.out.print("COFFEE NAME = " + NAME);
       System.out.print(" CODE = " + CODE);
       System.out.print(" PRICE = " + PRICE);
       System.out.print(" AMOUNT = " + AMOUNT);
       System.out.println(" TOTAL = " + TOTAL);
     }
   } finally {
     statement.dispose();
   }
}}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" COFFEE1
Mar 18, 2014 8:47:43 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: instantiated [temp]
Mar 18, 2014 8:47:43 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite4java-win32-x86 from C:\java\jre\lib\ext\sqlite4java-win32-x86.dll
Mar 18, 2014 8:47:43 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite 3.7.4, wrapper 0.2
```

```
Mar 18, 2014 8:47:43 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: opened
COFFEE NAME = espresso CODE = 150 PRICE = 9.99 AMOUNT = 110 TOTAL = 1098.9
COFFEE NAME = Venezuella CODE = 173 PRICE = 12.99 AMOUNT = 57 TOTAL = 740.4300000000001
COFFEE NAME = Brasilian CODE = 123 PRICE = 11.56 AMOUNT = 39 TOTAL = 450.84000000000003
COFFEE NAME = Colombian CODE = 111 PRICE = 15.99 AMOUNT = 20 TOTAL = 319.8

> Terminated with exit code 0.
```

A second example, in this program x and sin(x) is written into the database table sinx and values are read out of the table.

```
import com.almworks.sqlite4java.*;
import java.io.*;

public class sinx1{
public static void main(String args[]) throws SQLiteException
{
SQLiteConnection db = new SQLiteConnection(new File("temp"));
db.open(true);
db.exec("drop table if exists sinx");
String s1= "create table sinx (x double, y double);";
db.exec(s1);
double x[]={0.1, 0.2, 0.3, 0.4, 0.5 };
for(int i=0;i<x.length;i++)
{   s1="insert into sinx values("+x[i]+","+Math.sin(x[0])+");";
System.out.println(s1);
                db.exec(s1);
}
SQLiteStatement statement = db.prepare("select * from sinx");
try{
     while(statement.step())
     {
       // read the result set
       System.out.println("x = " + statement.columnDouble(0)+" y = " + statement.columnDouble(1));
     }
   }  finally {statement.dispose();}
}
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" sinx1
May 06, 2014 8:48:46 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: instantiated [temp]
May 06, 2014 8:48:46 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite4java-win32-x86 from C:\java\jre\lib\ext\sqlite4java-win32-x86.dll
May 06, 2014 8:48:46 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite 3.7.4, wrapper 0.2
May 06, 2014 8:48:46 AM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: opened
insert into sinx values(0.1,0.09983341664682815);
insert into sinx values(0.2,0.09983341664682815);
insert into sinx values(0.3,0.09983341664682815);
insert into sinx values(0.4,0.09983341664682815);
insert into sinx values(0.5,0.09983341664682815);
x = 0.1 y = 0.09983341664682815
x = 0.2 y = 0.09983341664682815
x = 0.3 y = 0.09983341664682815
x = 0.4 y = 0.09983341664682815
x = 0.5 y = 0.09983341664682815

> Terminated with exit code 0.
```

16.4 SQLite DATABASE INFORMATION

SQLite keywords:

The SQL standard specifies a huge number of keywords which may not be used as the names of tables, indices, columns, databases, user-defined functions, collations, virtual table modules, or any other named object. The list of keywords is so long that few people can remember them all. For most SQL code, your safest bet is to never use any English language word as the name of a user-defined object.

If you want to use a keyword as a name, you need to quote it. There are three ways of quoting keywords in SQLite:

'keyword' A keyword in single quotes is a string literal.

"keyword" A keyword in double-quotes is an identifier.

[keyword] A keyword enclosed in square brackets is an identifier. This is not standard SQL. This quoting mechanism is used by MS Access and SQL Server and is included in SQLite for compatibility.

`keyword` A keyword enclosed in grave accents (ASCII code 96) is an identifier. This is not standard SQL. This quoting mechanism is used by MySQL and is included in SQLite for compatibility.

For resilience when confronted with historical SQL statements, SQLite will sometimes bend the quoting rules above:

- If a keyword in single quotes (ex: **'key'** or **'glob'**) is used in a context where an identifier is allowed but where a string literal is not allowed, then the token is understood to be an identifier instead of a string literal.
- If a keyword in double quotes (ex: **"key"** or **"glob"**) is used in a context where it cannot be resolved to an identifier but where a string literal is allowed, then the token is understood to be a string literal instead of an identifier.

Programmers are cautioned not to use the two exceptions described in the previous bullets. We emphasize that they exist only so that old and ill-formed SQL statements will run correctly. Future versions of SQLite might change to raise errors instead of accepting the malformed statements covered by the exceptions above.

SQLite adds new keywords from time to time when it takes on new features. So to prevent your code from being broken by future enhancements, you should normally quote any identifier that is an English language word, even if you do not have to.

The list below shows all possible keywords used by any build of SQLite regardless of compile-time options. Most reasonable configurations use most or all of these keywords, but some keywords may be omitted when SQL language features are disabled. Regardless of the compile-time configuration, any identifier that is not on the following 121 element list is not a keyword to the SQL parser in SQLite:

ABORT	CREATE	FROM	NATURAL	ROW
ACTION	CROSS	FULL	NO	SAVEPOINT

ADD	CURRENT_DATE	GLOB	NOT	SELECT
AFTER	CURRENT_TIME	GROUP	NOTNULL	SET
ALL	CURRENT_TIMESTAMP	HAVING	NULL	TABLE
ALTER	DATABASE	IF	OF	TEMP
ANALYZE	DEFAULT	IGNORE	OFFSET	TEMPORARY
AND	DEFERRABLE	IMMEDIATE	ON	THEN
AS	DEFERRED	IN	OR	TO
ASC	DELETE	INDEX	ORDER	TRANSACTION
ATTACH	DESC	INDEXED	OUTER	TRIGGER
AUTOINCREMENT	DETACH	INITIALLY	PLAN	UNION
BEFORE	DISTINCT	INNER	PRAGMA	UNIQUE
BEGIN	DROP	INSERT	PRIMARY	UPDATE
BETWEEN	EACH	INSTEAD	QUERY	USING
BY	ELSE	INTERSECT	RAISE	VACUUM
CASCADE	END	INTO	REFERENCES	VALUES
CASE	ESCAPE	IS	REGEXP	VIEW
CAST	EXCEPT	ISNULL	REINDEX	VIRTUAL
CHECK	EXCLUSIVE	JOIN	RELEASE	WHEN
COLLATE	EXISTS	KEY	RENAME	WHERE
COLUMN	EXPLAIN	LEFT	REPLACE	
COMMIT	FAIL	LIKE	RESTRICT	
CONFLICT	FOR	LIMIT	RIGHT	
CONSTRAINT	FOREIGN	MATCH	ROLLBACK	

Core functions :

The core functions shown below are available by default. Date & Time functions and aggregate functions are documented separately.

abs(*X*)	The abs(X) function returns the absolute value of the numeric argument X. Abs(X) returns NULL if X is NULL. Abs(X) return 0.0 if X is a string or blob that cannot be converted to a numeric value. If X is the integer -9223372036854775807 then abs(X) throws an integer overflow error since there is no equivalent positive 64-bit two complement value.
changes()	The changes() function returns the number of database rows that were changed or inserted or deleted by the most recently completed INSERT, DELETE, or UPDATE statement, exclusive of statements in lower-level triggers. The changes() SQL function is a wrapper around the sqlite3_changes() C/C++ function and hence follows the same rules for counting changes.
coalesce(*X*,*Y*,...)	The coalesce() function returns a copy of its first non-NULL argument, or NULL if all arguments are NULL. Coalesce() must be at least 2 arguments.
glob(*X*,*Y*)	The glob(X,Y) function is equivalent to the expression "**Y GLOB X**". Note that the X and Y arguments are reversed in the glob() function relative to the infix GLOBoperator. If the sqlite3_create_function() interface is used to override the glob(X,Y) function with an alternative implementation then the GLOB operator will invoke the alternative implementation.

ifnull(*X*,*Y*)	The ifnull() function returns a copy of its first non-NULL argument, or NULL if both arguments are NULL. Ifnull() must have exactly 2 arguments. The ifnull() function is equivalent to coalesce() with two arguments.
hex(*X*)	The hex() function interprets its argument as a BLOB and returns a string which is the upper-case hexadecimal rendering of the content of that blob.
last_insert_rowid()	The last_insert_rowid() function returns the ROWID of the last row insert from the database connection which invoked the function. The last_insert_rowid() SQL function is a wrapper around the sqlite3_last_insert_rowid() C/C++ interface function.
length(*X*)	The length(X) function returns the length of X in characters if X is a string, or in bytes if X is a blob. If X is NULL then length(X) is NULL. If X is numeric then length(X) returns the length of a string representation of X.
like(*X*,*Y*) **like(*X*,*Y*,*Z*)**	The like() function is used to implement the "**Y LIKE X [ESCAPE Z]**" expression. If the optional ESCAPE clause is present, then the like() function is invoked with three arguments. Otherwise, it is invoked with two arguments only. Note that the X and Y parameters are reversed in the like() function relative to the infix LIKEoperator. The sqlite3_create_function() interface can be used to override the like() function and thereby change the operation of the LIKE operator. When overriding the like() function, it may be important to override both the two and three argument versions of the like() function. Otherwise, different code may be called to implement the LIKE operator depending on whether or not an ESCAPE clause was specified.
load_extension(*X*) **load_extension(*X*,*Y*)**	The load_extension(X,Y) function loads SQLite extensions out of the shared library file named X using the entry point Y. The result of load_extension() is always a NULL. If Y is omitted then the default entry point of**sqlite3_extension_init** is used. The load_extension() function raises an exception if the extension fails to load or initialize correctly. The load_extension() function will fail if the extension attempts to modify or delete an SQL function or collating sequence. The extension can add new functions or collating sequences, but cannot modify or delete existing functions or collating sequences because those functions and/or collating sequences might be used elsewhere in the currently running SQL statement. To load an extension that changes or deletes functions or collating sequences, use thesqlite3_load_extension() C-language API.
lower(*X*)	The lower(X) function returns a copy of string X with all ASCII characters converted to lower case. The default built-in lower() function works for ASCII characters only. To do case conversions on non-ASCII characters, load the ICU extension.
ltrim(*X*) **ltrim(*X*,*Y*)**	The ltrim(X,Y) function returns a string formed by removing any and all characters that appear in Y from the left side of X. If the Y argument is omitted, ltrim(X) removes spaces from the left side of X.
max(*X*,*Y*,...)	The multi-argument max() function returns the argument with the maximum value, or return NULL if any argument is NULL. The multi-argument max() function searches its arguments from left to right for an argument that defines a collating function and uses that collating function for all string comparisons. If none of the arguments to max() define a collating function, then the BINARY collating

	function is used. Note that **max()** is a simple function when it has 2 or more arguments but operates as an aggregate function if given only a single argument.
min(*X*,*Y*,...)	The multi-argument min() function returns the argument with the minimum value. The multi-argument min() function searches its arguments from left to right for an argument that defines a collating function and uses that collating function for all string comparisons. If none of the arguments to min() define a collating function, then the BINARY collating function is used. Note that **min()** is a simple function when it has 2 or more arguments but operates as an aggregate function if given only a single argument.
nullif(*X*,*Y*)	The nullif(X,Y) function returns its first argument if the arguments are different and NULL if the arguments are the same. The nullif(X,Y) function searches its arguments from left to right for an argument that defines a collating function and uses that collating function for all string comparisons. If neither argument to nullif() defines a collating function then the BINARY is used.
quote(*X*)	The quote(X) function returns a string which is the value of its argument suitable for inclusion into another SQL statement. Strings are surrounded by single-quotes with escapes on interior quotes as needed. BLOBs are encoded as hexadecimal literals.
random()	The random() function returns a pseudo-random integer between -9223372036854775808 and +9223372036854775807.
randomblob(*N*)	The randomblob(N) function return an N-byte blob containing pseudo-random bytes. If N is less than 1 then a 1-byte random blob is returned. Hint: applications can generate globally unique identifiers using this function together with hex() and/or lower() like this: hex(randomblob(16)) lower(hex(randomblob(16)))
replace(*X*,*Y*,*Z*)	The replace(X,Y,Z) function returns a string formed by substituting string Z for every occurrence of string Y in string X. The BINARY collating sequence is used for comparisons. If Y is an empty string then return X unchanged. If Z is not initially a string, it is cast to a UTF-8 string prior to processing.
round(*X*) **round(*X*,*Y*)**	The round(X,Y) function returns a string representation of the floating-point value X rounded to Y digits to the right of the decimal point. If the Y argument is omitted, the X value is truncated to an integer.
rtrim(*X*) **rtrim(*X*,*Y*)**	The rtrim(X,Y) function returns a string formed by removing any and all characters that appear in Y from the right side of X. If the Y argument is omitted, rtrim(X) removes spaces from the right side of X.
soundex(*X*)	The soundex(X) function returns a string that is the soundex encoding of the string X. The string "?000" is returned if the argument is NULL or contains no ASCII alphabetic characters. This function is omitted from SQLite by default. It is only available if the SQLITE_SOUNDEX compile-time option is used when SQLite is built.

sqlite_compileoption_get(*N*)	The sqlite_compileoption_get() SQL function is a wrapper around thesqlite3_compileoption_get() C/C++ function. This routine returns the N-th compile-time option used to build SQLite or NULL if N is out of range. See also the compile_options pragma.
sqlite_compileoption_used(*X*)	The sqlite_compileoption_used() SQL function is a wrapper around thesqlite3_compileoption_used() C/C++ function. When the argument X to sqlite_compileoption_used(X) is a string which is the name of a compile-time option, this routine returns true (1) or false (0) depending on whether or not that option was used during the build.
sqlite_source_id()	The sqlite_source_id() function returns a string that identifies the specific version of the source code that was used to build the SQLite library. The string returned by sqlite_source_id() begins with the date and time that the source code was checked in and is follows by an SHA1 hash that uniquely identifies the source tree. This function is an SQL wrapper around the sqlite3_sourceid() C interface.
sqlite_version()	The sqlite_version() function returns the version string for the SQLite library that is running. This function is an SQL wrapper around the sqlite3_libversion() C-interface.
substr(*X*,*Y*,*Z*) **substr(*X*,*Y*)**	The substr(X,Y,Z) function returns a substring of input string X that begins with the Y-th character and which is Z characters long. If Z is omitted then substr(X,Y) returns all characters through the end of the string X beginning with the Y-th. The left-most character of X is number 1. If Y is negative then the first character of the substring is found by counting from the right rather than the left. If Z is negative then the abs(Z) characters preceding the Y-th character are returned. If X is a string then characters indices refer to actual UTF-8 characters. If X is a BLOB then the indices refer to bytes.
total_changes()	The total_changes() function returns the number of row changes caused by INSERT, UPDATE or DELETE statements since the current database connection was opened. This function is a wrapper around the sqlite3_total_changes() C/C++ interface.
trim(*X*) **trim(*X*,*Y*)**	The trim(X,Y) function returns a string formed by removing any and all characters that appear in Y from both ends of X. If the Y argument is omitted, trim(X) removes spaces from both ends of X.
typeof(*X*)	The typeof(X) function returns a string that indicates the datatype of the expression X: "null", "integer", "real", "text", or "blob".
upper(*X*)	The upper(X) function returns a copy of input string X in which all lower-case ASCII characters are converted to their upper-case equivalent.
zeroblob(*N*)	The zeroblob(N) function returns a BLOB consisting of N bytes of 0x00. SQLite manages these zeroblobs very efficiently. Zeroblobs can be used to reserve space for a BLOB that is later written using incremental BLOB I/O. This SQL function is implemented using the sqlite3_result_zeroblob() routine from the C/C++ interface.

Aggregate functions :

The aggregate functions shown below are available by default. In any aggregate function that takes a single argument, that argument can be preceded by the keyword DISTINCT. In such cases, duplicate elements are filtered before being passed into the aggregate function. For example, the function "count(distinct X)" will return the number of distinct values of column X instead of the total number of non-null values in column X.

avg(*X*)	The avg() function returns the average value of all non-NULL *X* within a group. String and BLOB values that do not look like numbers are interpreted as 0. The result of avg() is always a floating point value as long as at there is at least one non-NULL input even if all inputs are integers. The result of avg() is NULL if and only if there are no non-NULL inputs.
count(*X*) **count(*)**	The count(X) function returns a count of the number of times that *X* is not NULL in a group. The count(*) function (with no arguments) returns the total number of rows in the group.
group_concat(*X*) **group_concat(*X*,*Y*)**	The group_concat() function returns a string which is the concatenation of all non-NULL values of *X*. If parameter *Y* is present then it is used as the separator between instances of*X*. A comma (",") is used as the separator if *Y* is omitted. The order of the concatenated elements is arbitrary.
max(*X*)	The max() aggregate function returns the maximum value of all values in the group. The maximum value is the value that would be returned last in an ORDER BY on the same column. Aggregate max() returns NULL if and only if there are no non-NULL values in the group.
min(*X*)	The min() aggregate function returns the minimum non-NULL value of all values in the group. The minimum value is the first non-NULL value that would appear in an ORDER BY of the column. Aggregate min() returns NULL if and only if there are no non-NULL values in the group.
sum(*X*) **total(*X*)**	The sum() and total() aggregate functions return sum of all non-NULL values in the group. If there are no non-NULL input rows then sum() returns NULL but total() returns 0.0. NULL is not normally a helpful result for the sum of no rows but the SQL standard requires it and most other SQL database engines implement sum() that way so SQLite does it in the same way in order to be compatible. The non-standard total() function is provided as a convenient way to work around this design problem in the SQL language. The result of total() is always a floating point value. The result of sum() is an integer value if all non-NULL inputs are integers. If any input to sum() is neither an integer or a NULL then sum() returns a floating point value which might be an approximation to the true sum. Sum() will throw an "integer overflow" exception if all inputs are integers or NULL and an integer overflow occurs at any point during the computation. Total() never throws an integer overflow.

Date And Time Functions:

SQLite supports five date and time functions as follows:

1. **date(***timestring, modifier, modifier, ...***)**
2. **time(***timestring, modifier, modifier, ...***)**
3. **datetime(***timestring, modifier, modifier, ...***)**
4. **julianday(***timestring, modifier, modifier, ...***)**
5. **strftime(***format, timestring, modifier, modifier, ...***)**

All five date and time functions take a time string as an argument. The time string is followed by zero or more modifiers. The strftime() function also takes a format string as its first argument.

The date and time functions use a subset of IS0-8601 date and time formats. The date() function returns the date in this format: YYYY-MM-DD. The time() function returns the time as HH:MM:SS. The datetime() function returns "YYYY-MM-DD HH:MM:SS". The julianday() function returns the Julian day - the number of days since noon in Greenwich on November 24, 4714 B.C. (Proleptic Gregorian calendar). The strftime() routine returns the date formatted according to the format string specified as the first argument. The format string supports the most common substitutions found in the strftime() function from the standard C library plus two new substitutions, %f and %J. The following is a complete list of valid strftime() substitutions:

%d	day of month: 00
%f	fractional seconds: SS.SSS
%H	hour: 00-24
%j	day of year: 001-366
%J	Julian day number
%m	month: 01-12
%M	minute: 00-59
%s	seconds since 1970-01-01
%S	seconds: 00-59
%w	day of week 0-6 with Sunday==0
%W	week of year: 00-53
%Y	year: 0000-9999
%%	%

Notice that all other date and time functions can be expressed in terms of strftime():

Function	**Equivalent strftime()**
date(...)	strftime('%Y-%m-%d', ...)
time(...)	strftime('%H:%M:%S', ...)
datetime(...)	strftime('%Y-%m-%d %H:%M:%S', ...)
julianday(...)	strftime('%J', ...)

The only reasons for providing functions other than strftime() is for convenience and for efficiency.

Time Strings

A time string can be in any of the following formats:

1. *YYYY-MM-DD*
2. *YYYY-MM-DD HH:MM*
3. *YYYY-MM-DD HH:MM:SS*
4. *YYYY-MM-DD HH:MM:SS.SSS*
5. *YYYY-MM-DD***T***HH:MM*
6. *YYYY-MM-DD***T***HH:MM:SS*
7. *YYYY-MM-DD***T***HH:MM:SS.SSS*
8. *HH:MM*
9. *HH:MM:SS*
10. *HH:MM:SS.SSS*
11. **now**
12. *DDDDDDDDDD*

In formats 5 through 7, the "T" is a literal character separating the date and the time, as required by ISO-8601. Formats 8 through 10 that specify only a time assume a date of 2000-01-01. Format 11, the string 'now', is converted into the current date and time as obtained from the xCurrentTime method of the sqlite3_vfs object in use.Universal Coordinated Time (UTC) is used. Format 12 is the Julian day number expressed as a floating point value.

Modifiers

The time string can be followed by zero or more modifiers that alter date and/or time. Each modifier is a transformation that is applied to the time value to its left. Modifiers are applied from left to right; order is important. The available modifiers are as follows.

1. NNN days
2. NNN hours
3. NNN minutes
4. NNN.NNNN seconds

5. NNN months
6. NNN years
7. start of month
8. start of year
9. start of day
10. weekday N
11. unixepoch
12. localtime
13. utc

The first six modifiers (1 through 6) simply add the specified amount of time to the date and time specified by the preceding timestring and modifiers. Note that "±NNN months" works by rendering the original date into the YYYY-MM-DD format, adding the ±NNN to the MM month value, then normalizing the result. Thus, for example, the data 2001-03-31 modified by '+1 month' initially yields 2001-04-31, but April only has 30 days so the date is normalized to 2001-05-01. A similar effect occurs when the original date is February 29 of a leapyear and the modifier is ±N years where N is not a multiple of four.

The "start of" modifiers (7 through 9) shift the date backwards to the beginning of the current month, year or day.

The "weekday" modifier advances the date forward to the next date where the weekday number is N. Sunday is 0, Monday is 1, and so forth.

The "unixepoch" modifier (11) only works if it immediately follows a timestring in the DDDDDDDDDD format. This modifier causes the DDDDDDDDDD to be interpreted not as a Julian day number as it normally would be, but as Unix Time - the number of seconds since 1970. If the "unixepoch" modifier does not follow a timestring of the form DDDDDDDDDD which expresses the number of seconds since 1970 or if other modifiers separate the "unixepoch" modifier from prior DDDDDDDDDD then the behavior is undefined. Due to precision limitations imposed by the implementations use of 64-bit integers, the "unixepoch" modifier only works for dates between 0000-01-01 00:00:00 and 5352-11-01 10:52:47 (unix times of -62167219200 through 10675199167).

The "localtime" modifier (12) assumes the time string to its left is in Universal Coordinated Time (UTC) and adjusts the time string so that it displays localtime. If "localtime" follows a time that is not UTC, then the behavior is undefined. The "utc" is the opposite of "localtime". "utc" assumes that the string to its left is in the local timezone and adjusts that string to be in UTC. If the prior string is not in localtime, then the result of "utc" is undefined.

Examples

Compute the current date.
SELECT date('now');

Compute the last day of the current month.
SELECT date('now','start of month','+1 month','-1 day');

Compute the date and time given a unix timestamp 1092941466.
SELECT datetime(1092941466, 'unixepoch');

Compute the date and time given a unix timestamp 1092941466, and compensate for your local timezone.
SELECT datetime(1092941466, 'unixepoch', 'localtime');

Compute the current unix timestamp.
SELECT strftime('%s','now');

Compute the number of days since the signing of the US Declaration of Independence.
SELECT julianday('now') - julianday('1776-07-04');

Compute the number of seconds since a particular moment in 2004:
SELECT strftime('%s','now') - strftime('%s','2004-01-01 02:34:56');

Compute the date of the first Tuesday in October for the current year.
SELECT date('now','start of year','+9 months','weekday 2');

Compute the time since the unix epoch in seconds (like strftime('%s','now') except includes fractional part):
SELECT (julianday('now') - 2440587.5)*86400.0;

WEEK 8 CLASS EXERSIZES

EX1) Temperature and specific heat of CO_2 gas is given in file co2_cv.txt . Create database table CO2 in database temp by using **create table CO2 (T float, Cv float, TK float);** command. Read data as java input from file, calculate TK=T+273 by using Update

update CO2 set TK=T+273.15;

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.io.*;
public class createCO2{
public static void main(String args[]) throws ClassNotFoundException
{
Class.forName("org.sqlite.JDBC");
Connection connection = null;
try {
connection = DriverManager.getConnection("jdbc:sqlite:temp.db");
Statement stmt = null;
ResultSet rs1 = null;
stmt = connection.createStatement();
String s1= "create table CO2 (T float, CV float, TK float );";
if (stmt.execute(s1)) {rs1 = stmt.getResultSet();}
} catch (SQLException ex) {
// handle any errors
System.out.println("SQLException: " + ex.getMessage());
System.out.println("SQLState: " + ex.getSQLState());
System.out.println("VendorError: " + ex.getErrorCode());
 }

}
}
```

```
 import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.sql.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

class data
        {
        public double x[];
        public data(double[] xi)
        {
        int n=xi.length;
        x=new double[n];
        for(int i=0;i<n;i++) {x[i]=xi[i];}
        }
        public String toString() {return ""+x[0]+" "+x[1];}
        }

public class insertCO2
{
```

```
    public static double[][] inputdouble(String filename) throws IOException
    {
                //reading double values from one line of data
                BufferedReader fin=new BufferedReader(new FileReader(filename));
    double a[][]=inputdouble(fin);
    return a;
    }

   public static double[][] inputdouble(BufferedReader fin) throws IOException
   {
                ArrayList<data> a=new ArrayList<data>();
                int mmax=0;
                double b[][];
                int n=0,m=0;
                String s1="";
                try{
     while(fin!=null)
     { s1=fin.readLine();
     if(s1==null)break;
       StringTokenizer token=new StringTokenizer(s1);
       m=token.countTokens();
       if(m>mmax) mmax=m;
       double aa[]=new double[m];
       int j=0;
       while(token.hasMoreTokens())
       { Double ax=new Double(token.nextToken());
         aa[j++]=ax.doubleValue();
       }
       data vx=new data(aa);
       a.add(vx);
     n++;
                  }
     } catch(EOFException e_eof)
        {
        //close ffile
        try{fin.close();
    }
    catch(NullPointerException e) {}
    catch(IOException e)
           {
           System.err.println("Error Closing File\n"+e.toString());
           System.exit(1);
           }
        } //End of EOFException
   b=new double[n][mmax];
   int k=0;
   Iterator i=a.iterator();
   while(i.hasNext()) {data vx=(data)i.next();b[k++]=vx.x;}
   return b;
   }
   public static String toString(double a[][])
   {
                String s="";
                for(int j=0;j<a.length;j++)
   {s+="T["+j+"] = "+a[j][0]+" Cv["+j+"] = "+a[j][1]+" 0.0\n";}
   s+="\n";
   return s;
                }

public static void main(String args[]) throws IOException,ClassNotFoundException
{ Class.forName("org.sqlite.JDBC");
  Connection conn = null;
  try {
  String s1="co2_cv.txt";
  File f=new File(s1);
  BufferedReader fin=new BufferedReader(new FileReader(f));
```

```
  double number[][]=inputdouble(fin);
  conn =DriverManager.getConnection("jdbc:sqlite:temp.db");
  Statement stmt = null;
  ResultSet rs = null;
  stmt = conn.createStatement();
  String s2="insert into CO2 values(";
  String s3="";
  for(int i=0;i<number.length;i++)
  {
  s3=s2+number[i][0]+","+number[i][1]+", 0.0);";
  stmt.executeUpdate(s3);
  }
  JOptionPane.showMessageDialog(null,toString(number),"data read into the database CO2",JOptionPane.PLAIN_MESSAGE);
  System.exit(0);
  }
   catch (SQLException ex) {
      System.out.println("SQLException: " + ex.getMessage());
      System.out.println("SQLState: " + ex.getSQLState());
      System.out.println("VendorError: " + ex.getErrorCode());
      }
}
}
```

EX 2) : Create table **sinx** with 2 column x and y. Enter x column values . 0.1 0.2 .03 0.4 0.5 , enter all y values as 0 . Change y values to sin(x) by using update statement. Write out the results.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.io.*;
public class sinx{
public static void main(String args[]) throws ClassNotFoundException
{
Class.forName("org.sqlite.JDBC");
Connection connection = null;
try {
connection = DriverManager.getConnection("jdbc:sqlite:temp.db");
Statement stmt = null;
ResultSet rs1 = null;
stmt = connection.createStatement();
stmt.setQueryTimeout(30);
stmt.executeUpdate("drop table if exists sinx");
String s1= "create table sinx (x double, y double);";
stmt.executeUpdate(s1);
double x[]={0.1, 0.2, 0.3, 0.4, 0.5 };
for(int i=0;i<x.length;i++)
{   s1="insert into sinx values("+x[i]+","+0+");";
System.out.println(s1);
                stmt.executeUpdate(s1);
}
s1="update sinx set y=sin(x);";
stmt.executeUpdate(s1);
ResultSet rs = stmt.executeQuery("select * from sinx");
     while(rs.next())
     {
       // read the result set
       System.out.println("x = " + rs.getString("x")+" y = " + rs.getString("y"));
     }
}catch(SQLException e)
    {
     // if the error message is "out of memory",
     // it probably means no database file is found
```

```
    System.err.println(e.getMessage());
   }
   finally
   {
    try
    {
     if(connection != null)
       connection.close();
    }
    catch(SQLException e)
    {
     // connection close failed.
     System.err.println(e);
    }
}}}
```

EX3) : Create table **student** with data fields **name String, surname String ,grade1 double, grade 2 double, result double** Input the following data

'Ali' 'Çiçek' 24 58
'Veli' 'Durmuş' 49 73
'Hasan' 'Yücel' 72 68
'Mehmet' 'Demir' 63 55

Calculate result as 40% grade1 and 60% grade2 and write down into database result column. Print out the results.

EX4) Temperature and specific heat data of CO_2 gas is entered to CO2 table in exercise 1 . Read the data, from database temp.CO2, calculate the avarage temperatures and avarage CV value and print out the results.

```
import java.sql.*;
import java.io.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class averageH8{
public static void main(String args[]) throws IOException,ClassNotFoundException
{
try {
Connection conn =DriverManager.getConnection("jdbc:sqlite:temp.db");
Statement stmt = null;
ResultSet rs1 = null;
stmt = conn.createStatement();
String query1="select AVG(CV) as Tavg from CO2";;
if (stmt.execute(query1)) {rs1 = stmt.getResultSet();}
while (rs1.next()) {
            double i= rs1.getDouble("Tavg");
System.out.println("average of CV = "+i);
          }
} catch (SQLException ex) {
// handle any errors
System.out.println("SQLException: " + ex.getMessage());
System.out.println("SQLState: " + ex.getSQLState());
System.out.println("VendorError: " + ex.getErrorCode());
 }
}
}
```

```
import java.sql.*;
import java.io.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class average1H8{
public static void main(String args[]) throws IOException,ClassNotFoundException
{
try {
Connection conn =DriverManager.getConnection("jdbc:sqlite:temp.db");
Statement stmt = null;
ResultSet rs1 = null;
stmt = conn.createStatement();
String query1="select CV from CO2";;
if (stmt.execute(query1)) {rs1 = stmt.getResultSet();}
int j=0;
double sum=0;
while (rs1.next()) {
 double number= rs1.getDouble("CV");
 sum+=number;
 j++;
        }
 double average=sum/j;
 System.out.println("average : "+average);
} catch (SQLException ex) {
// handle any errors
System.out.println("SQLException: " + ex.getMessage());
System.out.println("SQLState: " + ex.getSQLState());
System.out.println("VendorError: " + ex.getErrorCode());
 }
}
}
```

17. READING/WRITING DATA FROM THE SPREADSHEET PROGRAMS

17.1 GENERAL DEFINITIONS

Spreadsheet programs such as Microsoft spreadsheet, OpenOffice calc, Lotus, StarOffice Calc are common calculating and data recording environments nowadays. In this chapter, accessing (reading and writing) to spreadshets will be examined. In order to do this a public access java-spreadsheet interphase apache poi will be used. You can download free program apache poi from the internet adres http://poi.apache.org/After downloading, unzip the zip folder and open it. Under the lib directory, you will find jar files

poi-excelant-3.10-FINAL-20140208.jar	poi-ooxml-3.10-FINAL-20140208.jar
poi-ooxml-schemas-3.10-FINAL-20140208.jar	poi-scratchpad-3.10-FINAL-20140208.jar
poi-3.10-FINAL-20140208.jar	poi-examples-3.10-FINAL-20140208.jar
commons-codec-1.5.jar	commons-logging-1.1.jar
junit-4.11.jar	log4j-1.2.13.jar

copy jar files into /java home/jre/lib/ext folder. Now you are ready to access the spreadsheets.

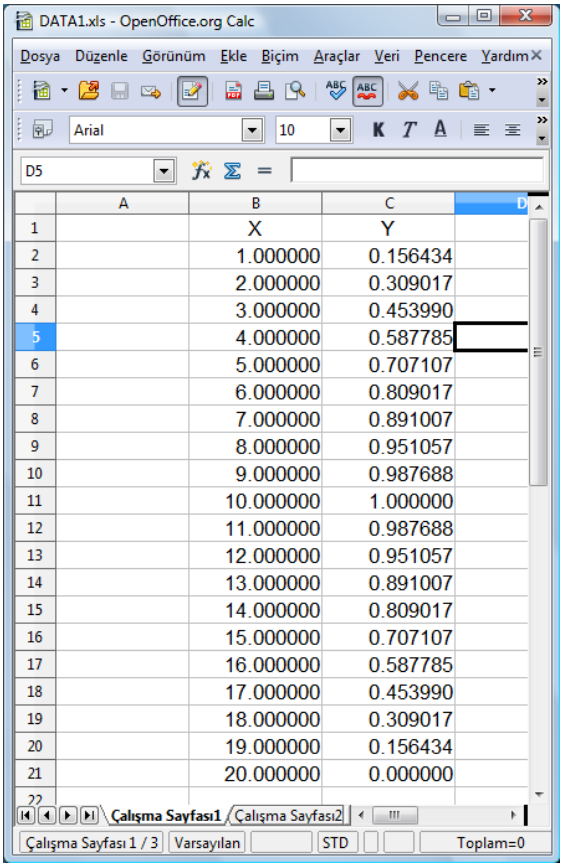
17.2 READING DATA FROM SPREADHEETS

Poi has subsystems to communucate with all Microsoft Office products including excel files. can read an excel spreadsheet from a file stored on the local filesystem or from some input stream. The first step when reading a spreadsheet from a file or input stream is to create a Workbook. In modern excel, there are two different systems, xlsx and xls files. In order to open xlsx files workbook class **XSSFWorkbook,** for xls files **HSSFWorkbook** class is used. The code fragment below illustrates creating a workbook from a file on the local filesystem.

```
import org.apache.poi.xssf.usermodel.*;
import java.io.*;
import java.util.*;
…….
FileInputStream file = new FileInputStream("DATA1.xlsx");
XSSFWorkbook xssfWork = new XSSFWorkbook(file);
```

Or if the file is an xls file

```
import org.apache.poi.hssf.usermodel.*;
import java.io.*;
import java.util.*;
.....
FileInputStream file = new FileInputStream(new File("DATA1.xls"));
HSSFWorkbook workbook = new HSSFWorkbook(file);
```



	A	B	C
1		X	Y
2		1.000000	0.156434
3		2.000000	0.309017
4		3.000000	0.453990
5		4.000000	0.587785
6		5.000000	0.707107
7		6.000000	0.809017
8		7.000000	0.891007
9		8.000000	0.951057
10		9.000000	0.987688
11		10.000000	1.000000
12		11.000000	0.987688
13		12.000000	0.951057
14		13.000000	0.891007
15		14.000000	0.809017
16		15.000000	0.707107
17		16.000000	0.587785
18		17.000000	0.453990
19		18.000000	0.309017
20		19.000000	0.156434
21		20.000000	0.000000

There is also a third way that can handle both xls and xlsx cases through an interphase class **org.apache.poi.ss.** By using this

```
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.*;
import java.io.*;
import java.util.*;
….
FileInputStream file = new FileInputStream("DATA1.xls");
Workbook workbook = new HSSFWorkbook(file) ;
```

```
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Cell;
import java.io.*;
import java.util.*;
.....
FileInputStream file = new FileInputStream(new File("DATA1.xlsx"));
Workbook workbook = new XSSFWorkbook(file);
```

Once you have accessed the **workbook**, you can use this to Access to the individual sheets. These are zero indexed - the first sheet being 0, the second sheet being 1, and so on
Sheet sheet = workbook.getSheetAt(0);

Once you have a sheet, you can then start accessing the cells.
Row r=sheet.getRow(i);
Cell c=r.getCell(j);

Once Cell object created content of the cell can be accessed by detecting cell type

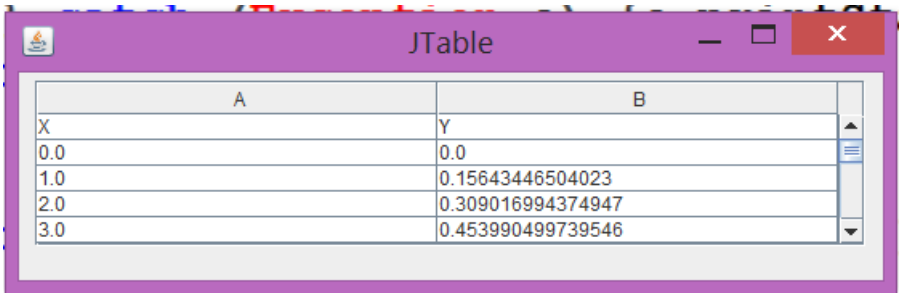
```
String s="";
        if(cell.getCellType()==Cell.CELL_TYPE_BOOLEAN)
        { s=cell.getBooleanCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_NUMERIC)
        { s=cell.getNumericCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_STRING)
        {s=cell.getStringCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_FORMULA)
        {s=cell.getCellFormula() + "\t\t";}
```

You can retrieve the cell's contents as a string by using the convenience method getStringCellValue(), or as a numerical value by using getNumericCellValue(). If Cell formula is needed to be extract directly getCellFormula() is available. In the following example codes below, data is extracted as String from the excell file.

PROGRAM 17.1 apachePOI3 program, an example of reading real data from spreadsheet

```
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI3
{
public static String[][] read_String_from_excel(String filename, int pageno)
{
  String a[][]=new String[10][10];
  try{
  int nx,ny=0,nx0=0,ny0=0,nx1=0,ny1=0;
  Workbook workbook =wb(filename);
  Sheet sheet= workbook.getSheetAt(pageno);
  nx1=sheet.getLastRowNum();
  nx0=sheet.getFirstRowNum();
  nx=nx1-nx0;
  for(int i=0;i<nx;i++)
  {   Row r=sheet.getRow(i+nx0);
      if(i==nx0)
      {ny0= r.getFirstCellNum();
               ny1= r.getLastCellNum();
               ny=ny1-ny0;
       a=new String[nx][ny];
      }
              for(int j=0;j<ny;j++)
      {  Cell c=r.getCell(j+ny0,Row.RETURN_BLANK_AS_NULL);
                a[i][j] =toString(c);
      }
  }
}
  catch(ArrayIndexOutOfBoundsException e2) {System.err.println("wrong array dimension");}
  return a;
}

public static String toString(Cell cell)
{
           String s="";
```

```
            if(cell.getCellType()==Cell.CELL_TYPE_BOOLEAN)
            { s=cell.getBooleanCellValue() + "\t\t";}
            else if(cell.getCellType()==Cell.CELL_TYPE_NUMERIC)
            { s=cell.getNumericCellValue() + "\t\t";}
            else if(cell.getCellType()==Cell.CELL_TYPE_STRING)
            {s=cell.getStringCellValue() + "\t\t";}
            else if(cell.getCellType()==Cell.CELL_TYPE_FORMULA)
            {s=cell.getNumericCellValue() + "\t\t";}
            return s;
}
    public static String word(String s)
    {  //returns extension group of file name
    StringTokenizer token=new StringTokenizer(s,".");
    int m=token.countTokens();
    String b[]=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    String ax=(String)token.nextToken();
    b[j++]=ax;
    }
    return b[m-1];
    }
public static Workbook wb(String filename)
{
   String s2=word(filename);
   Workbook Work=new HSSFWorkbook();
   try {
    FileInputStream inputStr= new FileInputStream(filename);
   if(s2.equals("xls"))
   {Work=new HSSFWorkbook(inputStr);}
   else
   {Work= new XSSFWorkbook(inputStr) ;}
   } catch (Exception e) {e.printStackTrace();}
   return Work;
}

public static void main(String arg[])
{
    String s[][]=read_String_from_excel("DATA1.xls",0);
    JTableP.print(s);
}
}
```

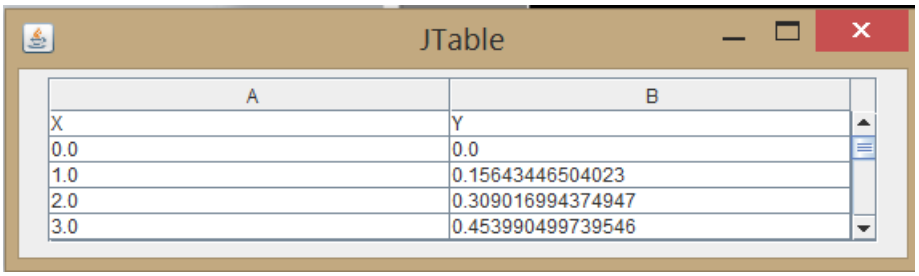


The next program is carrying out the same extraction but, it utilizes Iterator class for detecting the data.

PROGRAM 17.2 apachePOI2 program, an example of reading real data from spreadsheet

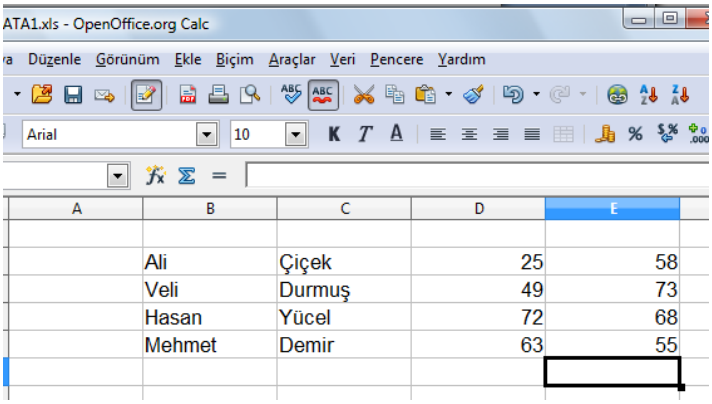
```
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI2
{
public static String[][] read_String_from_excel1(String filename, int pageno)
{
   Workbook Work=wb(filename);
   String s[][]=new String [2000][2000];
   int i=0,j=0,i1=0,j1=0;
   Sheet sheet1 = Work.getSheetAt(pageno);
   Iterator rowItr = sheet1.rowIterator();
   while ( rowItr.hasNext() )
   {   j=0;
                Row row = (Row) rowItr.next();
      Iterator cellItr = row.cellIterator();
      while ( cellItr.hasNext() )
      { s[i][j]=toString(cellItr);
        //in case of an empty line stop iteration
        if(s[i][j].charAt(0)==' ') {i=i1;j=j1;break;}
        //System.out.print("i="+i+"j="+j+"*"+s[i][j]+"* ");
        j++;j1=j;
      }
  i++;i1=i;
  }
  int n=i1;
  int m=j1;
  String s1[][]=new String [n][m];
  for(i=0;i<n;i++)
  {for(j=0;j<m;j++)
    {s1[i][j]=s[i][j];}
  }
  return s1;
}

public static String toString(Iterator cellItr)
{
         String s="";
         Cell cell = (Cell) cellItr.next();
         if(cell.getCellType()==Cell.CELL_TYPE_BOOLEAN)
         { s=cell.getBooleanCellValue() + "\t\t";}
         else if(cell.getCellType()==Cell.CELL_TYPE_NUMERIC)
         { s=cell.getNumericCellValue() + "\t\t";}
         else if(cell.getCellType()==Cell.CELL_TYPE_STRING)
         {s=cell.getStringCellValue() + "\t\t";}
         else if(cell.getCellType()==Cell.CELL_TYPE_FORMULA)
         {s=cell.getNumericCellValue() + "\t\t";}
         return s;
```

```
}
    public static String word(String s)
    {  //returns extension group of file name
    StringTokenizer token=new StringTokenizer(s,".");
    int m=token.countTokens();
    String b[]=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    String ax=(String)token.nextToken();
    b[j++]=ax;
    }
    return b[m-1];
    }
public static Workbook wb(String filename)
{
   String s2=word(filename);
   Workbook Work=new HSSFWorkbook();
   try {
    FileInputStream inputStr= new FileInputStream(filename);
   if(s2.equals("xls"))
   {Work=new HSSFWorkbook(inputStr);}
   else
   {Work= new XSSFWorkbook(inputStr) ;}
   } catch (Exception e) {e.printStackTrace();}
   return Work;
}

public static void main(String arg[])
{
    String s[][]=read_String_from_excel1("DATA1.xls",0);
    JTableP.print(s);
}}
```



A	B
X	Y
0.0	0.0
1.0	0.15643446504023
2.0	0.309016994374947
3.0	0.453990499739546

In the next program exact starting point of the data is defined by the program



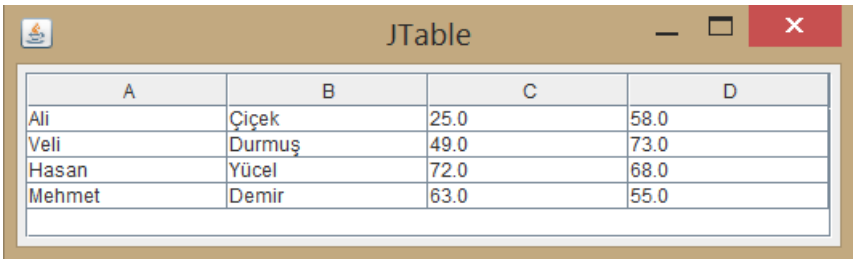
A	B	C	D	E
	Ali	Çiçek	25	58
	Veli	Durmuş	49	73
	Hasan	Yücel	72	68
	Mehmet	Demir	63	55

PROGRAM 17.3 apachePOI4 program, an example of reading real data from spreadsheet. Exact start and end is given as B2 to E5

```
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI4
{
public static String[][] read_String_from_excel(String filename, int pageno,String xy0,String xyn)
{int n0[]=excelcode(xy0);
 int n1[]=excelcode(xyn);
 int nx=n1[0]-n0[0]+1;
 int ny=n1[1]-n0[1]+1;
 return read_String_from_excel(filename,pageno,n0[0],n0[1],nx,ny);
}
public static String[][] read_String_from_excel(String filename, int pageno,int x0,int y0,int nx,int ny)
{
  String a[][]=new String[nx][ny];
  int nx1,ny1;
  //try{
  Workbook workbook =wb(filename);
  Sheet sheet= workbook.getSheetAt(pageno);
  int totalrows=sheet.getLastRowNum();
  if(totalrows<nx) nx1=totalrows;
  else nx1=nx;
  for(int i=0;i<nx1;i++)
  {   Row r=sheet.getRow(x0+i);
      int totalCells = r.getPhysicalNumberOfCells();
      if(totalCells<ny) ny1=totalCells;
      else ny1=ny;
                for(int j=0;j<ny1;j++)
      {  Cell c=r.getCell((y0+j),Row.RETURN_BLANK_AS_NULL);
                  if(c==null) break;
                  else {a[i][j] =toString(c);}
      }
  }
  return a;
}
public static boolean isLetter(char x)
{ boolean x1;
  if((x>='A' && x<='Z') || (x>='a' && x<='z')) x1=true;
  else x1=false;
  return x1;
}

public static boolean isNumber(char x)
{ boolean x1;
  if(x>='0' && x<='9') x1=true;
  else x1=false;
  return x1;
}
```

```
public static int charnumber(char x)
{
  char b1[]={'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};
  char b2[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'};
  for(int i=0;i<26;i++)
  {if(x==b1[i] || x==b2[i]) return i;}
  return 0;
}

public static int[] excelcode(String s)
{  s=s.trim();
   int len = s.length();
   //System.out.println("excelcode = "+s+"len="+len);
 int n[]=new int[2];
 String s1="";
 char a[]=new char[len];
 int n1[]=new int[2];
 for(int i=0;i<len;i++)
 {a[i]=s.charAt(i);
 }
 if(isLetter(a[0]) && isLetter(a[1]))
 {
 n1[1]=(charnumber(a[0])+1)*26+charnumber(a[1]);
 for(int i=2;i<len;i++) s1+=a[i];
 }
 else
 {n1[1]=charnumber(a[0]);
 for(int i=1;i<len;i++) s1+=a[i];
 }
 n1[0]=Integer.parseInt(s1)-1;
 return n1;
}

public static String toString(Cell cell)
{
        String s="";
        if(cell.getCellType()==Cell.CELL_TYPE_BOOLEAN)
        { s=cell.getBooleanCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_NUMERIC)
        { s=cell.getNumericCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_STRING)
        {s=cell.getStringCellValue() + "\t\t";}
        else if(cell.getCellType()==Cell.CELL_TYPE_FORMULA)
        {s=cell.getNumericCellValue() + "\t\t";}
        return s;
}
    public static String word(String s)
    {  //returns extension group of file name
    StringTokenizer token=new StringTokenizer(s,".");
    int m=token.countTokens();
    String b[]=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    String ax=(String)token.nextToken();
    b[j++]=ax;
    }
    return b[m-1];
```

```
    }
public static Workbook wb(String filename)
{
   String s2=word(filename);
   Workbook Work=new HSSFWorkbook();
   try {
    FileInputStream inputStr= new FileInputStream(filename);
   if(s2.equals("xls"))
   {Work=new HSSFWorkbook(inputStr);}
   else
   {Work= new XSSFWorkbook(inputStr) ;}
   } catch (Exception e) {e.printStackTrace();}
   return Work;
}

public static void main(String arg[])
{
    String s[][]=read_String_from_excel("DATA1.xls",1,"B2","D5");
    JTableP.print(s);
}
}
```



A	B	C	D
Ali	Çiçek	25.0	58.0
Veli	Durmuş	49.0	73.0
Hasan	Yücel	72.0	68.0
Mehmet	Demir	63.0	55.0

17.3 WRITING DATA INTO SPREADHEETS

This section describes how to write out simple spreadsheet data without any formatting information, such as fonts or decimal places. Similarly to reading a spreadsheet, the first step is to create a writable workbook using the factory method on the Workbook class.

```
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;...
…….,
```

Workbook workbook = new HSSFWorkbook();

Or

Workbook workbook = new SXSSFWorkbook() ;

And then Sheet and Row and Cell definitions are done similar to reading from excell file

Sheet sheet= workbook.createSheet(0);

…

Row row=sheet.createRow(i);

…

Cell cell=row.createCell(j);

Then cell value is set into the cell:
cell.setCellValue(cell_value);

After writing all the desired cells, the excell file is created and written through a FileOutputStream

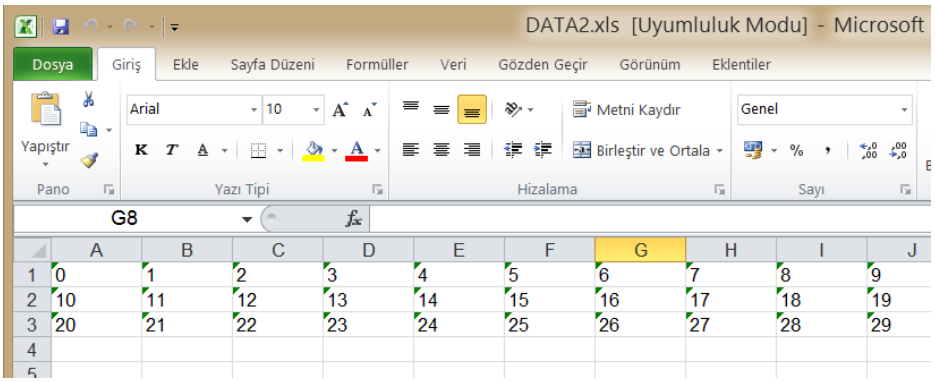
```
try {
    FileOutputStream out=new FileOutputStream(filename);
    workbook.write(out);
    out.close();
   } catch (Exception e) {e.printStackTrace();}
```

PROGRAM 17.4 an example of writing real data into spreadsheet

```
 import org.apache.poi.ss.usermodel.*;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI5
{
public static void write_to_excel(String filename, int pageno,String xy0,Object a[][])
{
int n0[]=excelcode(xy0);
int nx=n0[0];
int ny=n0[1];
Workbook workbook =wb1(filename);
String s1=""+pageno;
 Sheet sheet= workbook.createSheet(s1);
 int n=a.length;
 int m=a[0].length;
 for(int i=0;i<n;i++)
 { Row row=sheet.createRow(nx+i);
   for(int j=0;j<m;j++)
   {Cell cell=row.createCell(ny+j);
    String adress=new CellReference(cell).formatAsString();
    cell.setCellValue((String)a[i][j]);
   }
 }
try {
     FileOutputStream out=new FileOutputStream(filename);
     workbook.write(out);
     out.close();
    } catch (Exception e) {e.printStackTrace();}
}
public static boolean isLetter(char x)
{ boolean x1;
  if((x>='A' && x<='Z') || (x>='a' && x<='z')) x1=true;
  else x1=false;
  return x1;
}
```

```
public static boolean isNumber(char x)
{ boolean x1;
  if(x>='0' && x<='9') x1=true;
  else x1=false;
  return x1;
}

public static int charnumber(char x)
{
  char b1[]={'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};
  char b2[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'};
  for(int i=0;i<26;i++)
  {if(x==b1[i] || x==b2[i]) return i;}
  return 0;
}

public static int[] excelcode(String s)
{  s=s.trim();
   int len = s.length();
   //System.out.println("excelcode = "+s+"len="+len);
 int n[]=new int[2];
 String s1="";
 char a[]=new char[len];
 int n1[]=new int[2];
 for(int i=0;i<len;i++)
 {a[i]=s.charAt(i);
 }
 if(isLetter(a[0]) && isLetter(a[1]))
 {
 n1[1]=(charnumber(a[0])+1)*26+charnumber(a[1]);
 for(int i=2;i<len;i++) s1+=a[i];
 }
 else
 {n1[1]=charnumber(a[0]);
 for(int i=1;i<len;i++) s1+=a[i];
 }
 n1[0]=Integer.parseInt(s1)-1;
 return n1;
}
public static Workbook wb1(String filename)
{   String s2=word(filename);
    Workbook Work=new HSSFWorkbook();
    try {
    if(s2.equals("xls"))
    {Work=new HSSFWorkbook();}
    else
    {Work= new SXSSFWorkbook() ;}
    } catch (Exception e) {e.printStackTrace();}
    return Work;
}
     public static String word(String s)
     {  //returns extension group of file name
     StringTokenizer token=new StringTokenizer(s,".");
     int m=token.countTokens();
     String b[]=new String[m];
     int j=0;
     while(token.hasMoreTokens())
```

```
    {
    String ax=(String)token.nextToken();
    b[j++]=ax;
    }
    return b[m-1];
    }
public static void main(String arg[])
{
    String s[][]={{"0","1","2","3","4","5","6","7","8","9"},
                {"10","11","12","13","14","15","16","17","18","19"},
                {"20","21","22","23","24","25","26","27","28","29"}};
    write_to_excel("DATA2.xls",0,"A1",s);;
}
}
```



17.3 UPDATING THE SPREADSHEET

In order to update data first read data make the changes then close the communucation channel and reopened it as write channel to apply the changes. For example if the salaries of the following spreadsheet should be

	A	B	C	D
1	Emp No.	Name	Salary	rate
2	1	John	10000	0.1
3	2	Sam	20000	0.15
4	3	Dean	30000	0.2
5				

PROGRAM 17.5 an updating example of data into spreadsheet

```
 import org.apache.poi.ss.usermodel.*;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI6
{
public static void main(String arg[])
{
try {
   FileInputStream file = new FileInputStream(new File("new.xls"));

   Workbook workbook = new HSSFWorkbook(file);
```

```
    Sheet sheet = workbook.getSheetAt(0);
    Cell cell = null;

    //Update the value of cell
    double a[]=new double[3];
    double r[]=new double[3];
    for(int i=0;i<3;i++)
     {
                cell = sheet.getRow(i+1).getCell(3);
     r[i]=cell.getNumericCellValue();
     cell = sheet.getRow(i+1).getCell(2);
     a[i]=cell.getNumericCellValue();
     a[i]*=(1+r[i]);
     cell.setCellValue(a[i]);
     }
    file.close();

    FileOutputStream outFile =new FileOutputStream(new File("new.xls"));
    workbook.write(outFile);
    outFile.close();

} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
}
```

After the updating the spreadsheet will look as:

	A	B	C	D
1	Emp No.	Name	Salary	rate
2	1	John	11000	0.1
3	2	Sam	23000	0.15
4	3	Dean	36000	0.2

17.4 ADDING AND USİNG FORMULAS

Spreadsheet formulas can be add and use easily. IN the following example, data read from new.xls and copied into new1.xls, forthermore a new column E is added and a new salary formula is integrated and calculated. The Input spreadsheet new.xls will look like

	A	B	C	D	E
1	Emp No.	Name	Salary	rate	new_salary
2	1	John	10000	0.1	
3	2	Sam	20000	0.15	
4	3	Dean	30000	0.2	

PROGRAM 17.6 reading and writing into spreadsheet (Formulas using to recalculate)

```
 import org.apache.poi.ss.usermodel.*;
import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.xssf.usermodel.*;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellReference;
import org.apache.poi.xssf.streaming.SXSSFWorkbook;
import java.io.*;
import java.util.*;
import javax.swing.*;
// reading from excel
public class apachePOI7
```

```
{
public static void main(String arg[])
{ String header[]=new String[5];
  Cell cell = null;
  Cell cell1 = null;
  Workbook workbook=null;
  Sheet sheet=null;
     double a[]=new double[3];
    double r[]=new double[3];
    String s[]=new String[3];
    double n[]=new double[3];
       String name[]=new String[3];
    String r_ref,a_ref;
try {
    FileInputStream file = new FileInputStream(new File("C:\\okul\\sco1\\new.xls"));
    workbook = new HSSFWorkbook(file);
    sheet = workbook.getSheetAt(0);
    for(int j=0;j<5;j++)
    {cell = sheet.getRow(0).getCell(j);
     header[j]=cell.getStringCellValue();
    }

    for(int i=0;i<3;i++)
      {
                cell = sheet.getRow(i+1).getCell(3);
      r[i]=cell.getNumericCellValue();
      cell = sheet.getRow(i+1).getCell(2);
      a[i]=cell.getNumericCellValue();
      cell = sheet.getRow(i+1).getCell(1);
      name[i]=cell.getStringCellValue();
      cell = sheet.getRow(i+1).getCell(0);
      n[i]=cell.getNumericCellValue();
      r_ref=CellReference.convertNumToColString(3)+(i+2);
      a_ref=CellReference.convertNumToColString(2)+(i+2);
      s[i]=a_ref+"*(1.0+"+r_ref+")";
      System.out.println(r_ref+" "+a_ref+"s="+s[i]);
      }
    file.close();
    FileOutputStream outFile =new FileOutputStream(new File("C:\\okul\\sco1\\new1.xls"));
    HSSFWorkbook workbook1 = new HSSFWorkbook();
    Sheet sheet1 = workbook1.createSheet();
    Row head = sheet1.createRow(0);
    for(int j=0;j<5;j++)
    {  System.out.println("j="+j+" "+header[j]);
                head.createCell(j).setCellValue(header[j]);
                cell1 = head.getCell(j);
                System.out.println(cell1.getStringCellValue());
              }
     for(int i=0;i<3;i++)
     { Row r1=sheet.createRow(i+1);
                r1.createCell(0).setCellValue(n[i]);
                r1.createCell(1).setCellValue(name[i]);
                r1.createCell(2).setCellValue(a[i]);
                r1.createCell(3).setCellValue(r[i]);
                r1.createCell(4).setCellFormula(s[i]);
      }
    workbook.write(outFile);
    outFile.close();

} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
}
```

Output spreadsheet new1.xls :

	A	B	C	D	E
1	Emp No.	Name	Salary	rate	new_salary
2	1	John	10000	0.1	11000
3	2	Sam	20000	0.15	23000
4	3	Dean	30000	0.2	36000

17.5 ADDING FONT AND FORMAT INFORMATION INTO SPREADHEETS

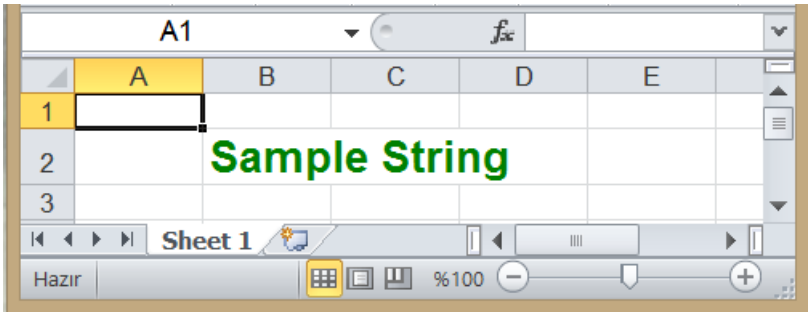
The previous sections illustrates the fundamentals of generating an Spreadsheet compatible spreadsheet using the apache POI. However, as it stands Spreadsheet will render the data in the default font. In order to supply formatting information to Spreadsheet, we must make use of the overloaded constructor, which takes an additional object containing the cell's formatting information (both the font and the style).

The code below illustrates creating a label cell for an arial 16 point font with GREEN color

PROGRAM 17.7 creating a label cell

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;

import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.hssf.util.*;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.ss.util.*;

public class apachePOI8 {

   public static void main(String[] args) {

      Workbook workbook = new HSSFWorkbook();

      Sheet firstSheet = workbook.createSheet("Sheet 1");

      // Write a String in Cell 2B
      Row row1 = firstSheet.createRow(1);
      Cell cell2B = row1.createCell(1);
      cell2B.setCellValue(new HSSFRichTextString("Sample String"));

      // Style Font in Cell 2B
      CellStyle cellStyle = workbook.createCellStyle();
      cellStyle = workbook.createCellStyle();
      Font hSSFFont = workbook.createFont();
      hSSFFont.setFontName(HSSFFont.FONT_ARIAL);
      hSSFFont.setFontHeightInPoints((short) 16);
      hSSFFont.setBoldweight(HSSFFont.BOLDWEIGHT_BOLD);
      hSSFFont.setColor(HSSFColor.GREEN.index);
      cellStyle.setFont(hSSFFont);
      cell2B.setCellStyle(cellStyle);

      FileOutputStream fileOutputStream = null;
      try {
         fileOutputStream = new FileOutputStream(new File("Test5.xls"));
         workbook.write(fileOutputStream);
      } catch (IOException e) {
         e.printStackTrace();
      } finally {
         if (fileOutputStream != null) {
```

```
            try {
                fileOutputStream.flush();
                fileOutputStream.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
  }

}
```

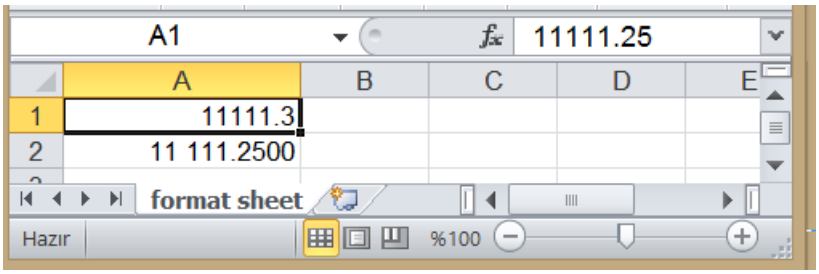


Number formatting information may be passed to the cell format object by a similar mechanism to that described for fonts.

PROGRAM 17.8 number formatting

```
import java.io.*;
import java.util.*;
import javax.swing.*;

import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.hssf.util.*;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.ss.util.*;

public class apachePOI9 {

   public static void main(String[] args) {

Workbook wb = new HSSFWorkbook();
Sheet sheet = wb.createSheet("format sheet");
CellStyle style;
DataFormat format = wb.createDataFormat();
Row row;
Cell cell;
short rowNum = 0;
short colNum = 0;

row = sheet.createRow(rowNum++);
cell = row.createCell(colNum);
cell.setCellValue(11111.25);
style = wb.createCellStyle();
style.setDataFormat(format.getFormat("0.0"));
cell.setCellStyle(style);

row = sheet.createRow(rowNum++);
cell = row.createCell(colNum);
cell.setCellValue(11111.25);
style = wb.createCellStyle();
style.setDataFormat(format.getFormat("#,##0.0000"));
cell.setCellStyle(style);
```

```
try{
FileOutputStream fileOut = new FileOutputStream("workbook.xls");
wb.write(fileOut);
fileOut.close();
}catch (FileNotFoundException e) {e.printStackTrace();}
 catch (IOException e1) {e1.printStackTrace();}
}
}
```

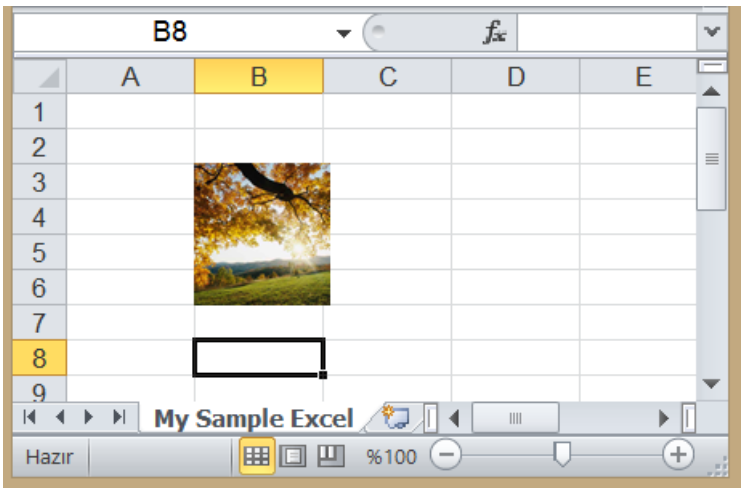


A Picture can be added into the spreadsheet by using anchor statement

PROGRAM 17.9 apachePO10 program, an example of adding a picture into spreadsheet

```
 import java.io.*;
import java.util.*;
import javax.swing.*;

import org.apache.poi.hssf.usermodel.*;
import org.apache.poi.hssf.util.*;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.ss.util.*;
import org.apache.poi.util.IOUtils;

public class apachePOI10 {

   public static void main(String[] args) {
  Workbook wb = new HSSFWorkbook();
  Sheet sheet = wb.createSheet("My Sample Excel");
  InputStream inputStream=null;
  try{
  inputStream = new FileInputStream("fall.jpg");
  }catch (FileNotFoundException e) {e.printStackTrace();}
  int pictureIdx=0;
 try{
   byte[] bytes = IOUtils.toByteArray(inputStream);
  pictureIdx = wb.addPicture(bytes, Workbook.PICTURE_TYPE_PNG);
  inputStream.close();
  }catch (IOException e1) {e1.printStackTrace();}
  CreationHelper helper = wb.getCreationHelper();
  Drawing drawing = sheet.createDrawingPatriarch();
  ClientAnchor anchor;
  anchor=new HSSFClientAnchor(0,0,0,255,(short)1,2,(short)7,10);
   anchor.setAnchorType(2);
   //= helper.createClientAnchor();
  //set top-left corner for the image
  anchor.setCol1(1);
  anchor.setRow1(2);
  Picture pict = drawing.createPicture(anchor, pictureIdx);
  pict.resize(0.1);
  try{
  FileOutputStream fileOut = new FileOutputStream("picture.xls");
```

```
  wb.write(fileOut);
  fileOut.close();
  }catch (FileNotFoundException e) {e.printStackTrace();}
   catch (IOException e1) {e1.printStackTrace();}

}
}
```



EXERCISE 17 (WEEK 8)

EX1) Temperature and specific heat of CO_2 gas is given in file co2_cv.xls . Write an excell program to read the data and print it out to the screen

EX 2) : Create table **sinx** with 2 column x and y. Enter x column values . 0.1 0.2 .03 0.4 0.5 , enter all y values as sin(x). Write the result into data6.xls spreadsheet program (excel)

EX3) : Add also Picture fall.png into data6.xls spreadsheet

HOMEWORKS 17 (WEEK 8)

HW 1 : Create spreadsheet **student.xls** with data fields **name , surname , grade1, grade 2, result**

Input the following data

'Ali' 'Çiçek' 24 58

'Veli' 'Durmuş' 49 73

'Hasan' 'Yücel' 72 68

'Mehmet' 'Demir' 63 55

Write a java program to read the data from **student.xls** . Calculate result as 40% grade1 and 60% grade2 and write down into spreadsheet result column the resulting data.

HW 2 : read the data from co2_cv.xls , Calculate avarage of temperature and Cv and write the results into co2_cv.xls.

.

18. REAL TIME PROGRAMMING AND MULTITHREADING

18.1 PROGRAMMING WITH THREADS

If all the real world processes carried out in a sequential order, some good results might be obtained out of it. Unfortunately real world processes seldomly sequential, usually happend and operates paralel to each other. Most of the natural systems including human body Works in paralel orders. When one process continues, another action goes on paralel to it. Modern coputers start to utilised papallel processing nowadays. It is possible tos hare the tasks in between different processors. Another way to paralel use of the programs is time sharing in one processor. Multithreading can be defined as using one processor (Central Processing Unit) by several programs in the same time, by dividing CPU time into small process time packages and assign each one several package to one program opratation while assigning other small process time packages(Threads) into other tasks or programs. If only one processor in operations when one thread operates, the others waits in a row, but if the computer speed is fast enough, it could give the impresion of paralel processing. While the CPU process one thread, the other program parts waits either in cach memory or other memory units. Multithreading usualy controls through operating systems. A programming language can achive multithreading through using opearing system facilities, bu in this case program can functions only for one operating system. Java is an operating system free language, therefore all the multithreading programming features are included inside the language. Java can operate multithreading through Thread class or Runnable interface. In table an example of running different programs in CPU threads (time sharing parts)

TABLE 18.1 A thread structure example

Thread1	Thread2	Thread3	Thread4	Thread5	Thread6	Thread7	Thread8	Thread9
PR1	PR1	PR2	PR3	PR1	PR2	PR2	PR3	PR3

An example program utilising threads are given as the following program. Class worker is a Thread program (extended from the class Thread). Two object of this class is created, and started by using start() method. In this one all the methods has the same priority so threads are shared equally between the objects

PROGRAM 18.1 Thread class example

```
class worker extends Thread
{
 public void run()
 { for(int i=0;i<5;i++) {System.out.println(this.getName()+"-->"+i);}}
}

public class C18E1
{
public static void main(String args[])
{worker i1=new worker();
 worker i2=new worker();
 i1.start();
 i2.start();
}
}
```

```
---------- Capture Output ----------
```

```
> "C:\turhan\java\bin\javaw.exe" C18E1
Thread-0-->0
Thread-1-->0
Thread-0-->1
Thread-1-->1
Thread-0-->2
Thread-1-->2
Thread-0-->3
Thread-1-->3
Thread-0-->4
Thread-1-->4

> Terminated with exit code 0.
```

Methods of Thread class is given in the following table

TABLE 18.2 Field and ethods of Thread class

Field Summary	
static int	MAX_PRIORITY The maximum priority that a thread can have.
static int	MIN_PRIORITY The minimum priority that a thread can have.
static int	NORM_PRIORITY The default priority that is assigned to a thread.

Constructor Summary
Thread() Allocates a new Thread object.
Thread(Runnable target) Allocates a new Thread object.
Thread(Runnable target, String name) Allocates a new Thread object.
Thread(String name) Allocates a new Thread object.
Thread(ThreadGroup group, Runnable target) Allocates a new Thread object.
Thread(ThreadGroup group, Runnable target, String name) Allocates a new Thread object so that it has target as its run object, has the specified name as its name, and belongs to the thread group referred to by group.

Thread(ThreadGroup group, Runnable target, String name, long stackSize) Allocates a new Thread object so that it has target as its run object, has the specified name as its name, belongs to the thread group referred to by group, and has the specified *stack size*.	
Thread(ThreadGroup group, String name) Allocates a new Thread object.	

Method Summary	
static int	activeCount() Returns the number of active threads in the current thread's thread group.
void	checkAccess() Determines if the currently running thread has permission to modify this thread.
int	countStackFrames() **Deprecated.** *The definition of this call depends on* suspend(), *which is deprecated. Further, the results of this call were never well-defined.*
static Thread	currentThread() Returns a reference to the currently executing thread object.
void	destroy() Destroys this thread, without any cleanup.
static void	dumpStack() Prints a stack trace of the current thread.
static int	enumerate(Thread[] tarray) Copies into the specified array every active thread in the current thread's thread group and its subgroups.
ClassLoader	getContextClassLoader() Returns the context ClassLoader for this Thread.
String	getName() Returns this thread's name.
int	getPriority() Returns this thread's priority.
ThreadGroup	getThreadGroup() Returns the thread group to which this thread belongs.
static boolean	holdsLock(Object obj) Returns true if and only if the current thread holds the monitor lock on the specified object.
void	interrupt()

	Interrupts this thread.
static boolean	interrupted() Tests whether the current thread has been interrupted.
boolean	isAlive() Tests if this thread is alive.
boolean	isDaemon() Tests if this thread is a daemon thread.
boolean	isInterrupted() Tests whether this thread has been interrupted.
void	join() Waits for this thread to die.
void	join(long millis) Waits at most millis milliseconds for this thread to die.
void	join(long millis, int nanos) Waits at most millis milliseconds plus nanos nanoseconds for this thread to die.
void	resume() **Deprecated.** *This method exists solely for use with* suspend(), *which has been deprecated because it is deadlock-prone. For more information, see* Why are Thread.stop, Thread.suspend and Thread.resume Deprecated?.
void	run() If this thread was constructed using a separate Runnable run object, then that Runnable object's run method is called; otherwise, this method does nothing and returns.
void	setContextClassLoader(ClassLoader cl) Sets the context ClassLoader for this Thread.
void	setDaemon(boolean on) Marks this thread as either a daemon thread or a user thread.
void	setName(String name) Changes the name of this thread to be equal to the argument name.
void	setPriority(int newPriority) Changes the priority of this thread.
static void	sleep(long millis) Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds.
static void	sleep(long millis, int nanos) Causes the currently executing thread to sleep (cease execution) for the specified number of milliseconds

	plus the specified number of nanoseconds.
void	start() Causes this thread to begin execution; the Java Virtual Machine calls the run method of this thread.
void	stop() **Deprecated.** *This method is inherently unsafe. Stopping a thread with Thread.stop causes it to unlock all of the monitors that it has locked (as a natural consequence of the unchecked ThreadDeath exception propagating up the stack). If any of the objects previously protected by these monitors were in an inconsistent state, the damaged objects become visible to other threads, potentially resulting in arbitrary behavior. Many uses of stop should be replaced by code that simply modifies some variable to indicate that the target thread should stop running. The target thread should check this variable regularly, and return from its run method in an orderly fashion if the variable indicates that it is to stop running. If the target thread waits for long periods (on a condition variable, for example), the interrupt method should be used to interrupt the wait. For more information, see* Why are Thread.stop, Thread.suspend and Thread.resume Deprecated?.
void	stop(Throwable obj) **Deprecated.** *This method is inherently unsafe. See* stop() *(with no arguments) for details. An additional danger of this method is that it may be used to generate exceptions that the target thread is unprepared to handle (including checked exceptions that the thread could not possibly throw, were it not for this method). For more information, see* Why are Thread.stop, Thread.suspend and Thread.resume Deprecated?.
void	suspend() **Deprecated.** *This method has been deprecated, as it is inherently deadlock-prone. If the target thread holds a lock on the monitor protecting a critical system resource when it is suspended, no thread can access this resource until the target thread is resumed. If the thread that would resume the target thread attempts to lock this monitor prior to calling resume, deadlock results. Such deadlocks typically manifest themselves as "frozen" processes. For more information, see* Why are Thread.stop, Thread.suspend and Thread.resume Deprecated?.
String	toString() Returns a string representation of this thread, including the thread's name, priority, and thread group.
static void	yield() Causes the currently executing thread object to temporarily pause and allow other threads to execute.

Methods inherited from class java.lang. Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

In the second example Thread.sleep method is used. When thread sleeps it is inactivated and waits to be activate again for a designated time(in microseconds)

PROGRAM 18.2 Thread class example

```
class worker1 extends Thread
{
 String isim;
```

```
 public worker1(String isim1)
 {isim=isim1;}

 public void run()
 { for(int i=0;i<5;i++) {
            try
            {
            if(isim.equals("ali")) Thread.sleep(1000);
            if(isim.equals("veli")) Thread.sleep(2000);
    } catch(InterruptedException ex) {System.out.println("error...");}
            System.out.println(this.getName()+"-->"+i);}}
}

public class C18E2
{
public static void main(String args[])
{worker1 i1=new worker1("ali");
 worker1 i2=new worker1("veli");
 i1.start();
 i2.start();
}
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" C18E2
Thread-0-->0
Thread-0-->1
Thread-1-->0
Thread-0-->2
Thread-1-->1
Thread-0-->3
Thread-0-->4
Thread-1-->2
Thread-1-->3
Thread-1-->4

> Terminated with exit code 0.
```

In the next example different thread priorities are tried. MIN_PRORITY gives slower Access.

PROGRAM 18.3 Thread class example with changing thread priorities

```
class Robot extends Thread
{
 public Robot(String isim) {super(isim);}
 public void run()
 { try{
            for(int i=0;i<10;i++)
   {System.out.println("Priority: "+this.getPriority()+this.getName()+"-->"+i);}
    } catch(Exception ex) {System.out.println("error..");}
}
}

public class C18E3
{
public static void main(String args[])
```

```
{Robot r1=new Robot("A");
 Robot r2=new Robot("B");
 Robot r3=new Robot("C");
 Robot r4=new Robot("D");
 r1.setPriority(Thread.MIN_PRIORITY);
 r2.setPriority(Thread.NORM_PRIORITY);
 r3.setPriority(Thread.MAX_PRIORITY);
 r4.setPriority(Thread.MAX_PRIORITY-2);

 r1.start();
 r2.start();
 r3.start();
 r4.start();
}
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" C18E3
Priority: 1A-->0
Priority: 5B-->0
Priority: 5B-->1
Priority: 10C-->0
Priority: 1A-->1
Priority: 10C-->1
Priority: 8D-->0
Priority: 5B-->2
Priority: 8D-->1
Priority: 10C-->2
Priority: 1A-->2
Priority: 10C-->3
Priority: 8D-->2
Priority: 5B-->3
Priority: 8D-->3
Priority: 10C-->4
Priority: 1A-->3
Priority: 10C-->5
Priority: 8D-->4
Priority: 5B-->4
Priority: 8D-->5
Priority: 10C-->6
Priority: 1A-->4
Priority: 10C-->7
Priority: 8D-->6
Priority: 5B-->5
Priority: 8D-->7
Priority: 10C-->8
Priority: 1A-->5
Priority: 10C-->9
Priority: 8D-->8
Priority: 5B-->6
Priority: 8D-->9
Priority: 1A-->6
Priority: 5B-->7
Priority: 1A-->7
Priority: 5B-->8
Priority: 1A-->8
Priority: 5B-->9
```

```
Priority: 1A-->9
> Terminated with exit code 0.
```

Now a computer clock can be programmed by using Thread.sleep method. If Thread wait 1000 microseconds in each period, a second counter is obtained.

PROGRAM 18.4 Thread class example a very simple clock

```
class clock extends Thread
{
 int second;
 int minute;
 long counter;
 public clock()
 {counter=0;}

 public void run()
 { while(true)
   {
            try
            {
            second=(int)counter%60;
            minute=(int)counter/60;
            Thread.sleep(1000);
            counter++;
    } catch(InterruptedException ex) {System.out.println("error...");}
            System.out.println("minute : "+minute+" second : "+second);
   }
 }
}

public class C18E5
{
public static void main(String args[])
{clock c1=new clock();
 c1.start();
}
}
```

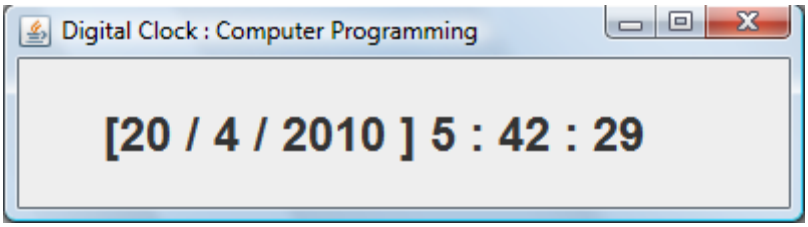
```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" C18E5
minute : 0second : 0
minute : 0second : 1
minute : 0second : 2
minute : 0second : 3
minute : 0second : 4
minute : 0second : 5
minute : 0second : 6
minute : 0second : 7
minute : 0second : 8
```

A clock program can be made more sofisticated if time is read from the java Date class. System stil waits 1000 microsecond before reactivating. Output is a graphic screen.

PROGRAM 18.5 Thread class example a digital clock with graphical output

```
import java.awt.*;
 import java.awt.event.*;
 import java.awt.geom.*;
 import javax.swing.*;
 import java.net.*;
 import java.io.*;
 import java.awt.Graphics;
 import java.util.*;
 import java.text.*;
 import java.util.Locale;

 class clock1 extends Thread
 {
 JFrame a;
 public clock1(JFrame ai)
 {a=ai;}

 public void run()
 {
 while(true)
 {
 a.validate();
 a.repaint();
 try {Thread.sleep(1000);}
 catch(InterruptedException e)
 {System.err.println(e.toString());}
 }
 }
 }

 public class C18E6 extends JFrame implements Serializable
 {
 JLabel jb;
 Font F;
 Date D;
 clock1 r;
 GregorianCalendar calender;

 public C18E6()
 {
 super("Digital Clock : Computer Programming");
 Container cc=getContentPane();
 jb=new JLabel("");
 F=new Font("TimesRoman",Font.BOLD,24);
 jb.setFont(F);
 // Türkiye standart (Greenwitch zamanına göre iki saat ileride)
 String[] ids = TimeZone.getAvailableIDs(+3 * 60 * 60 * 1000);
 SimpleTimeZone pdt = new SimpleTimeZone(+3 * 60 * 60 * 1000, ids[0]);
 calender=new GregorianCalendar(pdt);
 D=new Date();
 calender.setTime(D);
 r=new clock1(this);
 cc.add(jb);
```

```
r.start();
}

public void paint(Graphics g)
{
   D=new Date();
   calender.setTime(D);
   String s="     ["+
   calender.get(Calendar.DATE) +" / "+
   (calender.get(Calendar.MONTH)+1)+" / "+
   calender.get(Calendar.YEAR) +" ] "+
   calender.get(Calendar.HOUR_OF_DAY)+" : "+
   calender.get(Calendar.MINUTE)+" : "+
   calender.get(Calendar.SECOND);
   jb.setText(s);
}
  public static void main(String s[])
  {
              C18E6 f = new C18E6();
     f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     f.setSize(400,110);
     f.setVisible(true);
  }
}
```



Another method of creating threads is through **Runnable interface**. Extends can be used only once in a class creation. If a second library addition is needed, implements is used. Runnable is the implemented version for Thread use. To compare and see the differences, A runnable version of some of the programs given earlier is listed here

PROGRAM 18.1A Runnable interface Thread class example

```
class worker implements Runnable
{
 Thread t;
 public worker()
 {t=new Thread(this);t.start();}
 public void run()
 { for(int i=0;i<5;i++) {System.out.println(t.getName()+"-->"+i);}}
}

public class C18E1A
{
public static void main(String args[])
{worker i1=new worker();
```

```
 worker i2=new worker();
}
}
```

One advantage of Runnable Interface is the ability of starting threads by using Lambda expressions. Lambda version of the same program with Anonymous Thread classes is given as the next example:

PROGRAM 18.1B Runnable interface Thread class example with anonymous inner class Thread and Runnable class lambda expressions

```
public class C18E1B
{
public static void main(String args[])
{ new Thread(() -> {for(int i=0;i<20;i++){System.out.println("thread 1 i = "+i);}}).start();
  new Thread(() -> {for(int i=0;i<20;i++){System.out.println("thread 2 i = "+i);}}).start();
}
}
```

```
---------- Capture Output ----------
> "C:\java\bin\java.exe" C18E1B
thread 1 i = 0
thread 1 i = 1
thread 1 i = 2
thread 1 i = 3
thread 1 i = 4
thread 1 i = 5
thread 1 i = 6
thread 2 i = 0
thread 2 i = 1
thread 2 i = 2
thread 2 i = 3
thread 2 i = 4
thread 1 i = 7
thread 1 i = 8
thread 1 i = 9
thread 1 i = 10
thread 1 i = 11
thread 1 i = 12
thread 1 i = 13
thread 2 i = 5
thread 2 i = 6
thread 2 i = 7
thread 2 i = 8
thread 2 i = 9
thread 2 i = 10
thread 2 i = 11
thread 2 i = 12
```

```
thread 1 i = 14
thread 1 i = 15
thread 2 i = 13
thread 2 i = 14
thread 2 i = 15
thread 2 i = 16
thread 1 i = 16
thread 1 i = 17
thread 1 i = 18
thread 1 i = 19
thread 2 i = 17
thread 2 i = 18
thread 2 i = 19

> Terminated with exit code 0.
```

Another way of writing by using Lambda Expressions, but in this version inner classes is not anonymous.

PROGRAM 18.1C Runnable interface inner Thread classes t1 and t2 and and Runnable class lambda expressions

```
public class C18E1C
{
public static void main(String args[])
{ Runnable r1=() -> {for(int i=0;i<40;i++){System.out.println("thread r1 i = "+i);}};
  Thread t1=new Thread(r1);
  t1.start();
  Runnable r2=() -> {for(int i=0;i<40;i++){System.out.println("thread r2 i = "+i);}};
  Thread t2=new Thread(r2);
  t2.start();
}
}
```

Now the second example given with Thread class is turned into a Runnable interface example:

PROGRAM 18.2A Runnable interface example

```
class worker1 implements Runnable
{ Thread t;
 String isim;
 public worker1(String isim1)
 {isim=isim1;t=new Thread(this);t.start();}

 public void run()
 { for(int i=0;i<5;i++) {
            try
```

```
            {
            if(isim.equals("ali")) Thread.sleep(1000);
            if(isim.equals("veli")) Thread.sleep(2000);
    } catch(InterruptedException ex) {System.out.println("error...");}
            System.out.println(t.getName()+"-->"+i);}}
}

public class C18E2A
{
public static void main(String args[])
{worker1 i1=new worker1("ali");
 worker1 i2=new worker1("veli");

}
}
```

A lambda expression version of the same Runnable interface program:

PROGRAM 18.2B Runnable interface example, lambda expressions version

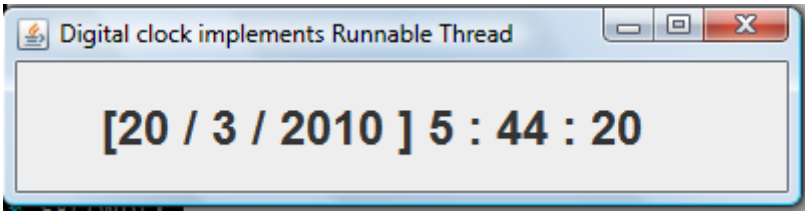
```
class worker1
{ Thread t;
 String isim;
 public worker1(String isim1,Runnable r)
 {isim=isim1;t=new Thread(r);t.start();}
}

public class C18E2B
{
public static void a(String isim)
{ for(int i=0;i<5;i++) {
            try
            {
            if(isim.equals("ali")) Thread.sleep(1000);
            if(isim.equals("veli")) Thread.sleep(2000);
    } catch(InterruptedException ex) {System.out.println("error...");}
            System.out.println(isim+"-->"+i);}
}
public static void main(String args[])
{   Runnable r1=() -> a("ali");
           worker1 i1=new worker1("ali",r1);
           Runnable r2=() -> a("veli");
   worker1 i2=new worker1("veli",r2);
}
}
```

A Runnable version of the previous Digital clock is created as the next example.

PROGRAM 18.6 Thread class and Runnable interphase example a digital clock with graphical output

```
import java.awt.Graphics;
 import java.awt.*;
 import java.util.*;
 import javax.swing.*;
 import java.text.*;
import java.awt.event.*;

 class clock2 extends Thread
 {
 JFrame a;
 public clock2(JFrame ai)
 {a=ai;}

 public void run()
 {
 while(true)
 {
 a.validate();
 a.repaint();
 try {Thread.sleep(1000);}
 catch(InterruptedException e)
 {System.err.println(e.toString());}
 }}}

 public class C18E7 extends JFrame implements Runnable
 {
 JLabel jb;
 Font F;
 Date D;
 clock2 r;
 GregorianCalendar calender;
 Thread runner;

 public C18E7()
 {
 super("Digital clock implements Runnable Thread");
 Container cc=getContentPane();
 jb=new JLabel("");
 F=new Font("TimesRoman",Font.BOLD,24);
 jb.setFont(F);
 // Türkiye standart time + 1 hour( 3 hours advanced from Greenwitch)
 String[] ids = TimeZone.getAvailableIDs(+3 * 60 * 60 * 1000);
 SimpleTimeZone pdt = new SimpleTimeZone(+3 * 60 * 60 * 1000, ids[0]);
 calender=new GregorianCalendar(pdt);
 D=new Date();
 calender.setTime(D);
 r=new clock2(this);
 cc.add(jb);
 r.start();
 }

   public void start()
   {
      if(runner==null)
      {
```

```
        runner=new Thread(this);
        runner.start();
      }
    }

    public void run()
    {
      while(true)
      {
      repaint();
      try {Thread.sleep(1000);}
      catch(InterruptedException e)
      {System.err.println(e.toString());}
      }
    }


    public void paint(Graphics g)
    {
    D=new Date();
    calender.setTime(D);
    String s="     ["+
    calender.get(Calendar.DATE) +" / "+
    calender.get(Calendar.MONTH)+" / "+
    calender.get(Calendar.YEAR) +" ] "+
    calender.get(Calendar.HOUR_OF_DAY)+" : "+
    calender.get(Calendar.MINUTE)+" : "+
    calender.get(Calendar.SECOND);
    jb.setText(s);
    }
    public static void main(String s[])
    {
      C18E7 f= new C18E7();
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      f.setSize(400,100);
      f.setVisible(true);
    }
  }
```



public interface **Runnable** has only one method

Method Summary

void	run() When an object implementing interface Runnable is used to create a thread, starting the thread causes the object's run method to be called in that separately executing thread.

To make the example more colorfull, the next example will create a analog clock by using real time graphics. In order to create the clock face polar coordinate and star class is used.

```
public class polar
{
//polar coordinates

protected double R,teta;

public polar()
{
R=0;
teta=0;
}

public polar(double Ri,double tetai)
{
R=Ri;
teta=tetai;
}

public polar(polar y)
{
R=y.ROku();
teta=y.tetaOku();
}

public void R_input(double Ri)
{
R=Ri;
}

public void teta_input(double tetai)
{
teta=tetai;
}


public void polar_input(double Ri,double tetai)
{
R=Ri;
teta=tetai;
}

public void kartesien_input(double xi,double xj)
{
R=Math.sqrt(xi*xi+xj*xj);
teta=Math.atan2(xj,xi);
}

public double ROku()
{
return R;
}

public double tetaOku()
```

```
{
return teta;
}

public double xi()
{
return R*Math.cos(teta);
}

public double xj()
{
return R*Math.sin(teta);
}

public void topla(polar y)
{
 kartesien_input((xi()+y.xi()),(xj()+y.xj()));
}


public void fark(polar y)
{
  kartesien_input((xi()-y.xi()),(xj()-y.xj()));
}

public polar polarOku()
{
return this;
}

public boolean esittir(polar v)
{
boolean b=((R==v.ROku())&&(teta==v.tetaOku()));
return b;
}

public boolean buyuktur(polar v)
{
  return (this.R>v.ROku());
}

public boolean kucuktur(polar v)
{
  return (this.R<v.ROku());
}

public String toString()
{
 return ""+R+"*exp("+teta+"i) ";
}

}
```

```
import javax.swing.*;
//import polar;
```

```
public class star
{

public static void drawYildiz1(Graphics g2,int xi,int yi, int n,int yildizboyu,double angle )
{
  // this star starts to draw from angle  teta=pi/2+angle radian

  Graphics2D g=(Graphics2D)g2;
  double teta=2.0*Math.PI/n;
  double R=yildizboyu;
  double r=yildizboyu*0.25;
  polar P1=new polar();
  polar P2=new polar();
  polar P3=new polar();
  for(int i=0;i<n;i++)
  {
    double teta1=teta*i+Math.PI/2.0+angle;
    double teta2=teta/2+teta1;
    P1.polar_input(R,teta1);
    P2.polar_input(r,teta2);
    g.drawLine((xi+(int)P1.xi()),(yi-(int)P1.xj()),
            (xi+(int)P2.xi()),(yi-(int)P2.xj()));
    double teta3=teta*(i+1)+Math.PI/2.0+angle;
    P3.polar_input(R,teta3);
    g.drawLine((xi+(int)P2.xi()),(yi-(int)P2.xj()),
            (xi+(int)P3.xi()),(yi-(int)P3.xj()));
    g.drawLine(xi,yi,
            (xi+(int)P1.xi()),(yi-(int)P1.xj()));
    g.drawLine(xi,yi,
            (xi+(int)P2.xi()),(yi-(int)P2.xj()));
  }

}  //drawYildiz1 metodu sonu

public static void drawYildiz1(Graphics g2,int xi,int yi, int n,int yildizboyu )
{
  drawYildiz1(g2,xi,yi,n,yildizboyu,0);
}  //drawYildiz1 metodu sonu

public static void drawYildiz(Graphics g2,int xi,int yi, int n,int yildizboyu,double angle )
{
  Graphics2D g=(Graphics2D)g2;
  // bu yıldız cizime teta=pi/2+angle radyandan baslar
  double teta=2.0*Math.PI/n;
  double R=yildizboyu;
  double r=yildizboyu*0.25;
  polar P1=new polar();
  polar P2=new polar();
  polar P3=new polar();
  int x[]=new int[2*n+2];
  int y[]=new int[2*n+2];
  GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
  x.length);

  for(int i=0;i<=n;i++)
  {
    double teta1=teta*i+Math.PI/2.0+angle;
```

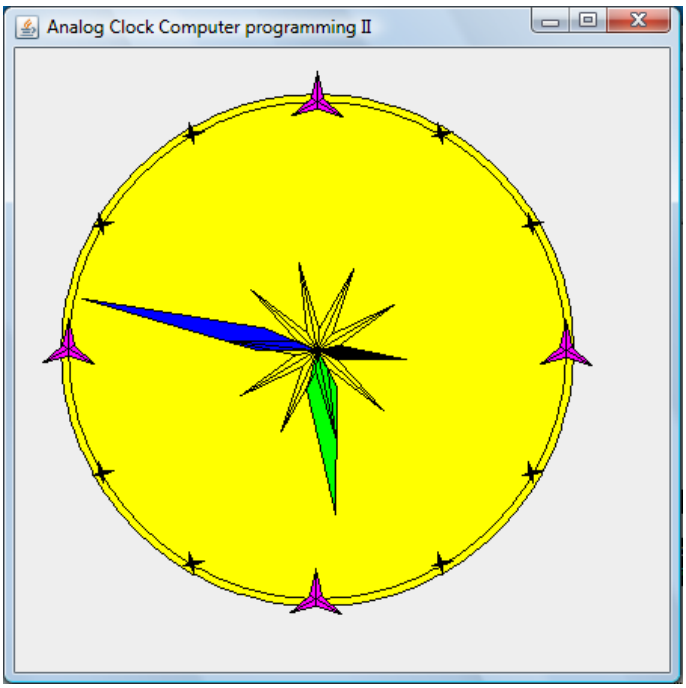
```
    double teta2=teta/2.0+teta1;
    P1.polar_input(R,teta1);
    P2.polar_input(r,teta2);
    x[2*i]=xi+(int)P1.xi();
    y[2*i]=yi-(int)P1.xj();
    x[2*i+1]=xi+(int)P2.xi();
    y[2*i+1]=yi-(int)P2.xj();
    if(i==n)
    {
    x[2*i]=x[0];
    y[2*i]=y[0];
    }
  }

  for(int i=0;i<=2*n;i++)
  {
        if(i==0)
        {
        polygon.moveTo(x[0],y[0]);
        }
        else
        {
         polygon.lineTo(x[i],y[i]);
        }
  }
  g.draw(polygon);
}  //drawYildiz1 metodu sonu


public static void fillYildiz(Graphics g2,int xi,int yi, int n,int yildizboyu,double angle )
{
  Graphics2D g=(Graphics2D)g2;
  // bu yıldız cizime teta=pi/2+angle radyandan baslar
  double teta=2.0*Math.PI/n;
  double R=yildizboyu;
  double r=yildizboyu*0.25;
  polar P1=new polar();
  polar P2=new polar();
  polar P3=new polar();
  int x[]=new int[2*n+2];
  int y[]=new int[2*n+2];
  GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
  x.length);

  for(int i=0;i<=n;i++)
  {
    double teta1=teta*i+Math.PI/2.0+angle;
    double teta2=teta/2.0+teta1;
    P1.polar_input(R,teta1);
    P2.polar_input(r,teta2);
    x[2*i]=xi+(int)P1.xi();
    y[2*i]=yi-(int)P1.xj();
    x[2*i+1]=xi+(int)P2.xi();
    y[2*i+1]=yi-(int)P2.xj();
    if(i==n)
    {
    x[2*i]=x[0];
    y[2*i]=y[0];
```

```
    }
  }

  for(int i=0;i<=2*n;i++)
  {
        if(i==0)
        {
        polygon.moveTo(x[0],y[0]);
        }
        else
        {
         polygon.lineTo(x[i],y[i]);
        }
  }
  g.fill(polygon);

}  //fillYildiz metodu sonu

public static void fillAkrep(Graphics g2,int xi,int yi, int n,int yildizboyu,double angle,int i )
{
  // bu yıldız cizime teta=pi/2+angle radyandan baslar
  // yildızin içini boyar
  Graphics2D g=(Graphics2D)g2;
  double teta=2.0*Math.PI/n;
  double R=yildizboyu;
  double r=yildizboyu*0.25;
  polar P1=new polar();
  polar P2=new polar();
  polar P3=new polar();
  int x[]=new int[5];
  int y[]=new int[5];
  GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
  x.length);
  x[0]=xi;
  y[0]=yi;
  x[4]=xi;
  y[4]=yi;
    double teta1=teta*i+Math.PI/2.0+angle;
    double teta2=teta1+teta/2.0;
    double teta3=teta1-teta/2+Math.PI*2;
    P1.polar_input(R,teta1);
    P2.polar_input(r,teta2);
    P3.polar_input(r,teta3);
    x[1]=xi+(int)P3.xi();
    y[1]=yi-(int)P3.xj();
    x[2]=xi+(int)P1.xi();
    y[2]=yi-(int)P1.xj();
    x[3]=xi+(int)P2.xi();
    y[3]=yi-(int)P2.xj();
  for(i=0;i<5;i++)
  {
        if(i==0)
        {
        polygon.moveTo(x[0],y[0]);
        }
        else
        {
         polygon.lineTo(x[i],y[i]);
```

```
        }
    }
    g.fill(polygon);
}  //fillAkrep metodu sonu

public static void drawAkrep(Graphics g2,int xi,int yi, int n,int yildizboyu,double angle,int i )
{
    Graphics2D g=(Graphics2D)g2;
    double teta=2.0*Math.PI/n;
    double R=yildizboyu;
    double r=yildizboyu*0.25;
    polar P1=new polar();
    polar P2=new polar();
    polar P3=new polar();
    int x[]=new int[5];
    int y[]=new int[5];
    GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
    x.length);
    x[0]=xi;
    y[0]=yi;
    x[4]=xi;
    y[4]=yi;
      double teta1=teta*i+Math.PI/2.0+angle;
      double teta2=teta1+teta/2.0;
      double teta3=teta1-teta/2+Math.PI*2;
      P1.polar_input(R,teta1);
      P2.polar_input(r,teta2);
      P3.polar_input(r,teta3);
      x[1]=xi+(int)P3.xi();
      y[1]=yi-(int)P3.xj();
      x[2]=xi+(int)P1.xi();
      y[2]=yi-(int)P1.xj();
      x[3]=xi+(int)P2.xi();
      y[3]=yi-(int)P2.xj();
    for(i=0;i<5;i++)
    {
        if(i==0)
        {
        polygon.moveTo(x[0],y[0]);
        }
        else
        {
         polygon.lineTo(x[i],y[i]);
        }
    }
    g.draw(polygon);

} //drawAkrep metodu sonu

public static void drawDaire(Graphics g2, int xi,int yi, int R)
{
    Graphics2D g=(Graphics2D)g2;
    g.draw(new Ellipse2D.Double((int)(xi-R),(int)(yi-R),2*R,2*R));
}

public static void fillDaire(Graphics g2, int xi,int yi, int R)
{
    Graphics2D g=(Graphics2D)g2;
```

```
   g.fill(new Ellipse2D.Double((int)(xi-R),(int)(yi-R),2*R,2*R));
}
}
```

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

public class AnalogClock extends JFrame
{
  Container c;
  JPanel d;

 public AnalogClock(String a)
 {
   super(a);
   c=getContentPane();
   d=new analogClockP();
   c.add(d);
 }

   public static void main(String s[])
   {
                AnalogClock f = new AnalogClock("Analog Clock Computer programming II");
      f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      f.setSize(450,450);
      f.setVisible(true);
   }
}
```



18.2 SYNCHRONISATION

When more than one program thread will process a common memory variable, we can not be ensured the order of access to the data. Thing about a common data problem which a common data point is created by one –thread and a second thread read the data. When a new data is created the ol done is destroyed so that second will read the latest data created. The First version of this program is given below. Program consist of classes CommonCell, ProduceInteger , ConsumeInteger and CreateInteger. The adres of the CreatedInteger is Shared by ProduceInteger and ConsumeInteger classes

18.2-1 CommonCell program

```
import java.util.Random;
class CommonCell
{
   public static void main(String args[])
   {
      CreateInteger h=new CreateInteger();
      ProduceInteger p=new ProduceInteger(h);
      ConsumeInteger c= new ConsumeInteger(h);

      p.start();
      c.start();
   }
}

class ProduceInteger extends Thread
{
   private CreateInteger pCatch;
   int total;

   public ProduceInteger(CreateInteger h)
   {
      pCatch=h;
      total=0;
   }

   public synchronized void run()
   {
      for(int count=0; count<10; count++)
      {
         pCatch.setCommonint(count);
         total+=count;
         System.out.println("Produce integer produced  "+count+"  Total produced = "+total);

        try{
          sleep( (int) (Math.random()*5000 ) );
          }
          catch(InterruptedException e)
            {
            System.err.println("Exeption "+e.toString());
            }
      }
   }
}

class ConsumeInteger extends Thread
{
   private CreateInteger cCatch;
   int total;
```

```
    public ConsumeInteger(CreateInteger h)
    {
        cCatch=h;
    }

    public void run()
    {
        int intvalue=0;


        while (intvalue !=9)
        {
            try{
                sleep( (int) (Math.random()*5000));
               }
            catch(InterruptedException e)
            {
                System.err.println("Exeption "+e.toString());
            }
        intvalue=cCatch.getCommonInteger();
        total+=intvalue;
        System.out.println("Consume integer catched : "+intvalue+"  Total consumed = "+total);

       }
    }
}

class CreateInteger
{
    private int Commonint;

    public  void setCommonint(int intvalue)
    {
        Commonint=intvalue;
    }

    public int getCommonInteger()
    {
        return Commonint;
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" CommonCell
Produce integer produced  0  Total produced = 0
Produce integer produced  1  Total produced = 1
Consume integer catched : 1  Total consumed = 1
Produce integer produced  2  Total produced = 3
Consume integer catched : 2  Total consumed = 3
Produce integer produced  3  Total produced = 6
Consume integer catched : 3  Total consumed = 6
Produce integer produced  4  Total produced = 10
Consume integer catched : 4  Total consumed = 10
Produce integer produced  5  Total produced = 15
```

```
Produce integer produced  6  Total produced = 21
Consume integer catched : 6  Total consumed = 16
Produce integer produced  7  Total produced = 28
Consume integer catched : 7  Total consumed = 23
Produce integer produced  8  Total produced = 36
Consume integer catched : 8  Total consumed = 31
Consume integer catched : 8  Total consumed = 39
Produce integer produced  9  Total produced = 45
Consume integer catched : 9  Total consumed = 48

> Terminated with exit code 0.
```

As it is seen from the list Consume integer is never get Access to number 5 it reads 4 and then 6, because produceInteger thread operates twice. Also ConsumeInteger get 8 twice. It is an Unsysnchronised program. A second version of the program is given with Runnable Interphase

18.2-2 CommonCellRunnable program

```
İmport java.util.*;
import java.awt.*;
import java.io.*;
//runnable interface

class ProduceInteger implements Runnable {
   private CreateInteger pTut;
   private Thread thread;
   public ProduceInteger(CreateInteger h) {
      pTut=h;
      thread = new Thread(this);
      thread.start();

   }
   public void run() {
      for(int count=0; count < 10; count++) {
         pTut.setCommonInteger(count);
         System.out.println("Produce integer produced  " + count);

      try {
         thread.sleep((int)(Math.random()*3000));
      }
      catch(InterruptedException exception) {
         System.err.println("Exception " + exception.toString());
      }
      }
   }
}
class ConsumeInteger implements Runnable{
   private CreateInteger cTut;
   private Thread thread;
   public ConsumeInteger(CreateInteger h) {
      cTut = h;
      thread= new Thread(this);
      thread.start();
   }
   public void run() {
      int IntValue=0;
      while(IntValue != 9) {
```

```
            try {
                thread.sleep((int)(Math.random()*3000));
            }
            catch(InterruptedException e) {
                System.err.println("Exception " + e.toString());
            }
            IntValue = cTut.getCommonInteger();
            System.out.println("Consume integer catched  " + IntValue);

        }
    }
}

class CreateInteger {
    private int CommonInteger;
    public void setCommonInteger(int IntValue) {CommonInteger = IntValue;}
    public int getCommonInteger() { return CommonInteger;}
}

public class CommonCellRunnable {
    public static void main(String[] args) {
        CreateInteger h= new CreateInteger();
        new ProduceInteger(h);
        new ConsumeInteger(h);
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" CommonCellRunnable
Produce integer produced  0
Produce integer produced  1
Consume integer catched  1
Consume integer catched  1
Produce integer produced  2
Consume integer catched  2
Consume integer catched  2
Produce integer produced  3
Produce integer produced  4
Consume integer catched  4
Produce integer produced  5
Consume integer catched  5
Consume integer catched  5
Produce integer produced  6
Produce integer produced  7
Produce integer produced  8
Consume integer catched  8
Produce integer produced  9
Consume integer catched  9
> Terminated with exit code 0.
```

This time 0 is never consumed, 2 and 5 consume twice. Two version of the program will use for synchronized programs. In the first one, an integer array buffer is used, instead of a single common integerBuffer size can be adjusted to secure a completely filled buffer.

18.2-3 CommonCellBuff program, synchronised common variable by using a buffer

```
import java.awt.*;
import java.applet.Applet;
import java.text.DecimalFormat;
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;

class CommonCellBuf
{
   public static void main(String args[])
   {
      CatchInteger h=new CatchInteger();
      ProduceInteger p=new ProduceInteger(h);
      Consumeinteger c= new Consumeinteger(h);
      p.start();
      c.start();
   }
}

class ProduceInteger extends Thread
{
   private CatchInteger pTut;
   private TextArea cikti;

   public ProduceInteger(CatchInteger h)
   {
      pTut=h;
   }

   public void run()
   {
      for(int count=0; count<10; count++)
      {
         pTut.setCommonInteger(count);
         //System.out.println("ProduceInteger created  "+count);

         try{
              sleep( (int) (Math.random()*500 ) );
          }
          catch(InterruptedException e)
          {
            System.err.println("Error:  "+e.toString());
         }
      }
   }

   }

class Consumeinteger extends Thread
{
   private CatchInteger cTut;

   public Consumeinteger(CatchInteger h)
   {
      cTut=h;
   }
```

```
    public void run()
    {
        int intValue;

        intValue=cTut.getCommonInteger();

        while (intValue !=9)
        {
            try{
                sleep( (int) (Math.random()*500));
               }
            catch(InterruptedException e)
            {
                System.err.println("Error "+e.toString());
            }
            intValue=cTut.getCommonInteger();
            System.out.println("ConsumeInteger caught "+intValue);

       }
    }
}

class CatchInteger
{
    private int Commonint[] = {9,9,9,9,9};
    private boolean writeable = true;
    private boolean readable = false;
    private int LocalRead =0, LocalPrint=0;
    public synchronized void setCommonInteger(int intValue)
    {
        while (!writeable)
        {
          try{
            wait();
             }
          catch (InterruptedException e)
          {
            System.err.println("Error :"+e.toString() );
          }
        }

        Commonint[LocalPrint]=intValue;
        readable = true;
        LocalPrint = ++LocalPrint % 5;
        System.out.println("Produced Integer : "+intValue);
        printBuffer(Commonint);
        if(LocalPrint==LocalRead)
        {
            writeable=false;
            System.out.println("\nSTORAGE SPACE ˜ FULL ");
        }
        //notify();
    }

    public synchronized int getCommonInteger()
    {
        int intValue;
         while (!readable)
```

```
        {
         try{
           System.out.println(" WAITING TO BE USED ");
           wait();
            }
         catch (InterruptedException e)
         {
           System.err.println("Error:"+e.toString() );
         }
        }

        writeable=true;
        intValue=Commonint[LocalRead];
        LocalRead = ++LocalRead % 5;
        if (LocalRead==LocalPrint)
        {readable=false;
        System.out.println("\nINTEGER STRORAGE EMPTY");
        }
        notify();
        return intValue;
    }

    public void printBuffer(int buf[])
    {
                String s="";
        s+="\t storage: ";
        for(int i=0; i<buf.length; i++)
          s+="  "+buf[i];
        System.out.println(s);
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" CommonCellBuf
Produced Integer : 0
            storage:   0  9  9  9  9

INTEGER STRORAGE EMPTY
Produced Integer : 1
            storage:   0  1  9  9  9

INTEGER STRORAGE EMPTY
ConsumeInteger caught 1
Produced Integer : 2
            storage:   0  1  2  9  9
Produced Integer : 3
            storage:   0  1  2  3  9
Produced Integer : 4
            storage:   0  1  2  3  4
ConsumeInteger caught 2
Produced Integer : 5
            storage:   5  1  2  3  4
ConsumeInteger caught 3
```

```
Produced Integer : 6
            storage:   5  6  2  3  4
Produced Integer : 7
            storage:   5  6  7  3  4
ConsumeInteger caught 4
Produced Integer : 8
            storage:   5  6  7  8  4
Produced Integer : 9
            storage:   5  6  7  8  9

STORAGE SPACE ˜ FULL
ConsumeInteger caught 5
ConsumeInteger caught 6
ConsumeInteger caught 7
ConsumeInteger caught 8

INTEGER STRORAGE EMPTY
ConsumeInteger caught 9

> Terminated with exit code 0.
```

In the next program , execution of the thread will be forced to wait untill the other thread is executed. It is similar to wait in a crossroad trafic signal.

18.2-4 CommonCellSyn program, synchronised common variable by using the other variable wait till the way is clear

```
class CommonCellSyn
{
   public static void main(String args[])
   {
      CreateInteger h=new CreateInteger();
      ProduceInteger p=new ProduceInteger(h);
      ConsumeInteger c= new ConsumeInteger(h);

      p.start();
      c.start();
   }
}

class ProduceInteger extends Thread
{
   private CreateInteger pTut;

   public ProduceInteger(CreateInteger h)
   {
      pTut=h;
   }

   public void run()
   {
      for(int count=0; count<10; count++)
      {
         pTut.setCommonInt(count);
         System.out.println("ProduceInteger created  "+count);

        try{
```

```
                sleep( (int) (Math.random()*3000 ) );
            }
            catch(InterruptedException e)
            {
              System.err.println("Exeption "+e.toString());
           }
        }
    }
}

class ConsumeInteger extends Thread
{
    private CreateInteger cTut;

    public ConsumeInteger(CreateInteger h)
    {
        cTut=h;
    }

    public void run()
    {
        int IntValue=0;

        while (IntValue !=9)
        {
            try{
                sleep( (int) (Math.random()*3000));
               }
            catch(InterruptedException e)
            {
                System.err.println("Exeption "+e.toString());
            }
            IntValue=cTut.getCommonInt();
            System.out.println("ConsumeInteger caught "+IntValue);
       }
    }
}

class CreateInteger
{
    private int CommonInt;
    private boolean Writable = true;


    public synchronized void setCommonInt(int IntValue)
    {
        while (!Writable)
        {
          try{
            wait();
             }
          catch (InterruptedException e)
          {
            System.err.println("Exception:"+e.toString() );
          }
        }

        CommonInt=IntValue;
```

```
        Writable=false;
        notify();

    }

    public synchronized int getCommonInt()
    {
         while (Writable)
        {
          try{
            wait();
             }
          catch (InterruptedException e)
          {
            System.err.println("Exception:"+e.toString() );
          }
        }


        Writable=true;
        notify();

        return CommonInt;
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" CommonCellSyn
ProduceInteger created  0
ProduceInteger created  1
ConsumeInteger caught 0
ConsumeInteger caught 1
ProduceInteger created  2
ConsumeInteger caught 2
ProduceInteger created  3
ConsumeInteger caught 3
ProduceInteger created  4
ConsumeInteger caught 4
ProduceInteger created  5
ConsumeInteger caught 5
ProduceInteger created  6
ConsumeInteger caught 6
ProduceInteger created  7
ConsumeInteger caught 7
ProduceInteger created  8
ConsumeInteger caught 8
ProduceInteger created  9
ConsumeInteger caught 9
> Terminated with exit code 0.
```

As a last example a pump program is given. A storage Tank of 10 m^3 is connected to 2 pumps. The first pump is filling up the tank while the second pump is emyting it. Pumps flow rates can change in time. In the simulation pump flow rates will be taken between 0 to 5

18.2-5 Pump_Tank_simulation program

```
class Pump_Tank_simulation
{

public static void main(String args[])
{
  Tank h=new Tank();
            Pump1 p1 = new Pump1(h);
            Pump2 p2 = new Pump2(h);
     p1.start();
     p2.start();
    }
}

class Pump1 extends Thread
{
    private Tank pTut;
    private double flow_rate;

    public Pump1(Tank h)
    {
        pTut=h;
    }

    public void run()
    {
        while(true)
        {
                        while((flow_rate = Math.abs((Math.random()*5))) == 0);
            pTut.addWater(flow_rate);
           try{
               sleep( (int) (Math.random()*3000 ) );
            }
            catch(InterruptedException e)
            {
              System.err.println("Exeption "+e.toString());
           }
        }
    }
}

class Pump2 extends Thread
{
    private Tank cTut;
    private double flow_rate;

    public Pump2(Tank h)
    {
        cTut=h;
    }

    public void run()
    {
        while (true)
        {
            try{
                sleep( (int) (Math.random()*3000));
               }
            catch(InterruptedException e)
```

```
            {
                System.err.println("Exeption "+e.toString());
            }
            while((flow_rate = Math.abs((Math.random()*5))) == 0);
            cTut.cikarSu(flow_rate);
        }
    }
}

class Tank
{
    private double TankWater = 0.0;
    private boolean Pump1 = true;
    private boolean Pump2 = false;
    private final double kapasite = 10.0;

    public synchronized void addWater(double amount_of_water)
    {
        while (!Pump1)
        {
          try{
            wait();
             }
          catch (InterruptedException e)
          {
            System.err.println("Exception:"+e.toString() );
          }
        }

        TankWater += amount_of_water;
                if (TankWater >= kapasite){
                    Pump1 = false;
                        System.out.println("Flow rate of pump 1 " + amount_of_water + " Storage tank is Full(10).");
                        TankWater = kapasite;
                }
                else
                        System.out.println("Flow rate of pump 1 " + amount_of_water + "water content of Storage tank is " +
TankWater);

                Pump2 = true;
        notify();
    }

    public synchronized void cikarSu(double amount_of_water)
    {
         while (!Pump2)
        {
          try{
            wait();
             }
          catch (InterruptedException e)
          {
            System.err.println("Exception:"+e.toString() );
          }
        }

    TankWater -= amount_of_water;
                if (TankWater <= 0)
```

```
    {
     Pump2=false;
                        System.out.println("Flow rate of pump 2 " + amount_of_water + " Storage tank is empty(0)");
                        TankWater = 0;
              }
              else
                        System.out.println("Flow rate of pump 2 " + amount_of_water + "water content of Storage tank is " +
TankWater);
              Pump1 = true;
    notify();
    }
}
```

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" Pump_Tank_simulation
Flow rate of pump 1 2.3467302907416965water content of Storage tank is 2.3467302907416965
Flow rate of pump 2 4.561512486251294 Storage tank is empty(0)
Flow rate of pump 1 0.5867555472676406water content of Storage tank is 0.5867555472676406
Flow rate of pump 2 2.600533616696703 Storage tank is empty(0)
Flow rate of pump 1 3.80887023404572water content of Storage tank is 3.80887023404572
Flow rate of pump 1 3.9397103967673575water content of Storage tank is 7.748580630813077
Flow rate of pump 2 0.9527778587374092water content of Storage tank is 6.7958027720756675
Flow rate of pump 1 0.6006220170762883water content of Storage tank is 7.396424789151956
Flow rate of pump 1 0.021814412492205393water content of Storage tank is 7.418239201644162
Flow rate of pump 2 2.8249175749160464water content of Storage tank is 4.593321626728115
Flow rate of pump 1 0.5102548266882029water content of Storage tank is 5.103576453416318
Flow rate of pump 1 0.5141968938471103water content of Storage tank is 5.617773347263428
Flow rate of pump 1 2.1940998083020187water content of Storage tank is 7.811873155565447
Flow rate of pump 1 2.42826921346704 Storage tank is Full(10).
Flow rate of pump 2 2.2087842517393335water content of Storage tank is 7.791215748260667
```

EXERCISE WEEK 11 2011

EXERCISE 1

Investigate clock.java. Add ınto program so that it will also write hour

EXERCISE 2

Investigate Pump_Tank_simulation program. .Change pool volume to 50 m^3 and rerun the program

EXERCISE 3

Investigate how AnalogClock program and how it is converted to a jar. Convert clock.java program to clock.jar so that it will run

HOMEWORK WEEK 11 - 2011

HOMEWORK 1

heat transfer simulation will be investigated. Outside temperature will be simulated as a sinus function with small random variations between a daily maximum and a daily minimum temperatures. Inside

temperature will be 20 C. Program will calculate heat transfer at any instance and total energy loss for the day. Heat transfer equation is $Q=10(T_{room}-T_{outside})$ Watt

Where Δt is the time period for each tread running. To check your program take $T_{max}=0$ C and $T_{min}=20$ C and calculate Daily energy requirement. A part of this program will be asked in the final exam be sure that you solve it

HOMEWORK 2 Convert problem in the Homework 1 into a graphic program with room temperature, minimum and maximum outside temperatures as input fields and Daily energy use as an output field. Convert the program into a jar.

19. COMPILING JAVA PROGRAMS FROM INSIDE ANOTHER PROGRAM (REAL-TIME PROGRAM COMPILING)

19.1 COMPILING JAVA PROGRAMS IN REAL TIME

Java compiler, javac, can be invoked from a java program. In order to start a java compiler the following code can be used.

```
        try{
    JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
    int rc = javac.run(null,null,null,fileName);}
    catch(NullPointerException e1) {System.err.println("NULL POINTER");}
```

In this statement javac is the name of JavaCompiler class object.
ToolProvider.getSystemJavaCompiler(); locates the place where java compiler is located and made it ready for compiling. **javac.run(null,null,null,fileName);** compiles the file "**fileName"** and created class file. As a first example a java program welcome.java is given, this program is compiled inside the program realtime1.java

Program 19.1-1 welcome.java example class

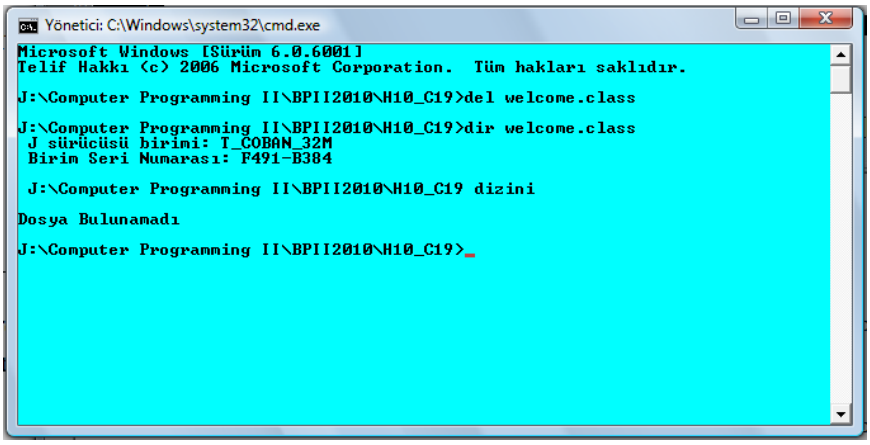
```
public class welcome
{
    // main method begins execution of Java application
    public welcome()
    {
      System.out.println( "Welcome to real time Java Programming!" );
    }
}
```

Program 19.1-2 welcome.java example class

```
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime2
{
    public static void compileandrun(String fileName)
    {
```

```
            try{
    JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
    int rc = javac.run(null,null,null,fileName);}
    catch(NullPointerException e1) {System.err.println("NULL POINTER");}
   }


   public static void main(String arg[])
   {
            //String fileName=JOptionPane.showInputDialog("Enter file name (a.txt): ");
            compileandrun("welcome.java");
            welcome x=new welcome();
            }

}
```

In order to be sure that compilation occur insde our programs, welcome.class can be deleted in console environment.



And then when program realtime2.java compiled in crimson editor(or any system you are normally compiling) and run by java command. The result is:

```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" realtime2
Welcome to real time Java Programming!

> Terminated with exit code 0.
```

As, it is clear from the result program welcome.java is first compiled and then used inside realtime2 class.

19.2 CREATING PROGRAMS AND COMPILING THEM IN REAL TIME

The programs can be written in java classes and compile and results can be used in other java programs. As an example to this following 3 classes are given. Class realtime3A will write class fa.java and compile it. This class has 2 extended class connection class f_x and class Mathd. After it is compiled it will be evaluated in class realtime3B. All this will be called from a main program in class realtime3C.

Program 19.2-1 realtime3A creating file fa.java, and compiling it.

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime3A
{
    public static void compile(String name)
    {
            try{
     JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
     int rc = javac.run(null,null,null,name);}
     catch(NullPointerException e1) {System.err.println("NULL POINTER");}
    }

 public static void createfunction(String name,String function)
 {
 String s="";
 String name1=name+".java";
 try {PrintWriter ffout=new PrintWriter(new BufferedWriter(new FileWriter(name1)));
 s+="class "+name+" extends f_x\n";
 s+=" {\n";
 s+=" public double func (double x)\n";
 s+="{ double y="+function+";\n";
 s+=" return y;\n";
 s+="}\n";
 s+="}\n";
 ffout.println(s);
 //System.out.println(s);
 ffout.close();
 }
 catch(IOException e1) {System.err.println("girdi hatası.");}
 catch(NumberFormatException e2){};
 }

    public realtime3A()
    {
            String fileName="fa";
            String function=JOptionPane.showInputDialog("Enter function statement f(x) ");
            createfunction(fileName,function);
            compile(fileName+".java");
            }
}
```

Program 19.2-2 Abstract root class for creating fa.java class

```
import static java.lang.Math.*;
```

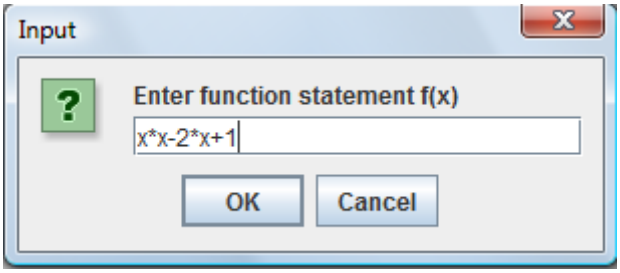
```
abstract public class f_x extends Mathd
{
  // single function single independent variable
  // example f=x*x
  abstract public double func(double x);
}
```

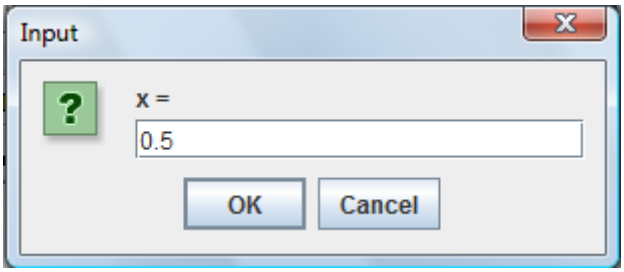
Program 19.2-3 realtime3B evaluating the value of created function fa.java

```
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime3B
{

    public realtime3B()
    {
            fa ff=new fa();
            double x=Double.parseDouble(JOptionPane.showInputDialog("x = "));
            System.out.println("x="+x+"y="+ff.func(x));
            }
}
```

Program 19.2-4 realtime3B evaluating the value of created function fa.java

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime3C
{

    public static void main(String arg[])
    {
             realtime3A x1=new realtime3A();
             realtime3B x2=new realtime3B();
            }
}
```






```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" realtime3C
x=0.5y=0.25
```

Program 19.2-5 fa.java created and compiled by realtime3A

```
class fa extends f_x
 {
 public double func (double x)
{ double y=x*x-2*x+1;
 return y;
}
}
```

In the next example the similar task will be run but this time the task fort he function is to take the derivative

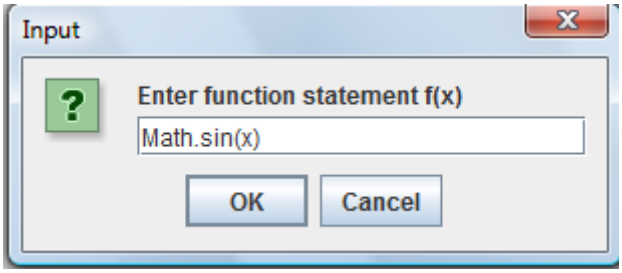
Program 19.2-6 realtime4B

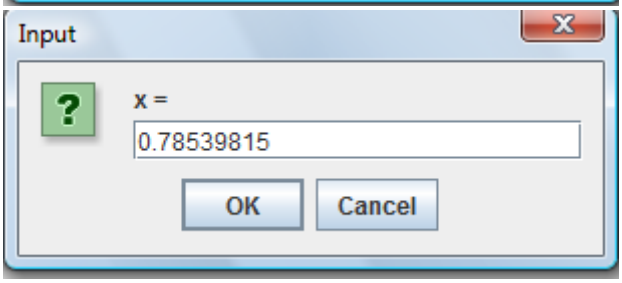
```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime4B
{
```

```
    public realtime4B()
    {
            fa ff=new fa();
            double x=Double.parseDouble(JOptionPane.showInputDialog("x = "));
            System.out.println("x="+x+"y="+ff.func(x)+"derivative = "+numeric.derivative(ff,x));
            }
}
```

Program 19.2-6 realtime4C

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

public class realtime4C
{
    public static void main(String arg[])
    {
            realtime3A x1=new realtime3A();
            realtime4B x2=new realtime4B();
            }
}
```






```
---------- Capture Output ----------
> "C:\turhan\java\bin\javaw.exe" realtime4C
x=0.78539815y=0.707106771713121derivative = 0.7071067906501878
> Terminated with exit code 0.
```

As the next example function plotting with real time input will be shown. It is very similar to the previous example. FunctionGenerator write the function and compile it. Girdi2 is a graphical interphase for function writing window. It is a real time program that will wait until the button pushed. Ce1 is the main class calling these classes. Class ce3 i prepared for plotting the written function.

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class ce1
{               String s;
                public  ce1()
                {girdi2 g=new girdi2("f2D"," ");
                 s=g.vericiktisi();
                 try{ g.finalize();}
                 catch(Throwable e1) {System.err.println("input error.");}
                }

                public  ce1(String function)
                {               functiongenerator g=new functiongenerator("f2D",function);
                   s=g.createfunction();
                   try{ g.finalize();}
                   catch(Throwable e1) {System.err.println("input error.");}
                }
    public static double[] enterdoublearray(String s)
    {
    String s1=JOptionPane.showInputDialog(s);
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }

                public static void main(String arg[])
                {
               ce1 x0=new ce1();
               double a[]=enterdoublearray("Enter minimum and maximum values of the function xmin xmax");
```

```
            ce3 x1=new ce3(a[0],a[1],400,x0.s);
            }
}
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
import java.io.*;
import java.util.*;


public class girdi2 extends Thread implements ActionListener
{
            Container c;
            String s=" ";
            String s1="";
            functiongenerator fg;
            JTextArea jt;
            JButton bb;
            boolean basildi;

            public girdi2(String isim,String fonksiyon)
            {
            jt=new JTextArea(s,5,30);
            jt.setFont(new Font("SansSerif",Font.PLAIN,20));
            fg=new functiongenerator();
            bb=new JButton("push button after entering the function into to the window : x");
            JScrollPane jsp=new JScrollPane(jt);
    JFrame cerceve=new JFrame("function input window example: Math.sin(x)-1/(x*x-1) ");
    cerceve.addWindowListener(new BasicWindowMonitor());
    c=cerceve.getContentPane();
    c.setLayout(new BorderLayout());
    c.add(jsp,BorderLayout.NORTH);
    c.add(bb,BorderLayout.SOUTH);
    bb.addActionListener(this);
    cerceve.pack();
    cerceve.setVisible(true);
    basildi=false;
    }

    public String vericiktisi()
    {
      while(!basildi)
      {
      try {Thread.sleep(100);}
      catch(InterruptedException e)  {System.err.println(e.toString());}
      }
              return s1;
     }

    protected void finalize() throws Throwable
    {super.finalize();}

public void actionPerformed( ActionEvent e)
{
  if(e.getSource()==bb)
```

```
  {s=jt.getText();fg.addfunction(s);s1=fg.createfunction();
  basildi=true;
  try{fg.finalize();finalize();
    } catch(Throwable e1) {System.err.println("girdi hatası.");}
  }
}
}
```

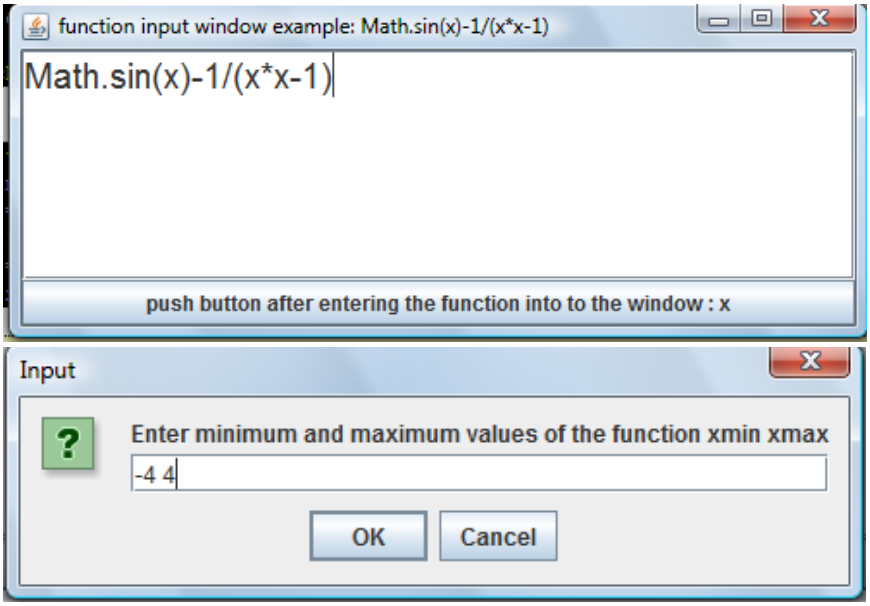
```
import java.util.*;
import java.lang.Math.*;
import java.lang.*;

class functiongenerator
 {
  String fonksiyon;
  String dosyaismi;
  String sinif;
  PrintStream fout;
  PrintStream ferr;

  public functiongenerator(String dosyaismi,String fonksiyon)
  { this.fonksiyon=fonksiyon;
    this.dosyaismi=dosyaismi;
    try {
    fout=new PrintStream(new FileOutputStream("print.txt"));
    ferr=new PrintStream(new FileOutputStream("error.txt"));
    }catch(FileNotFoundException e1) {System.err.println("dosya bulunamadı");}
  }

  public functiongenerator(String fonksiyon)
  {this.fonksiyon=fonksiyon;
   dosyaismi="f2D";
  }

  public void addfunction(String fonksiyon)
  {this.fonksiyon=fonksiyon;}

  public String toString()
  {return fonksiyon;}

  public functiongenerator()
  {fonksiyon="x*x-4.0";
   dosyaismi="f2D";
  }

 public  String createfunction()
 {String isim=dosyaismi+".java";
 String s="";
 try {PrintWriter ffout=new PrintWriter(new BufferedWriter(new FileWriter(isim)));
 s+="class "+dosyaismi+" extends f_x\n";
 s+=" {\n";
 s+=" public double func (double x)\n";
 s+="{ double y="+fonksiyon+";\n";
 s+=" return y;\n";
 s+="}\n";
 s+="}\n";
 sinif=s;
 ffout.println(s);
```

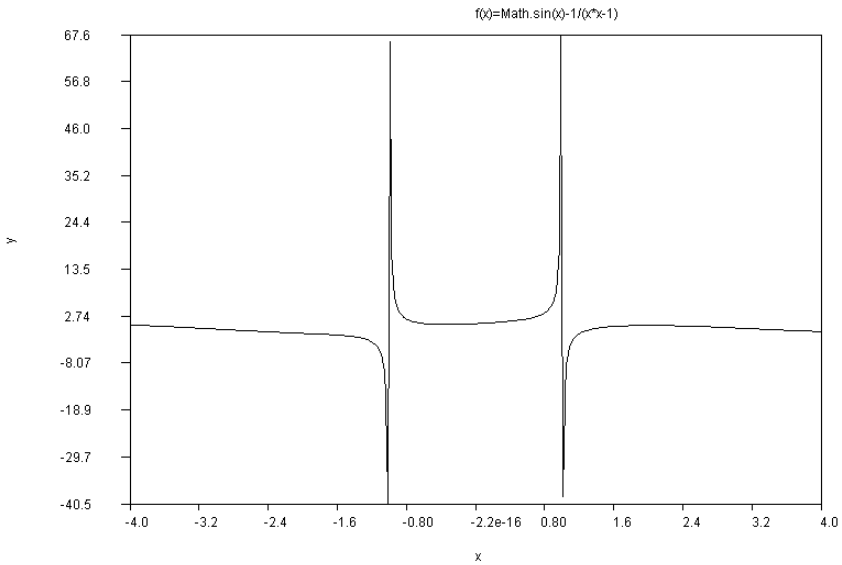
```
 //System.out.println(s);
 ffout.close();
 }
 catch(IOException e1) {System.err.println("girdi hatası.");}
 catch(NumberFormatException e2){};

 try{
 JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
 int rc = javac.run(null,null,null,isim);}
 catch(NullPointerException e1) {System.err.println("NULL POINTER");}
 return fonksiyon;
}

 protected void finalize() throws Throwable
 {super.finalize();}
 }
```

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class ce3 extends JApplet
{
            public ce3(double x1,double x2,int n,String s)
            {
            Plot pp=new Plot(new f2D(),x1,x2,n);
             pp.setPlabel("f(x)="+s);
             pp.plot();
            }

            public ce3(double x1,double x2,int n)
            {Plot pp=new Plot(new f2D(),x1,x2,n);
             pp.plot();
            }

            public ce3(double x1,double x2)
            {Plot pp=new Plot(new f2D(),x1,x2,50);
             pp.plot();
            }

             public static void main(String arg[])
             {double a=Double.parseDouble(JOptionPane.showInputDialog(" Fonksiyonun alt sınırını giriniz x_minimum  :
"));
      double b=Double.parseDouble(JOptionPane.showInputDialog(" Fonksiyonun üst sınırını giriniz x_maksimum : "));
              ce3 xx=new ce3(a,b,400);}
```

```
}
```

The same





One additional note: it can be used as an application program. In order to do that one can write a bat file (or equivalent in unix) to contain statement such as

plot.bat

```
java  ce1
```

when this bat file run in yor computer it will plot for you the function you write into the screen

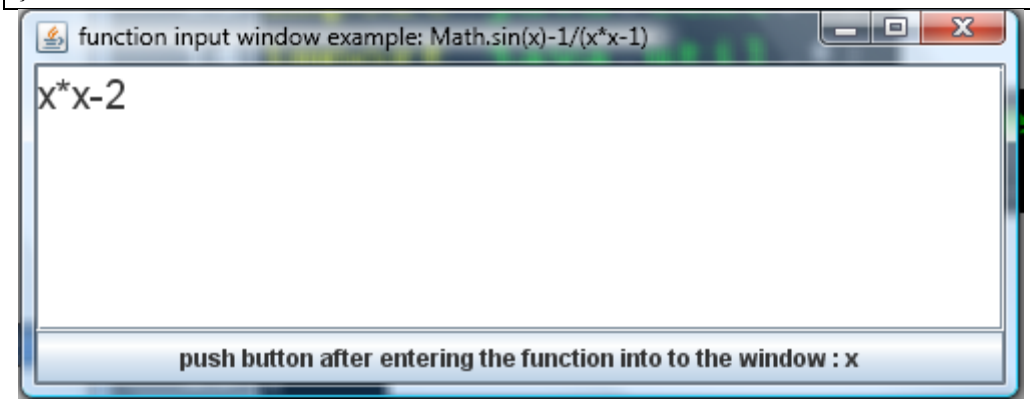
This concept can be used for any function writing required programming. For example in the following programs root of a function is to be found by using bisection method

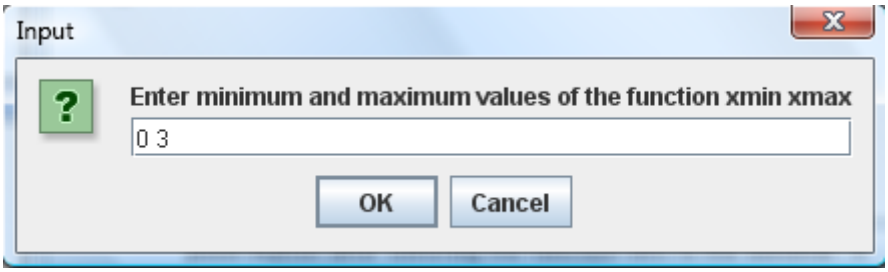
```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
```

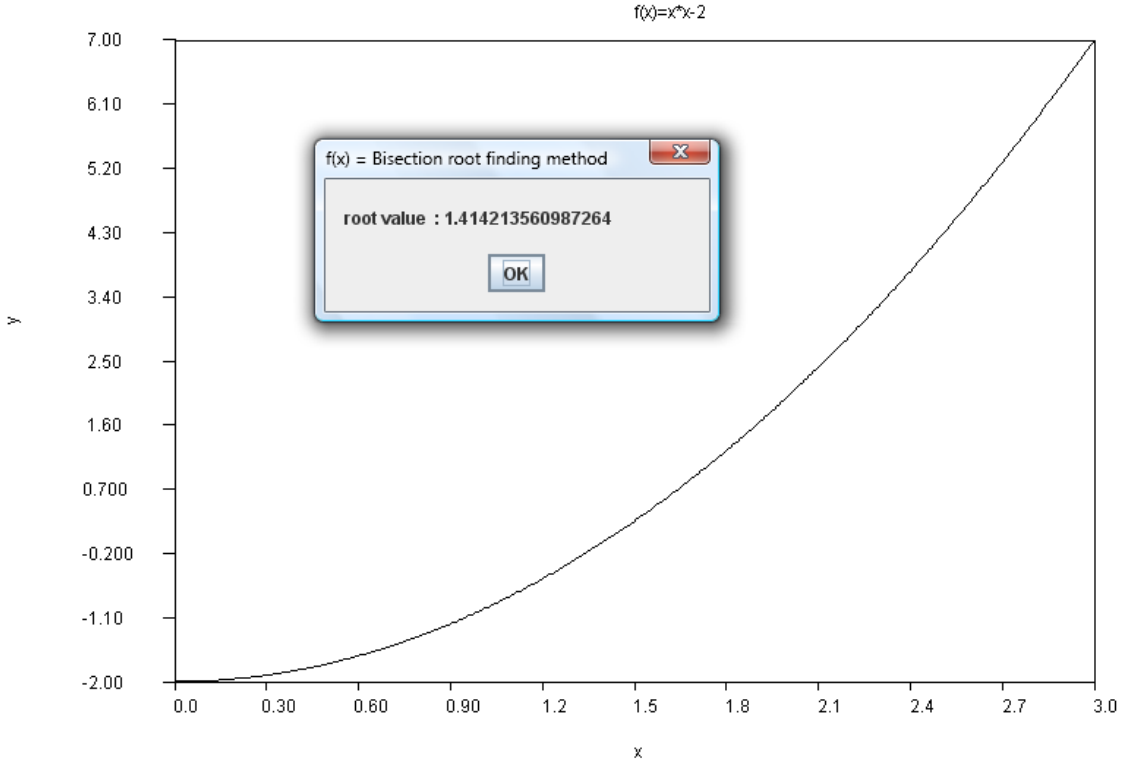
```
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;

class ce2
{
                public ce2(double x1,double x2,String s)
                {
        double a[]={x1,x2};
        double r;
        f2D ff=new f2D();
        r= bisection(ff,a[0],a[1]);
        JOptionPane.showMessageDialog(null," root value  : "+r,
        "f(x) = "+s,JOptionPane.PLAIN_MESSAGE);
    }

public static double bisection(f_x f,double a,double b)
{
//bisection root finding method
double test;
double p=0;
double es,ea;
double fa,fp;
int maxit=500,iter=0;
es=0.0000001;
ea=1.1*es;
while((ea>es)&&(iter<maxit))
{
   p=(a+b)/2.0;
   iter++;
   if((a+b)!=0)
       { ea=Math.abs((b-a)/(a+b))*100;}
   fa= f.func(a);
   fp= f.func(p);
   test= fa*fp;
   if(test==0.0)      ea=0;
   else if(test<0.0)  b=p;
   else               {a=p;}
}
if(iter>=maxit) JOptionPane.showMessageDialog(null,"Maximum number of iteration is exceded \n"+
   " result might not be valid ","MAXIMUM ITERATION WARNING",JOptionPane.WARNING_MESSAGE);
return p;
}
}
```

```
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
```

```
import javax.swing.table.*;
import static java.lang.Math.*;

class ce4
{
  public static void main(String arg[])
  {
    ce1 x0=new ce1();
    double a[]=ce1.enterdoublearray("Enter minimum and maximum values of the function xmin xmax");
    ce3 x1=new ce3(a[0],a[1],400,x0.s);
    ce2 x2=new ce2(a[0],a[1],"Bisection root finding method");
   }
}
```







In the next example an html file will be generated by using

EXERCISE 19(WEEK 10)

EXERCISE 1

Following programs are given. To integrate a function

```
//abstract sinif ve boyutlu değiskenler
abstract class f_x
{ abstract double func(double x);}

public class numeric
{
public static double derivative(f_x f,double x)
{  double h=0.00001;
  return (-f.func(x+2.0*h)+8.0*f.func(x+h)-8.0*f.func(x-h)+f.func(x-2.0*h))/(12.0*h);
}
public static double integral(f_x f_xnt,double a,double b)
{
//integral f(x)dx
 double r[]={-0.973906528517171,-0.865063366688984,-0.679409568299024,-0.433395394129247,-0.148874338981631,
 0.148874338981631,0.433395394129247,0.679409568299024,0.865063366688984,0.973906528517171};
 double c[]={0.066671344308684,0.149451349150580,0.219086362515982,0.269266719309996,0.295524224714752,
 0.295524224714752,0.269266719309996,0.219086362515982,0.149451349150580,0.066671344308684};
double z=0,x,y;
double k1=(b-a)/2.0;
double k2=(b+a)/2.0;
for(int i=0;i<r.length;i++)
{
x=k2+k1*r[i];
y=f_xnt.func(x);
```

```
z+=k1*c[i]*y;
}
return z;
}
}
```

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;

class integ1
{
   public integ1(double x1,double x2,String s)
   {
      double a[]={x1,x2};
      double r;
      f2D ff=new f2D();
      r= numeric.integral(ff,a[0],a[1]);
      JOptionPane.showMessageDialog(null," Gauss-Legendre Integral  : "+r,
      "f(x) = "+s,JOptionPane.PLAIN_MESSAGE);
    }
 }
```

Write a program, integ.java, similar to ce4.java to take integral of a function defined by the user.

EXERCISE 2

Write a program, deriv1.java, similar to integ1.java . it will use the method

public static double derivative(f_x f,double x)

in numeric.java to take derivative of a function defined by the user. Then call it with program deric.java to take derivative of a function similar to ce4.java

HOMEWORK 19(WEEK 10)

HOMEWORK 1

Following programs are created to calculate a differential equation in real time. Investigate and run the program. As it is, program uses 6th order Runge_kutta method to solve differential eqıuation.

public static double[][] RK6(f_xy f,double x0,double y0,double xn,double h)

change it to Runge-Kutta-Fehlberg method also given in the program

public static double[][] RKF6(f_xy f,double x0,double y0,double xn,double h,double eps)

solve differential equation

$$\frac{dy}{dx} = 0.5y + 1e^{(x^2/10)^2}$$

x=0 y(0)=0.5 from x=0 to 4 start with h=0.1 eps=0.001

Functiongenerator1.java

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Math.*;
import java.lang.*;

abstract class f_xy
{
  //single function single independent variable
  // example f=x*x
  abstract double func(double x,double y);
}


class functiongenerator1
 {
  String fonksiyon;
  String dosyaismi;
  String sinif;
  PrintStream fout;
  PrintStream ferr;

  public functiongenerator1(String dosyaismi,String fonksiyon)
  { this.fonksiyon=fonksiyon;
    this.dosyaismi=dosyaismi;
    try {
    fout=new PrintStream(new FileOutputStream("print.txt"));
    ferr=new PrintStream(new FileOutputStream("error.txt"));
    }catch(FileNotFoundException e1) {System.err.println("file is not found");}
  }

  public functiongenerator1(String fonksiyon)
  {this.fonksiyon=fonksiyon;
   dosyaismi="fDEq";
  }

  public void addfunction(String fonksiyon)
  {this.fonksiyon=fonksiyon;}
```

```
 public String toString()
 {return fonksiyon;}

 public functiongenerator1()
 {fonksiyon="x*x*y";
  dosyaismi="fDEq";
 }

public  String createfunction()
{String isim=dosyaismi+".java";
String s="";
try {PrintWriter ffout=new PrintWriter(new BufferedWriter(new FileWriter(isim)));
s+="class "+dosyaismi+" extends f_xy\n";
s+=" {\n";
s+=" public double func (double x,double y)\n";
s+="{ double z="+fonksiyon+";\n";
s+=" return z;\n";
s+="}\n";
s+="}\n";
sinif=s;
ffout.println(s);
System.out.println(s);
ffout.close();
}
catch(IOException e1) {System.err.println("input error.");}
catch(NumberFormatException e2){};
//compiling it
try{
JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
int rc = javac.run(null,null,null,isim);}
catch(NullPointerException e1) {System.err.println("NULL POINTER");}
return fonksiyon;
}

protected void finalize() throws Throwable
{super.finalize();}
}
```

inputFDE.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
import java.io.*;
import java.util.*;
public class inputFDE extends Thread implements ActionListener
```

```
{
            Container c;
            String s=" ";
            String s1="";
            functiongenerator1 fg;
            JTextArea jt;
            JButton bb;
            boolean basildi;

            public inputFDE(String isim,String fonksiyon)
            {
            jt=new JTextArea(s,5,30);
            jt.setFont(new Font("SansSerif",Font.PLAIN,20));
            fg=new functiongenerator1();
            bb=new JButton("push button after entering the function into to the window : x");
            JScrollPane jsp=new JScrollPane(jt);
    JFrame cerceve=new JFrame("function input window example:x*x*y-y");
    cerceve.addWindowListener(new BasicWindowMonitor());
    c=cerceve.getContentPane();
    c.setLayout(new BorderLayout());
    c.add(jsp,BorderLayout.NORTH);
    c.add(bb,BorderLayout.SOUTH);
    bb.addActionListener(this);
    cerceve.pack();
    cerceve.setVisible(true);
    basildi=false;
    }

    public String vericiktisi()
    {
      while(!basildi)
      {
      try {Thread.sleep(100);}
      catch(InterruptedException e)  {System.err.println(e.toString());}
      }
              return s1;
            }

    protected void finalize() throws Throwable
    {super.finalize();}

public void actionPerformed( ActionEvent e)
{
  if(e.getSource()==bb)
  {s=jt.getText();fg.addfunction(s);s1=fg.createfunction();
  basildi=true;
  try{fg.finalize();finalize();
    } catch(Throwable e1) {System.err.println("girdi hatası.");}
```

```
  }
}
}
```

de1.java

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class de1
{           String s;
            public  de1()
            { inputFDE g=new inputFDE("fDEq","  ");
              s=g.vericiktisi();
             try{ g.finalize();}
             catch(Throwable e1) {System.err.println("input error.");}
            }

    public static double[] enterdoublearray(String s)
    {
    String s1=JOptionPane.showInputDialog(s);
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }
```

```
            public static void main(String arg[])
            {
                    de1 x0=new de1();
      double a[]=enterdoublearray("Enter minimum and maximum values of the function xmin
xmax");
      ce3 x1=new ce3(a[0],a[1],400,x0.s);
            }
}
```

differential1.java

```
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;

class differential1
{
            public differential1(double x0,double y0,double xn,double h)
            {
      fDEq ff=new fDEq();
      double r[][]= RK6(ff,x0,y0,xn,h);
      Plot pp=new Plot(r[0],r[1]);
              pp.plot();
      Text.print(Text.T(r));
    }

    public static double[] enterdoublearray(String s)
    {
    String s1=JOptionPane.showInputDialog(s);
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }
```

```
 public static double[][] RK6(f_xy f,double x0,double y0,double xn,double h)
 {
 // Runge-Kutta 6
 int n=(int)((xn-x0)/h);
 double a[][]=new double[2][n+1];
 int i=0;
 a[0][0]=x0;
 a[1][0]=y0;
 double k1,k2,k3,k4,k5,k6;
 for(double xx=x0;xx<xn;xx+=h)
  {a[0][i+1]=xx+h;
  k1=f.func(a[0][i],a[1][i]);
  k2=f.func(a[0][i]+0.25*h,a[1][i]+0.25*k1*h);
  k3=f.func(a[0][i]+0.25*h,a[1][i]+1.0/8.0*k1*h+1.0/8.0*k2*h);
  k4=f.func(a[0][i]+0.5*h,a[1][i]-0.5*k2*h+k3*h);
  k5=f.func(a[0][i]+3.0/4.0*h,a[1][i]+3.0/16.0*k1*h+9.0/16.0*k4*h);
  k6=f.func(a[0][i]+h,a[1][i]-3.0/7.0*k1*h+2.0/7.0*k2*h+12.0/7.0*k3*h-12.0/7.0*k4*h+8.0/7.0*k5*h);
  a[1][i+1]=a[1][i]+(7.0*k1+32.0*k3+12.0*k4+32.0*k5+7*k6)/90.0*h;
  i++;
  }
 return a;
 }
public static double[][] RKF6(f_xy f,double x0,double y0,double xn,double h,double eps)
 {
 // Runge-Kutta Fehlberg 6 değişken stepli
 //eps bağıl hata
 double alfa;
 //double a[][]=new double[2][n+1];
 int i=0;
 //a[0][0]=x0;
 //a[1][0]=y0;
 double delta=h;
 double deltayeni=eps*y0;
 double k1,k2,k3,k4,k5,k6;
 Vector<Double> xi=new Vector<Double>();
 Vector<Double> yi=new Vector<Double>();
 double x,y,yp1_5,yp1_6,xeski,yeski;
 x=x0;
 y=y0;
 xeski=x;
 yeski=y;
 Double Xi=new Double(x);
 Double Yi=new Double(y);
 xi.add(Xi);
 yi.add(Yi);
 while((x+h)<=xn)
  {
  xeski=x;
```

```
yeski=y;
k1=f.func(x,y);
k2=f.func(x+1.0/5.0*h,y+1.0/5.0*k1*h);
k3=f.func(x+3.0/10.0*h,y+3.0/40.0*k1*h+9.0/40.0*k2*h);
k4=f.func(x+3.0/5.0*h,y+3.0/10.0*k1*h-9.0/10.0*k2*h+6.0/5.0*k3*h);
k5=f.func(x+h,y-11.0/54.0*k1*h+5.0/2.0*k2*h-70.0/27.0*k3*h+35/27.0*k4*h);
k6=f.func(x+7.0/8.0*h,y+1631.0/55296.0*k1*h+175.0/512.0*k2*h+575.0/13824.0*k3*h+44275.0/110592.0*k4*h+25
yp1_5=y+(37.0/378.0*k1+250.0/621.0*k3+125.0/594.0*k4+512.0/1771.0*k6)*h;
yp1_6=y+(2825.0/27648.0*k1+18575.0/48384.0*k3+13525.0/55296.0*k4+277.0/14336.0*k5+1.0/4.0*k6)*h;
y=yp1_6;
x=x+h;
Xi=new Double(x);
Yi=new Double(y);
xi.add(Xi);
yi.add(Yi);
delta=Math.abs(yp1_6-yp1_5);
deltayeni=Math.abs(eps*yp1_6);
if(delta<=deltayeni) alfa=0.2;
else                 alfa=0.25;
h=h*Math.pow((deltayeni/delta),alfa);
i++;
}
if(x<xn)//son bir step daha
{
h=xn-x;
k1=f.func(x,y);
k2=f.func(x+1.0/5.0*h,y+1.0/5.0*k1*h);
k3=f.func(x+3.0/10.0*h,y+3.0/40.0*k1*h+9.0/40.0*k2*h);
k4=f.func(x+3.0/5.0*h,y+3.0/10.0*k1*h-9.0/10.0*k2*h+6.0/5.0*k3*h);
k5=f.func(x+h,y-11.0/54.0*k1*h+5.0/2.0*k2*h-70.0/27.0*k3*h+35/27.0*k4*h);

k6=f.func(x+7.0/8.0*h,y+1631.0/55296.0*k1*h+175.0/512.0*k2*h+575.0/13824.0*k3*h+44275.0/110592.0*k4*h+25
yp1_5=y+(37.0/378.0*k1+250.0/621.0*k3+125.0/594.0*k4+512.0/1771.0*k6)*h;
yp1_6=y+(2825.0/27648.0*k1+18575.0/48384.0*k3+13525.0/55296.0*k4+277.0/14336.0*k5+1.0/4.0*k6)*h;
y=yp1_6;
//System.out.println("x="+x+"h="+h+"y="+yp1_6);
x=xn;
Xi=new Double(x);
Yi=new Double(y);
xi.add(Xi);
yi.add(Yi);
i++;
}
//vektöre aktarılan değerleri double matrise aktar
int n=i;
    double aa[][]=new double[2][n+1];
    Enumeration nx=xi.elements();
    Enumeration ny=yi.elements();
```

```
        for(i=0;i<=n;i++)
        {
                //String ss1=(String)nx.nextElement();
                //String ss2=(String)ny.nextElement();
                Double ax=(Double)nx.nextElement();
                Double ay=(Double)ny.nextElement();
        aa[0][i]=ax.doubleValue();
        aa[1][i]=ay.doubleValue();
        }
   return aa;
  }
 }
```

differential.java

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class differential
{
  public static void main(String arg[])
  {
    de1 dd=new de1();
            double a[]=de1.enterdoublearray("enter initial_x,initial_y,final_x,step_size_h ");
    differential1 df1=new differential1(a[0],a[1],a[2],a[3]);
   }
}
```

HOMEWORK 2

The following chronometer class is defined. Pack the class into jar file so that it can work from your computer screen directly.

```
import java.awt.*;
 import java.awt.event.*;
 import java.awt.geom.*;
 import javax.swing.*;
```

```
import java.net.*;
import java.io.*;
import java.awt.Graphics;
import java.util.*;
import java.text.*;
import java.util.Locale;

class chronometerThread extends Thread
{
public long i;
JFrame a;
public boolean done;
public chronometerThread(JFrame ai)
{a=ai;i=0;done=true;}

public void reset() {i=0;}


public void run()
{
while(done)
{
a.validate();
a.repaint();
try {Thread.sleep(10);i++;}
catch(InterruptedException e)
{System.err.println(e.toString());}
}
}
}

public class chronometer extends JFrame implements Serializable,ActionListener
{
private static final long serialVersionUID=358044808532797990L;
            JTextField jb;
JButton b;
 Font F;
chronometerThread r;
int minute;
int  second;
int  centisecond;

public chronometer()
{
super("chronometer");
 second=0;
Container cc=getContentPane();
jb=new JTextField("");
b=new JButton("start/stop");
F=new Font("TimesRoman",Font.BOLD,24);
jb.setFont(F);
b.setFont(F);
r=new chronometerThread(this);
cc.add(jb,BorderLayout.NORTH);
cc.add(b,BorderLayout.SOUTH);
b.addActionListener(this);
}
```

```
public void actionPerformed( ActionEvent e)
{
if(r.i==0) r.start();
else {r.done=false;r=new chronometerThread(this);}
}
public void paint(Graphics g)
{    minute=(int)(r.i/6000);
     second=(int)(r.i/100%60);
     centisecond=(int)(r.i%100);
     jb.setText(""+minute+" "+second+"  "+centisecond);
}
   public static void main(String s[])
   {  chronometer f = new chronometer();
      f.addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent e) {System.exit(0);}
      });
      f.setSize(200,120);
      f.setVisible(true);
   }
}
```

20.0 NETWORKING

20.1 UNIFORM RESOURCE LOCATERS (URL)

Class URL represents a Uniform Resource Locator, a pointer to a "resource" on the World Wide Web. A resource can be something as simple as a file or a directory, or it can be a reference to a more complicated object, such as a query to a database or to a search engine. URL uses TCP/IP protocol to send and receive the data. TCP short for Transmission interface protocol, and IP is short for internet protocol.

IP caries the code with the following sub-parts

0-10 : heading

10-12 : control

12-16 sender adress

16-20 arrival adress

20-24 : oprions

24....data

There are no guarantee that IP packages will arrive to the destination. IP packages are similar to telegram packages (one direction). TCP packagges are like telephone services, connect both sides. Details of a TCP package

0-2 senders adress

2-4 arrival adress

4-8 package number

8-12 package arrival number

12-20 headings

20+ data

Some oft he TCP channels is utilised for specific purposes. For example channel 7 reflect back the send data, Channel 13 is the time of local machine that data is arrived. Channel 20-21 is ftp protocol. Channel 23 telnet protocol and channel 80 is internet protocol.

Our first application class is URL class. By using URL class, actual internet adresses can be reached and data can be accessed throgh this channel. Using URL is similar to other data channel usages. Detail oft he URL class methods are given below as a table.

Constructor Summary
URL(String spec) Creates a URL object from the String representation.
URL(String protocol, String host, int port, String file) Creates a URL object from the specified protocol, host, port number, and file.
URL(String protocol, String host, int port, String file, URLStreamHandler handler) Creates a URL object from the specified protocol, host, port number, file, and handler.
URL(String protocol, String host, String file)

	Creates a URL from the specified protocol name, host name, and file name.
URL(URL context, String spec) Creates a URL by parsing the given spec within a specified context.	
URL(URL context, String spec, URLStreamHandler handler) Creates a URL by parsing the given spec with the specified handler within a specified context.	

Method Summary	
boolean	equals(Object obj) Compares this URL for equality with another object.
String	getAuthority() Gets the authority part of this URL.
Object	getContent() Gets the contents of this URL.
Object	getContent(Class[] classes) Gets the contents of this URL.
int	getDefaultPort() Gets the default port number of the protocol associated with this URL.
String	getFile() Gets the file name of this URL.
String	getHost() Gets the host name of this URL, if applicable.
String	getPath() Gets the path part of this URL.
int	getPort() Gets the port number of this URL.
String	getProtocol() Gets the protocol name of this URL.
String	getQuery() Gets the query part of this URL.
String	getRef() Gets the anchor (also known as the "reference") of this URL.
String	getUserInfo() Gets the userInfo part of this URL.
int	hashCode() Creates an integer suitable for hash table indexing.
URLConnection	openConnection() Returns a URLConnection object that represents a connection to the remote object referred to by the URL.
URLConnection	openConnection(Proxy proxy) Same as openConnection(), except that the connection will be made through the specified proxy; Protocol handlers that do not support proxing will ignore the proxy parameter and make a normal connection.
InputStream	openStream() Opens a connection to this URL and returns an InputStream for reading from that connection.
boolean	sameFile(URL other) Compares two URLs, excluding the fragment component.
protected void	set(String protocol, String host, int port, String file, String ref) Sets the fields of the URL.
protected void	set(String protocol, String host, int port, String authority, String userInfo, String path, String query, String ref) Sets the specified 8 fields of the URL.
static void	setURLStreamHandlerFactory(URLStreamHandlerFactory fac) Sets an application's URLStreamHandlerFactory.
String	toExternalForm() Constructs a string representation of this URL.

String	toString() Constructs a string representation of this URL.
URI	toURI() Returns a URI equivalent to this URL.

Methods inherited from class java.lang.Object
clone, finalize, getClass, notify, notifyAll, wait, wait, wait

Our first example will select an internet site from the given list of the sites. Program is given as an applet in order to jump directly to selected adres in http(internet) protocol.

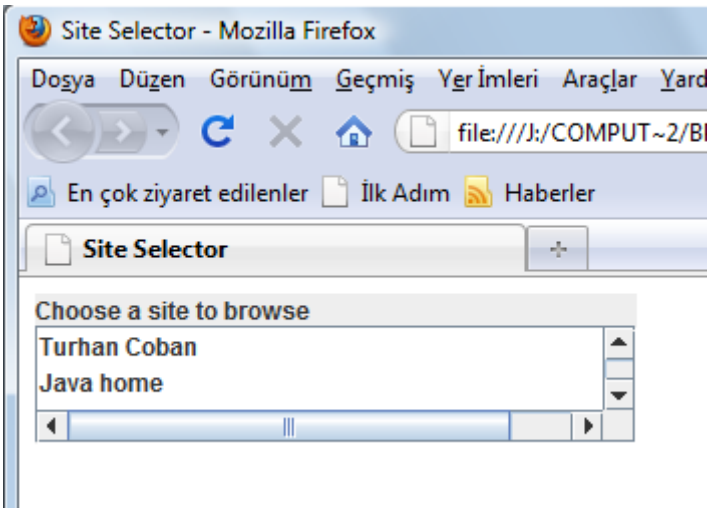
Program 20.1.1 SiteSelector.java

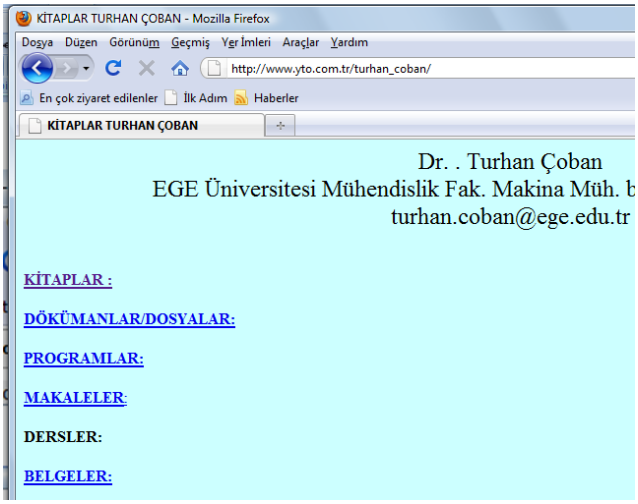
```
import    java.net.MalformedURLException;

import    java.net.URL;
import    java.util.HashMap;
import    java.util.ArrayList;
import    java.awt.BorderLayout;
import    java.applet.AppletContext;

import    javax.swing.JApplet;

import    javax.swing.JLabel;

import    javax.swing.JList;
import    javax.swing.JScrollPane;

import    javax.swing.event.ListSelectionEvent;

import    javax.swing.event.ListSelectionListener;

import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;

public    class    SiteSelector    extends    JApplet    implements ListSelectionListener

{
private    HashMap<Object,  URL    >    sites;    //    site    names    and    URLs
private    ArrayList<    String    >    siteNames;    //    site    names

private    JList    siteChooser;    //    list    of    sites    to    choose    from
private String name[]={"Turhan Coban",
                    "Java home",
                    "Ege University",
                    "E.U. School of Engineering, Mechanical engineering dept."};
private String adress[]={"http://www.yto.com.tr/turhan_coban/",
                     "http://java.sun.com/",
```

```
                    "http://www.ege.edu.tr/",
                    "http://me.ege.edu.tr/"};

public void init()
{
// create GUI components and layout interface
add( new JLabel( "Choose a site to browse" ), BorderLayout.NORTH );
siteChooser = new JList(name);
siteChooser.setVisibleRowCount(5);
siteChooser.addListSelectionListener(this);
add(new JScrollPane(siteChooser ), BorderLayout.CENTER );
}

public void valueChanged( ListSelectionEvent event )

{
// use site name to locate corresponding URL
String site=adress[siteChooser.getSelectedIndex()];
try{
URL newDocument = new URL(site);
System.out.println("www="+newDocument.toString());
AppletContext browser = getAppletContext();
browser.showDocument(newDocument);
 }catch ( MalformedURLException urlException )
{urlException.printStackTrace();} // end catch

} // end method valueChanged

} // end class SiteSelector
```

Program 20.1.2 SiteSelector.html

```
<html>
 <title>Site Selector</title>
 <body>
   <applet code = "SiteSelector.class" width = "300" height = "75">
   </applet>
 </body>
 </html>
```

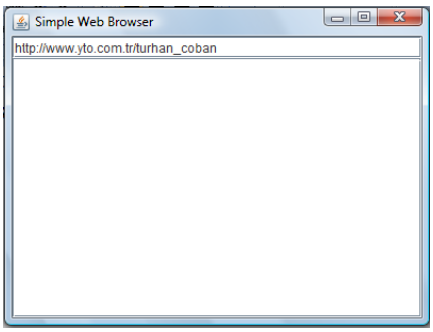


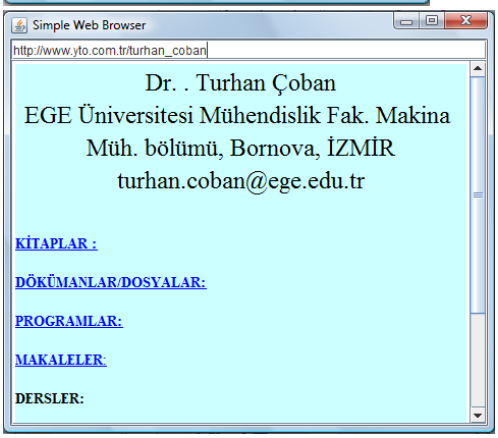


Second application is the same as the first one, but the adres requires is written into a TextField area. So it behaves as a very simple browser program

```
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import javax.swing.JEditorPane;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTextField;
import javax.swing.event.HyperlinkEvent;
import javax.swing.event.HyperlinkListener;

public class web extends JFrame implements ActionListener,HyperlinkListener
{
  private JTextField enterField; // JTextField to enter site name
  private JEditorPane contentsArea; // to display website

  // set up GUI
  public web()
  {
    super( "Simple Web Browser" );

    // create enterField and register its listener
    enterField = new JTextField( "Enter file URL here" );
    enterField.addActionListener(this);
    add( enterField, BorderLayout.NORTH );

    contentsArea = new JEditorPane(); // create contentsArea
    contentsArea.setEditable( false );
    contentsArea.addHyperlinkListener(this);
    add( new JScrollPane( contentsArea ), BorderLayout.CENTER );
    setSize( 400, 300 ); // set size of window
    setVisible( true ); // show window
  } // end web constructor

  public void actionPerformed( ActionEvent event )
        {
          getThePage( event.getActionCommand() );
        } // end method actionPerformed
```

```
    public void hyperlinkUpdate( HyperlinkEvent event )
        {
          if ( event.getEventType() ==
              HyperlinkEvent.EventType.ACTIVATED )
            getThePage( event.getURL().toString() );
        } // end method hyperlinkUpdate

    // load document
    private void getThePage( String location )
    {
      try // load document and display location
      {
        contentsArea.setPage( location ); // set the page
        enterField.setText( location ); // set the text
      } // end try
      catch ( IOException ioException )
      {
        JOptionPane.showMessageDialog( this,
          "Error retrieving specified URL", "Bad URL",
          JOptionPane.ERROR_MESSAGE );
      } // end catch
    } // end method getThePage
    public static void main( String args[] )
    {
      web application = new web();
      application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    } // end main
  } // end class web
```





A HyperlinkListener is used as a listener class for updating URL in the window. A method called public void hyperlinkUpdate(HyperlinkEvent event) is defined to take the action when a new page is requested.

20.2 SOCKET PROGRAMMING (TCP/IP)

In this section communication through TCP/IP protocol using java classes ServerSocket and ClientSocket. A server socket waits for requests to come in over the network. It performs some operation based on that request, and then possibly returns a result to the requester.

ServerSocket server = new ServerSocket(portNumber, queueLength);

Complete definition of the ServerSocket class is given below.

Constructor Summary
ServerSocket() Creates an unbound server socket.
ServerSocket(int port) Creates a server socket, bound to the specified port.
ServerSocket(int port, int backlog) Creates a server socket and binds it to the specified local port number, with the specified backlog.
ServerSocket(int port, int backlog, InetAddress bindAddr) Create a server with the specified port, listen backlog, and local IP address to bind to.

Method Summary	
Socket	accept() Listens for a connection to be made to this socket and accepts it.
void	bind(SocketAddress endpoint) Binds the ServerSocket to a specific address (IP address and port number).
void	bind(SocketAddress endpoint, int backlog) Binds the ServerSocket to a specific address (IP address and port number).
void	close() Closes this socket.
ServerSocketChannel	getChannel() Returns the unique ServerSocketChannel object associated with this socket, if any.
InetAddress	getInetAddress() Returns the local address of this server socket.
int	getLocalPort() Returns the port on which this socket is listening.
SocketAddress	getLocalSocketAddress() Returns the address of the endpoint this socket is bound to, or null if it is not bound yet.
int	getReceiveBufferSize() Gets the value of the SO_RCVBUF option for this ServerSocket, that is the proposed buffer size that will be used for Sockets accepted from this ServerSocket.
boolean	getReuseAddress() Tests if SO_REUSEADDR is enabled.
int	getSoTimeout() Retrive setting for SO_TIMEOUT.
protected void	implAccept(Socket s) Subclasses of ServerSocket use this method to override accept() to return their own subclass of socket.
boolean	isBound()

	Returns the binding state of the ServerSocket.
boolean	isClosed() Returns the closed state of the ServerSocket.
void	setPerformancePreferences(int connectionTime, int latency, int bandwidth) Sets performance preferences for this ServerSocket.
void	setReceiveBufferSize(int size) Sets a default proposed value for the SO_RCVBUF option for sockets accepted from this ServerSocket.
void	setReuseAddress(boolean on) Enable/disable the SO_REUSEADDR socket option.
static void	setSocketFactory(SocketImplFactory fac) Sets the server socket implementation factory for the application.
void	setSoTimeout(int timeout) Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds.
String	toString() Returns the implementation address and implementation port of this socket as a String.

In order to establish a client server, class Socket will be used. A socket is an endpoint for communication between two machines. In order to call Socket class
Socket connection = new Socket(serverAddress, port);
Statement can be used. The detailed description of the Socket class is given below.

Constructor Summary	
	Socket() Creates an unconnected socket, with the system-default type of SocketImpl.
	Socket(InetAddress address, int port) Creates a stream socket and connects it to the specified port number at the specified IP address.
	Socket(InetAddress host, int port, boolean stream) **Deprecated.** *Use DatagramSocket instead for UDP transport.* Socket(InetAddress address, int port, InetAddress localAddr, int localPort) Creates a socket and connects it to the specified remote address on the specified remote port.
protected	Socket(SocketImpl impl) Creates an unconnected Socket with a user-specified SocketImpl.
	Socket(String host, int port) Creates a stream socket and connects it to the specified port number on the named host.
	Socket(String host, int port, boolean stream) **Deprecated.** *Use DatagramSocket instead for UDP transport.*
	Socket(String host, int port, InetAddress localAddr, int localPort) Creates a socket and connects it to the specified remote host on the specified remote port.

Method Summary	
void	bind(SocketAddress bindpoint) Binds the socket to a local address.
void	close() Closes this socket.
void	connect(SocketAddress endpoint) Connects this socket to the server.
void	connect(SocketAddress endpoint, int timeout) Connects this socket to the server with a specified timeout value.
SocketChannel	getChannel() Returns the unique SocketChannel object associated with this socket, if any.
InetAddress	getInetAddress()

	Returns the address to which the socket is connected.
InputStream	getInputStream() Returns an input stream for this socket.
boolean	getKeepAlive() Tests if SO_KEEPALIVE is enabled.
InetAddress	getLocalAddress() Gets the local address to which the socket is bound.
int	getLocalPort() Returns the local port to which this socket is bound.
SocketAddress	getLocalSocketAddress() Returns the address of the endpoint this socket is bound to, or null if it is not bound yet.
boolean	getOOBInline() Tests if OOBINLINE is enabled.
OutputStream	getOutputStream() Returns an output stream for this socket.
int	getPort() Returns the remote port to which this socket is connected.
int	getReceiveBufferSize() Gets the value of the SO_RCVBUF option for this Socket, that is the buffer size used by the platform for input on this Socket.
SocketAddress	getRemoteSocketAddress() Returns the address of the endpoint this socket is connected to, or null if it is unconnected.
boolean	getReuseAddress() Tests if SO_REUSEADDR is enabled.
int	getSendBufferSize() Get value of the SO_SNDBUF option for this Socket, that is the buffer size used by the platform for output on this Socket.
int	getSoLinger() Returns setting for SO_LINGER.
int	getSoTimeout() Returns setting for SO_TIMEOUT.
boolean	getTcpNoDelay() Tests if TCP_NODELAY is enabled.
int	getTrafficClass() Gets traffic class or type-of-service in the IP header for packets sent from this Socket
boolean	isBound() Returns the binding state of the socket.
boolean	isClosed() Returns the closed state of the socket.
boolean	isConnected() Returns the connection state of the socket.
boolean	isInputShutdown() Returns wether the read-half of the socket connection is closed.
boolean	isOutputShutdown() Returns wether the write-half of the socket connection is closed.
void	sendUrgentData(int data) Send one byte of urgent data on the socket.
void	setKeepAlive(boolean on) Enable/disable SO_KEEPALIVE.
void	setOOBInline(boolean on) Enable/disable OOBINLINE (receipt of TCP urgent data) By default, this option is disabled and TCP urgent data received on a socket is silently discarded.
void	setReceiveBufferSize(int size) Sets the SO_RCVBUF option to the specified value for this Socket.

void	setReuseAddress(boolean on) Enable/disable the SO_REUSEADDR socket option.
void	setSendBufferSize(int size) Sets the SO_SNDBUF option to the specified value for this Socket.
static void	setSocketImplFactory(SocketImplFactory fac) Sets the client socket implementation factory for the application.
void	setSoLinger(boolean on, int linger) Enable/disable SO_LINGER with the specified linger time in seconds.
void	setSoTimeout(int timeout) Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds.
void	setTcpNoDelay(boolean on) Enable/disable TCP_NODELAY (disable/enable Nagle's algorithm).
void	setTrafficClass(int tc) Sets traffic class or type-of-service octet in the IP header for packets sent from this Socket.
void	shutdownInput() Places the input stream for this socket at "end of stream".
void	shutdownOutput() Disables the output stream for this socket.
String	toString() Converts this socket to a String.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Program 20.2-1 Server.java

```
// Set up a server that will receive a connection froma client, send
// a string to the client, and close the connection.
import java.io.EOFException;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.SwingUtilities;

public class Server extends JFrame
{
private JTextField enterField; // inputs message from user
private JTextArea displayArea; // display information to user
private ObjectOutputStream output; // output stream to client
private ObjectInputStream input; // input stream from client
private ServerSocket server; // server socket
private Socket connection; // connection to client
private int counter = 1; // counter of number of connections

// set up GUI
```

```
public Server()
{
super( "Server" );

enterField = new JTextField(); // create enterField
enterField.setEditable( false );
enterField.addActionListener(
new ActionListener()
{
// send message to client
public void actionPerformed( ActionEvent event )
{
sendData( event.getActionCommand() );
enterField.setText( "" );
} // end method actionPerformed
} // end anonymous inner class
); // end call to addActionListener

add( enterField, BorderLayout.NORTH );

displayArea = new JTextArea(); // create displayArea
add( new JScrollPane( displayArea ), BorderLayout.CENTER );

setSize( 300, 150 ); // set size of window
setVisible( true ); // show window
} // end Server constructor

// set up and run server
public void runServer()
{
try // set up server to receive connections; process connections
{
server = new ServerSocket( 12345, 100 ); // create ServerSocket

while ( true )
{
try
{
waitForConnection(); // wait for a connection
getStreams(); // get input & output streams
processConnection(); // process connection
} // end try
catch ( EOFException eofException )
{
displayMessage( "\nServer terminated connection" );
} // end catch
finally
{
closeConnection(); // close connection
counter++;
} // end finally
} // end while
} // end try
catch ( IOException ioException )
{
ioException.printStackTrace();
} // end catch
} // end method runServer
```

```
// wait for connection to arrive, then display connection info
private void waitForConnection() throws IOException
{
displayMessage( "Waiting for connection\n" );
connection = server.accept(); // allow server toaccept connection
displayMessage( "Connection " + counter + " received from: " +
connection.getInetAddress().getHostName() );
} // end method waitForConnection

// get streams to send and receive data
private void getStreams() throws IOException
{
// set up output stream for objects
output = new ObjectOutputStream( connection.getOutputStream() );
output.flush(); // flush output buffer to send header information

// set up input stream for objects
input = new ObjectInputStream( connection.getInputStream() );

displayMessage( "\nGot I/O streams\n" );
} // end method getStreams

// process connection with client
private void processConnection() throws IOException
{
String message = "Connection successful";
sendData( message ); // send connection successfulmessage

// enable enterField so server user can send messages
setTextFieldEditable( true );

do // process messages sent from client
{
try // read message and display it
{
message = ( String ) input.readObject(); //read new message
displayMessage( "\n" + message ); // displaymessage
} // end try
catch ( ClassNotFoundException classNotFoundException )
{
displayMessage( "\nUnknown object type received" );
} // end catch

} while ( !message.equals( "CLIENT>>> TERMINATE")); 
} // end method processConnection

// close streams and socket
private void closeConnection()
{
displayMessage( "\nTerminating connection\n" );
setTextFieldEditable( false ); // disable enterField

try
{
output.close(); // close output stream
input.close(); // close input stream
connection.close(); // close socket
```

```
} // end try
catch ( IOException ioException )
{
ioException.printStackTrace();
} // end catch
} // end method closeConnection

// send message to client
private void sendData( String message )
{
try // send object to client
{
output.writeObject( "SERVER>>> " + message );
output.flush(); // flush output to client
displayMessage( "\nSERVER>>> " + message );
} // end try
catch ( IOException ioException )
{
displayArea.append( "\nError writing object");
} // end catch
} // end method sendData

// manipulates displayArea in the event-dispatch thread
private void displayMessage( final String messageToDisplay )
{
SwingUtilities.invokeLater(
new Runnable()
{
public void run() // updates displayArea
{
displayArea.append( messageToDisplay );// append message
} // end method run
} // end anonymous inner class
); // end call to SwingUtilities.invokeLater
} // end method displayMessage

// manipulates enterField in the event-dispatch thread
private void setTextFieldEditable( final boolean editable )
{
SwingUtilities.invokeLater(
new Runnable()
{
public void run() // sets enterField's editability
{
enterField.setEditable( editable );
} // end method run
} // end inner class
); // end call to SwingUtilities.invokeLater
} // end method setTextFieldEditable
      public static void main( String args[] )
      {
        Server application = new Server(); // create server
       application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
       application.runServer(); // run server application
     } // end main
} // end class Server
```

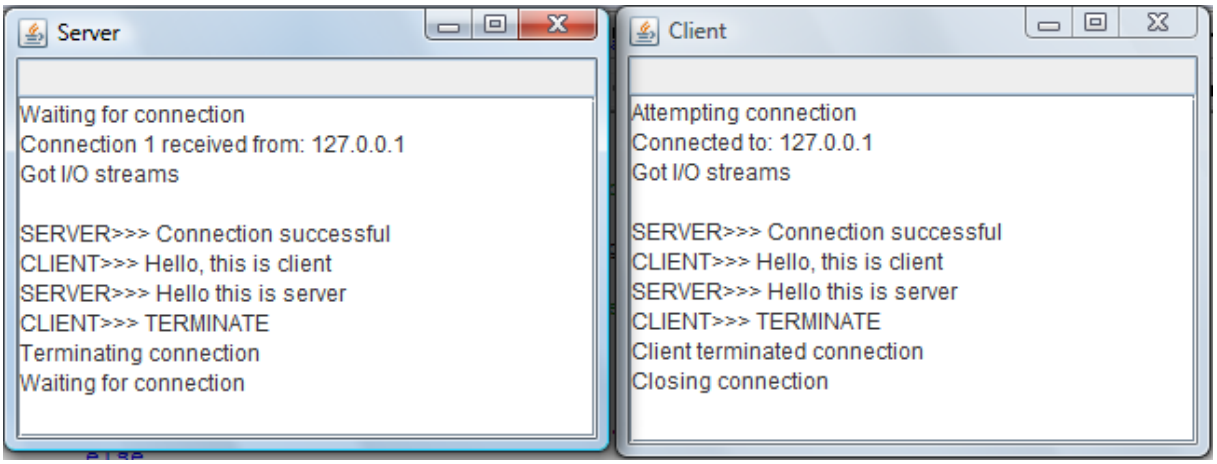
Program 20.2-2 Client.java

```
//  Client.java
// Client that reads and displays information sent from a Server.
import java.io.EOFException;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.InetAddress;
import java.net.Socket;
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.SwingUtilities;

public class Client extends JFrame
{
   private JTextField enterField; // enters information from user
   private JTextArea displayArea; // display information to user
   private ObjectOutputStream output; // output stream to server
   private ObjectInputStream input; // input stream from server
   private String message = ""; // message from server
   private String chatServer; // host server for this application
   private Socket client; // socket to communicate with server

   // initialize chatServer and set up GUI
   public Client( String host )
   {
      super( "Client" );

      chatServer = host; // set server to which this client connects

      enterField = new JTextField(); // create enterField
      enterField.setEditable( false );
      enterField.addActionListener(
         new ActionListener()
         {
            // send message to server
            public void actionPerformed( ActionEvent event )
            {
               sendData( event.getActionCommand() );
               enterField.setText( "" );
            } // end method actionPerformed
         } // end anonymous inner class
      ); // end call to addActionListener

      add( enterField, BorderLayout.NORTH );

      displayArea = new JTextArea(); // create displayArea
      add( new JScrollPane( displayArea ), BorderLayout.CENTER );

     setSize( 300, 150 ); // set size of window
      setVisible( true ); // show window
   } // end Client constructor

   // connect to server and process messages from server
```

```
public void runClient()
{
   try // connect to server, get streams, process connection
   {
      connectToServer(); // create a Socket to make connection
      getStreams(); // get the input and output streams
      processConnection(); // process connection
   } // end try
   catch ( EOFException eofException )
   {
      displayMessage( "\nClient terminated connection" );
   } // end catch
   catch ( IOException ioException )
   {
      ioException.printStackTrace();
   } // end catch
   finally
   {
      closeConnection(); // close connection
   } // end finally
} // end method runClient

// connect to server
private void connectToServer() throws IOException
{
   displayMessage( "Attempting connection\n" );

   // create Socket to make connection to server
   client = new Socket( InetAddress.getByName( chatServer ), 12345 );

   // display connection information
   displayMessage( "Connected to: " +
      client.getInetAddress().getHostName() );
} // end method connectToServer

// get streams to send and receive data
private void getStreams() throws IOException
{
   // set up output stream for objects
   output = new ObjectOutputStream( client.getOutputStream() );
   output.flush(); // flush output buffer to send header information

   // set up input stream for objects
   input = new ObjectInputStream( client.getInputStream() );

   displayMessage( "\nGot I/O streams\n" );
} // end method getStreams

// process connection with server
private void processConnection() throws IOException
{
   // enable enterField so client user can send messages
   setTextFieldEditable( true );

   do // process messages sent from server
   {
      try // read message and display it
      {
```

```
        message = ( String ) input.readObject(); // read new message
        displayMessage( "\n" + message ); // display message
      } // end try
      catch ( ClassNotFoundException classNotFoundException )
      {
        displayMessage( "\nUnknown object type received" );
      } // end catch

    } while ( !message.equals( "SERVER>>> TERMINATE" ) );
  } // end method processConnection

  // close streams and socket
  private void closeConnection()
  {
    displayMessage( "\nClosing connection" );
    setTextFieldEditable( false ); // disable enterField

    try
    {
      output.close(); // close output stream
      input.close(); // close input stream  1
      client.close(); // close socket
    } // end try
    catch ( IOException ioException )
    {
      ioException.printStackTrace();
    } // end catch
  } // end method closeConnection

  // send message to server
  private void sendData( String message )
  {
    try // send object to server
    {
      output.writeObject( "CLIENT>>> " + message );
      output.flush(); // flush data to output
      displayMessage( "\nCLIENT>>> " + message );
    } // end try
    catch ( IOException ioException )
    {
      displayArea.append( "\nError writing object" );
    } // end catch
  } // end method sendData

  // manipulates displayArea in the event-dispatch thread
  private void displayMessage( final String messageToDisplay )
  {
    SwingUtilities.invokeLater(
      new Runnable()
      {
        public void run() // updates displayArea
        {
          displayArea.append( messageToDisplay );
        } // end method run
      } // end anonymous inner class
    ); // end call to SwingUtilities.invokeLater
  } // end method displayMessage
```

```
  // manipulates enterField in the event-dispatch thread
  private void setTextFieldEditable( final boolean editable )
  {
    SwingUtilities.invokeLater(
      new Runnable()
      {
        public void run() // sets enterField's editability
        {
          enterField.setEditable( editable );
        } // end method run
      } // end anonymous inner class
    ); // end call to SwingUtilities.invokeLater
  } // end method setTextFieldEditable
      public static void main( String args[] )
    {
      Client application; // declare client application

     // if no command line args
     if ( args.length == 0 )
       application = new Client( "127.0.0.1" ); // connect to localhost
     else
       application = new Client( args[ 0 ] ); // use args to connect

     application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
     application.runClient(); // run client application
    } // end main
} // end class Client
```



Our Client server programs operating together defines a chat program. They are commonly used in internet environment nowadays.

20.3 USER DATAGRAM PROTOCOL (UDP)

Connection-oriented transmission (TCP/IP) that covered in the previous section is like the telephone system in which you dial and are given a connection to the telephone of the person with whom you wish to communicate. Connectionless transmission with datagrams is more like the way the postal service carries mail. User Datagram Protocol (UDP) is the standart protocol to send data through Datagrams. In order to send and receive data through Datagrams class DatagramSocket is used. The detailed method description of this class is given below.

Constructor Summary	
	DatagramSocket() Constructs a datagram socket and binds it to any available port on the local host machine.
protected	DatagramSocket(DatagramSocketImpl impl) Creates an unbound datagram socket with the specified DatagramSocketImpl.
	DatagramSocket(int port) Constructs a datagram socket and binds it to the specified port on the local host machine.
	DatagramSocket(int port, InetAddress laddr) Creates a datagram socket, bound to the specified local address.
	DatagramSocket(SocketAddress bindaddr) Creates a datagram socket, bound to the specified local socket address.

Method Summary	
void	bind(SocketAddress addr) Binds this DatagramSocket to a specific address & port.
void void	close() Closes this datagram socket. connect(InetAddress address, int port) Connects the socket to a remote address for this socket.
void	connect(SocketAddress addr) Connects this socket to a remote socket address (IP address + port number).
void boolean	disconnect() Disconnects the socket. getBroadcast() Tests if SO_BROADCAST is enabled.
DatagramChannel InetAddress InetAddress	getChannel() Returns the unique DatagramChannel object associated with this datagram socket, if any. getInetAddress() Returns the address to which this socket is connected. getLocalAddress() Gets the local address to which the socket is bound.
int	getLocalPort() Returns the port number on the local host to which this socket is bound.

SocketAddress	getLocalSocketAddress() Returns the address of the endpoint this socket is bound to, or null if it is not bound yet.
int	getPort() Returns the port for this socket.
int	getReceiveBufferSize() Get value of the SO_RCVBUF option for this DatagramSocket, that is the buffer size used by the platform for input on this DatagramSocket.
SocketAddress	getRemoteSocketAddress() Returns the address of the endpoint this socket is connected to, or null if it is unconnected.
boolean	getReuseAddress() Tests if SO_REUSEADDR is enabled.
int	getSendBufferSize() Get value of the SO_SNDBUF option for this DatagramSocket, that is the buffer size used by the platform for output on this DatagramSocket.
int	getSoTimeout() Retrive setting for SO_TIMEOUT.
int	getTrafficClass() Gets traffic class or type-of-service in the IP datagram header for packets sent from this DatagramSocket.
boolean	isBound() Returns the binding state of the socket.
boolean	isClosed() Returns wether the socket is closed or not.
boolean	isConnected() Returns the connection state of the socket.
void	receive(DatagramPacket p) Receives a datagram packet from this socket.
void	send(DatagramPacket p) Sends a datagram packet from this socket.
void	setBroadcast(boolean on) Enable/disable SO_BROADCAST.
static void	setDatagramSocketImplFactory(DatagramSocketImplFactory fac) Sets the datagram socket implementation factory for the application.
void	setReceiveBufferSize(int size) Sets the SO_RCVBUF option to the specified value for this DatagramSocket.

void	setReuseAddress(boolean on) Enable/disable the SO_REUSEADDR socket option.
void	setSendBufferSize(int size) Sets the SO_SNDBUF option to the specified value for this DatagramSocket.
void	setSoTimeout(int timeout) Enable/disable SO_TIMEOUT with the specified timeout, in milliseconds.
void	setTrafficClass(int tc) Sets traffic class or type-of-service octet in the IP datagram header for datagrams sent from this DatagramSocket.

Methods inherited from class java.lang. Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

The following server/Client example utilises DatagramSocket class.

Program 20.3-1 Server.java with DatagramSocket class

```
// Server.java
// Server that receives and sends packets from/to a client.
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.SocketException;
import java.awt.BorderLayout;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;

public class Server extends JFrame
{
  private JTextArea displayArea; // displays packets received
  private DatagramSocket socket; // socket to connect to client

  // set up GUI and DatagramSocket
  public Server()
  {
    super( "Server" );

    displayArea = new JTextArea(); // create displayArea
    add( new JScrollPane( displayArea ), BorderLayout.CENTER );
    setSize( 400, 300 ); // set size of window
    setVisible( true ); // show window

    try // create DatagramSocket for sending and receiving packets
    {
      socket = new DatagramSocket( 5000 );
    } // end try
    catch ( SocketException socketException )
    {
      socketException.printStackTrace();
      System.exit( 1 );
    } // end catch
  } // end Server constructor
```

```
// wait for packets to arrive, display data and echo packet to client
public void waitForPackets()
{
   while ( true )
   {
      try // receive packet, display contents, return copy to client
      {
         byte data[] = new byte[ 100 ]; // set up packet
         DatagramPacket receivePacket =
            new DatagramPacket( data, data.length );

         socket.receive( receivePacket ); // wait to receive packet

         // display information from received packet
         displayMessage( "\nPacket received:" +

            "\nFrom host: " + receivePacket.getAddress() +
            "\nHost port: " + receivePacket.getPort() +
            "\nLength: " + receivePacket.getLength() +
            "\nContaining:\n\t" + new String( receivePacket.getData(),
               0, receivePacket.getLength() ) );

         sendPacketToClient( receivePacket ); // send packet to client
      } // end try
      catch ( IOException ioException )
      {
         displayMessage( ioException.toString() + "\n" );
         ioException.printStackTrace();
      } // end catch
   } // end while
} // end method waitForPackets

// echo packet to client
private void sendPacketToClient( DatagramPacket receivePacket )
   throws IOException
{
   displayMessage( "\n\nEcho data to client..." );

   // create packet to send
   DatagramPacket sendPacket = new DatagramPacket(
      receivePacket.getData(), receivePacket.getLength(),
      receivePacket.getAddress(), receivePacket.getPort() );

   socket.send( sendPacket ); // send packet to client
   displayMessage( "Packet sent\n" );
} // end method sendPacketToClient

// manipulates displayArea in the event-dispatch thread
private void displayMessage( final String messageToDisplay )
{
   SwingUtilities.invokeLater(
      new Runnable()
      {
         public void run() // updates displayArea
         {
            displayArea.append( messageToDisplay ); // display message
         } // end method run
      } // end anonymous inner class
   ); // end call to SwingUtilities.invokeLater
} // end method displayMessage
 public static void main( String args[] )
 {
    Server application = new Server(); // create server
```

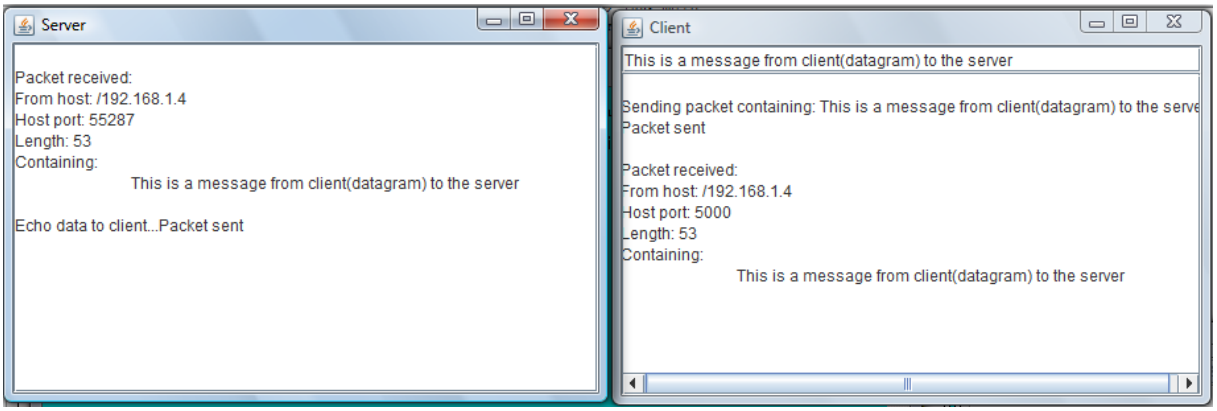
```
      application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      application.waitForPackets(); // run server application
   } // end main



} // end class Server
```

Program 20.3-1 Client.java with DatagramSocket class

```
// Client.java
// Client that sends and receives packets to/from a server.
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.SwingUtilities;

public class Client extends JFrame
{
   private JTextField enterField; // for entering messages
   private JTextArea displayArea; // for displaying messages
   private DatagramSocket socket; // socket to connect to server

   // set up GUI and DatagramSocket
   public Client()
   {
      super( "Client" );

      enterField = new JTextField( "Type message here" );
      enterField.addActionListener(
         new ActionListener()
         {
            public void actionPerformed( ActionEvent event )
            {
               try // create and send packet
               {
                  // get message from textfield
                  String message = event.getActionCommand();
                  displayArea.append( "\nSending packet containing: " +
                     message + "\n" );

                  byte data[] = message.getBytes(); // convert to bytes

                  // create sendPacket
                  DatagramPacket sendPacket = new DatagramPacket( data,
                     data.length, InetAddress.getLocalHost(), 5000 );

                  socket.send( sendPacket ); // send packet
                  displayArea.append( "Packet sent\n" );
                  displayArea.setCaretPosition(
                     displayArea.getText().length() );
               } // end try
               catch ( IOException ioException )
               {
                  displayMessage( ioException.toString() + "\n" );
                  ioException.printStackTrace();
               } // end catch
            } // end actionPerformed
         } // end inner class
```

```
    ); // end call to addActionListener

    add( enterField, BorderLayout.NORTH );

    displayArea = new JTextArea();
    add( new JScrollPane( displayArea ), BorderLayout.CENTER );

    setSize( 400, 300 ); // set window size
    setVisible( true ); // show window

    try // create DatagramSocket for sending and receiving packets
    {
      socket = new DatagramSocket();
    } // end try
    catch ( SocketException socketException )
    {
      socketException.printStackTrace();
      System.exit( 1 );
    } // end catch
  } // end Client constructor

  // wait for packets to arrive from Server, display packet contents
  public void waitForPackets()
  {
    while ( true )
    {
      try // receive packet and display contents
      {
        byte data[] = new byte[ 100 ]; // set up packet
        DatagramPacket receivePacket = new DatagramPacket(
          data, data.length );

        socket.receive( receivePacket ); // wait for packet

        // display packet contents
        displayMessage( "\nPacket received:" +
          "\nFrom host: " + receivePacket.getAddress() +
          "\nHost port: " + receivePacket.getPort() +
          "\nLength: " + receivePacket.getLength() +
          "\nContaining:\n\t" + new String( receivePacket.getData(),
            0, receivePacket.getLength() ) );
      } // end try
      catch ( IOException exception )
      {
        displayMessage( exception.toString() + "\n" );
        exception.printStackTrace();
      } // end catch
    } // end while
  } // end method waitForPackets

  // manipulates displayArea in the event-dispatch thread
  private void displayMessage( final String messageToDisplay )
  {
    SwingUtilities.invokeLater(
      new Runnable()
      {
        public void run() // updates displayArea
        {
          displayArea.append( messageToDisplay );
        } // end method run
      } // end inner class
    ); // end call to SwingUtilities.invokeLater
  } // end method displayMessage
    public static void main( String args[] )
    {
     Client application = new Client(); // create client
     application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
```

```
        application.waitForPackets(); // run client application
    } // end main

} // end class Client
```



21. **PLOTTING DATA, DATA INPUT FROM PLOT SCREEN (DIGITIZING)**

21.1 PLOTTING TWO DIMENSIONAL DATA AND FUNCTIONS

Plot program aims a simple yet sophisticated set of plots to be drawn for two dimensional engineering plot applications. Seeing the data is always easier for evaluating the results. There are commercially available plot programs, but the purpose of this package is to create an absolutely free and open code supporting plot definitions. Most of the subprograms in the package is add on methods from the actual requirements arised in numerical analysis and engineering applications.

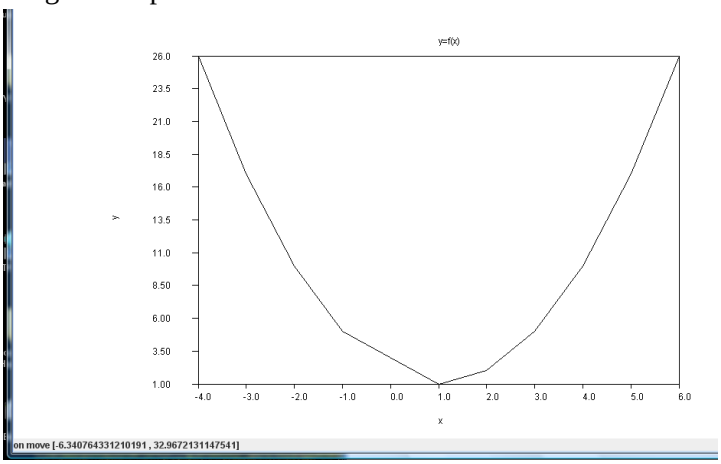
Program 21.1. PlotT1.java SAMPLE PROGRAM

```
import java.io.*;
 class PlotT1
 {  public static void main(String args[]) throws IOException
   {Plot pp=new Plot("in.txt");
    pp.plot();}}
```

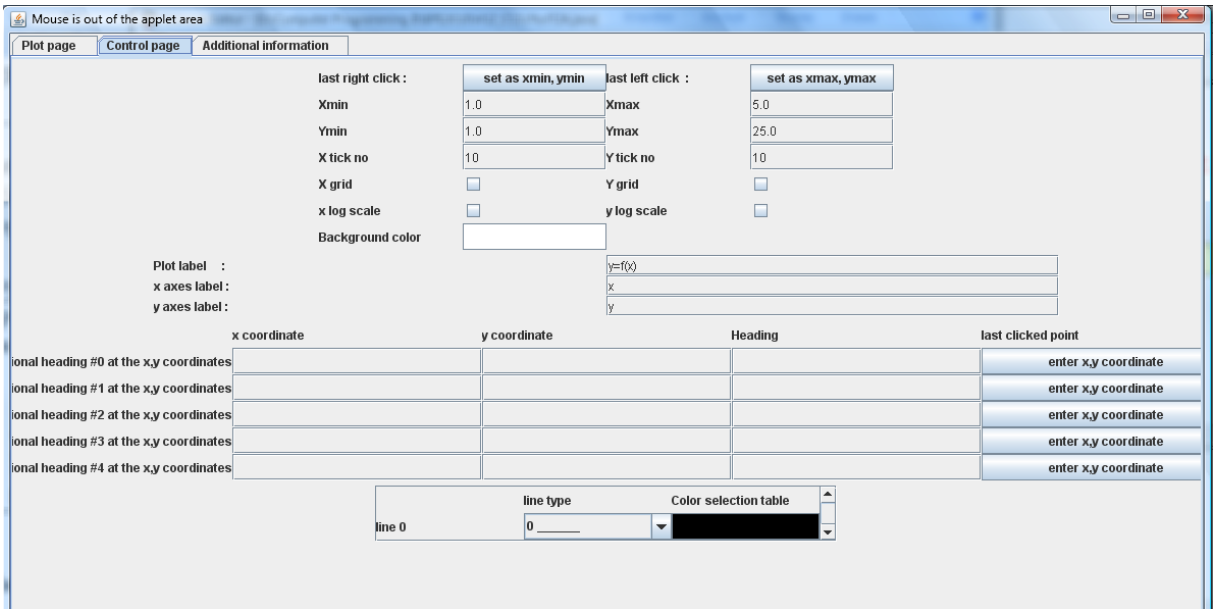
Data in.txt

```
-4 26.0
-3 17.0
-2 10.0
-1 5.0
1 1.0
2 2.0
3 5.0
4 10.0
5 17.0
6 26.0
```

Program output:



Our first example code reads data from file in.txt and plot it The first line is the constructor line **Plot pp=new Plot("in.txt");** it inputs data from file in.txt **and pp.plot();** term is actually draw the plot window. This term will always be used to open plot window In the second page of the graphic window several editing features are available such as plot shapes, headings, colors, grid, logarithmic scale, manual maximum and minimum x an y scales, additional heading space in any desired location of the plot. Page can be grided into a desired size, Graphic color can be changed by using standard java JColorChooser class graphic editor.



In our second example program data in file in.txt is drawn by a single static definition

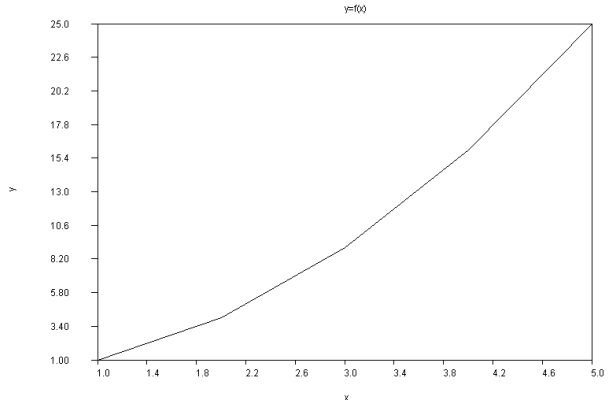
Program 21.1-2 PlotT1A.java sample program

```
import java.io.*;
class plotT1A
 {
   public static void main(String args[]) throws IOException
   {Plot.file("in.txt");}
 }
```

Static definitions are usually preferred for one step plot drawing If several different plot jobs will be defined, dynamic definitions should be preferrred. In the next example data is given inside of the program as two double vectors

Program 21.1-3 PlotT2.java program plotting directly from arrays

```
import java.io.*;
 class PlotT2
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
      double yy[]={1,4,9,16,25};
      Plot pp=new Plot(xx,yy);
      pp.plot();}}
```



The static version of the same program :

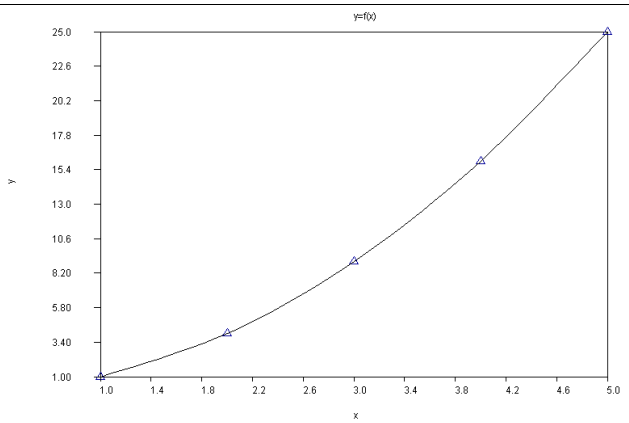
Program 21.1-4 PlotT2A, plotting a vector data

```
import java.io.*;
 class PlotT2A
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
       double yy[]={1,4,9,16,25};
       Plot.data(xx,yy);}}
```

The next program first defines the data and then add up cubic spline curve fittings to the given data (2 plot lines are to be drawn)

Program 21.1.-5 PlotT2B, plotting a vector data and plotting cubic spline curve fitting of the data

```
import java.io.*;
 class PlotT2B
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
       double yy[]={1,4,9,16,25};
       Plot pp=new Plot(xx,yy);
       pp.addSpline(xx,yy,10);
       pp.plot();}}
```



In order to do Cubic spline curve fitting adding term **pp.addSpline(xx,yy,10);** was enough

10 additional cubic spline fit data is added in between every point. Plot data added into the plot window has an index number starting from 0 and goes upwards for every new curve to be added into the pot window. If the original curve index number is known, it can also be used to curve feet additional data.

Program 21.1.-6 PlotT2B, plotting a vector data and plotting cubic spline curve fitting of the data index number referencing to data is used for spline fitting

```
 import java.io.*;
class PlotT2B2
{    public static void main (String args[]) throws IOException
     {double xx[]={1,2,3,4,5};
      double yy[]={1,4,9,16,25};
      Plot pp=new Plot(xx,yy);
      pp.addSpline(0,10);
      pp.plot();}}
```

Static definitions can not be applied in the plot drawings with more than one command such as in the above program, but you can always write your user specific static method to collect your own group of commands

Program 21.1.-7 PlotT2B, plotting a vector data and plotting cubic spline curve fitting of the data user defined static method is used.

```
import java.io.*;
class PlotT2B3
{
     public static void plot(double xx[],double yy[])
     {Plot pp=new Plot(xx,yy);
      pp.addSpline(xx,yy,10);
      pp.plot();}

     public static void main (String args[]) throws IOException
     {double xx[]={1,2,3,4,5};
      double yy[]={1,4,9,16,25};
      plot(xx,yy);}}
```

The same interpolation calculations can also be carried out by B spline interpolation with term **pp.addB_Spline(xx,yy,10);**

Program 21.1.-8 PlotT2B1, plotting a vector data and plotting cubic spline curve fitting of the data user defined static method is used.

```
import java.io.*;
class PlotT2B1
{    public static void main (String args[]) throws IOException
     {double xx[]={1,2,3,4,5};
      double yy[]={1,4,9,16,25};
      Plot pp=new Plot(xx,yy);
      pp.addB_Spline(xx,yy,10);
      pp.plot();}}
```

If second degree orthogonal polynomial to be used for curve fitting

Program 21.1.-9 PlotT2C, plotting a vector data and plotting Orthogonal polynomial least square curve fitting of the data.

```
import java.io.*;
 class PlotT2C
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
       double yy[]={1,4,9,16,25};
       Plot pp=new Plot(xx,yy);
       pp.addOPEKK(xx,yy,2,10);
       pp.plot();}}
```

if a second degree polynomial is desired from curve fitting (fx=a0+a1*x+a2*x2)

Program 21.1.-10 PlotT2D, plotting a vector data and plotting quadratic polynomial least square curve fitting of the data.

```
import java.io.*;
 class PlotT2D
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
       double yy[]={1,4,9,16,25};
       Plot pp=new Plot(xx,yy);
       pp.addEKK(xx,yy,2,10);
       pp.plot();}}
```

Lagrange interpolation can also be used.

Program 21.1.-11 PlotT2E, plotting a vector data and plotting Lagrange interpolation curve fitting of the data.

```
import java.io.*;
 class PlotT2E
 {    public static void main (String args[]) throws IOException
      {double xx[]={1,2,3,4,5};
       double yy[]={1,4,9,16,25};
       Plot pp=new Plot(xx,yy);
       pp.addLagrange(xx,yy,10);
       pp.plot();}}
```

The detailed information about interpolation routines are given in Numerical Analysis by using java programs text book.

Constructor in the plot program can be given as completely empty.

Program 21.1.-12 PlotT01, plotting data from information given in Plot.txt file

```
import java.io.*;

class PlotT01
{
   public static void main (String args[]) throws java.io.IOException
```

```
   {
          Plot pp=new Plot();
   pp.plot();
    }
}
```

Program 21.1.-13 PlotT2B1, plotting data from information given in Plot.txt file static version

```
// (Static call)
 import java.io.*;
 class PlotT2B1
 {    public static void main (String args[]) throws IOException
      {Plot.file();}}
```

In this case program will look the data file Plot.txt for data.

Data is given in the following form in this data
Plot label
x axis label
y axis label
2
in.txt 0 0 0 0
out.txt 3 0 0 255

The meaning of this data is as follows "Plot label" "x axis label"
"y axis label"
are defines graphic label headings
2 indicates number of data files data to be taken from in.txt and out.txt
in this case, the next number is the plot line type and the last 3 numbers are plot color
(red green and blue 0 to 255).
Plot line type codes are as follows :
"1 __ __ __ __ ",
"2 . _ . _ . _ ",
"3 . _ _ . . _ _ .",
"4 - - - - - - - - ",
"5 _____",
"6 __ __ __ __ ",
"7 . _ . _ . _ ",
"8 . _ _ . . _ _ .",
"9 - - - - - ",
"10 * ",
"11 # ",
"12 $ ",
"13 % ",
"14 & ",
"15 - ",
"16 ~ ",
"17 + ",

```
"18  x       ",
"19  =       ",
"20  square",
"21  filled square",
"22  circle",
"23  filled circle",
"24  triangle",
"25  diamond ",
"26  pentagon",
"27  hexagon",
"28  filled triangle",
"28  filled diamond",
"30  filled pentagon",
"31  filled hexagon",
"32  3 sided star",
"33  4 sided star",
"34  5 sided star",
"35  6 sided star",
"36  filled 3 sided star",
"37  filled 4 sided star",
"38  filled 5 sided star",
"39  filled 6 sided star",
"40  bar graphic",
"41  filled bar graphic"};
.
```

In engineering and sciene, seeing behavior of a function gives us clues for several investigation purpuses. Class plot is defined to be used to plot function and data in 2 dimension problems. In order to write computer codes in this book all the functions are defined as defined from an abstract form. The abstract root function is defined in class f_x. This definition is used both in plot class and numerical analysis classes requires

Program 21.1.-14 Abstract class f_x

```
import static java.lang.Math.*;

abstract public class f_x extends Mathd
{
  // single function single independent variable
  // example f=x*x
  abstract public double func(double x);
}
```

It is also possible to use multidemensional function with one dimensional input abstract function fi_x with the plot class.

Program 21.1.-15 Abstract class fi_x

```
abstract class fi_x extends Mathd
{
  // multifunction single independent variable
  // vector of dependent variables are returned
  // example f[0]=x+sin(x)
  //              f[1]=x*x-x
  // func(x) returns the value of f[0] and f[1]
```

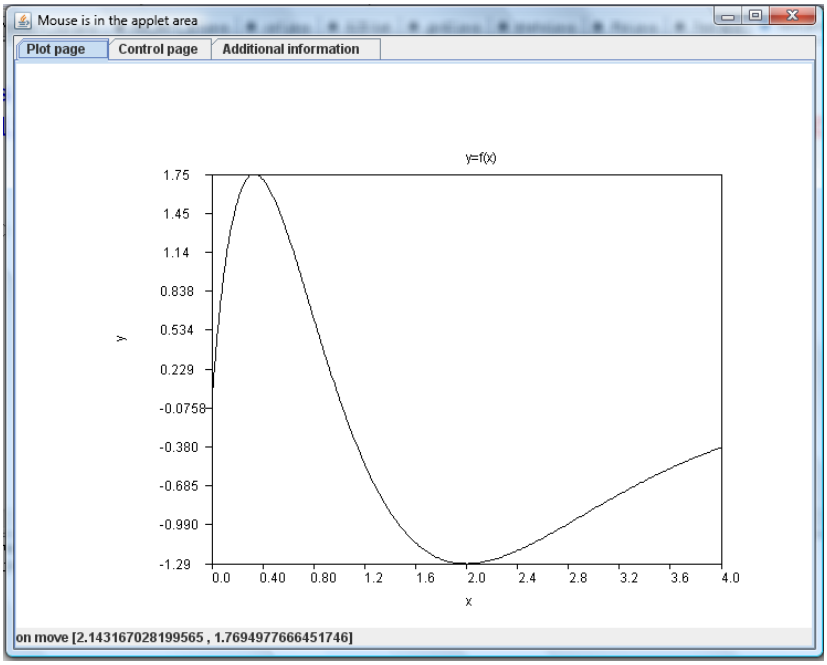
```
  // as a two dimensional vector
  abstract double[] func(double x);
}
```

By using the abstract class concept the actual function can be defined as a new class and be used directly with the plot class to plot 2D functions. It should be also note that abstract classes are derived from class Mathd. This is an addtional function library that defines functions frequently use in engineering, but not included Standard java Math library suh as Bessel functions, error functions, gamma functions etc. A further plus of extending root function classes from Mathd class is ability to utilise method names directly. For example instead of using **Math.sin(x)** required in java **sin(x)** can be used. Some example using of plot class is given below:

PROGRAM 21.1-16 Plot class example NA0

```
import java.io.*;

class f3 extends f_x
{
public double func(double x)
{
double ff;
ff=13*(x - x*x)*Math.exp(-3*x/2);
return ff;
}
}

class NA0
{public static void main (String args[]) throws IOException
  {   Plot.func(new f3(),0.0,4.0);}}
```

The result will be

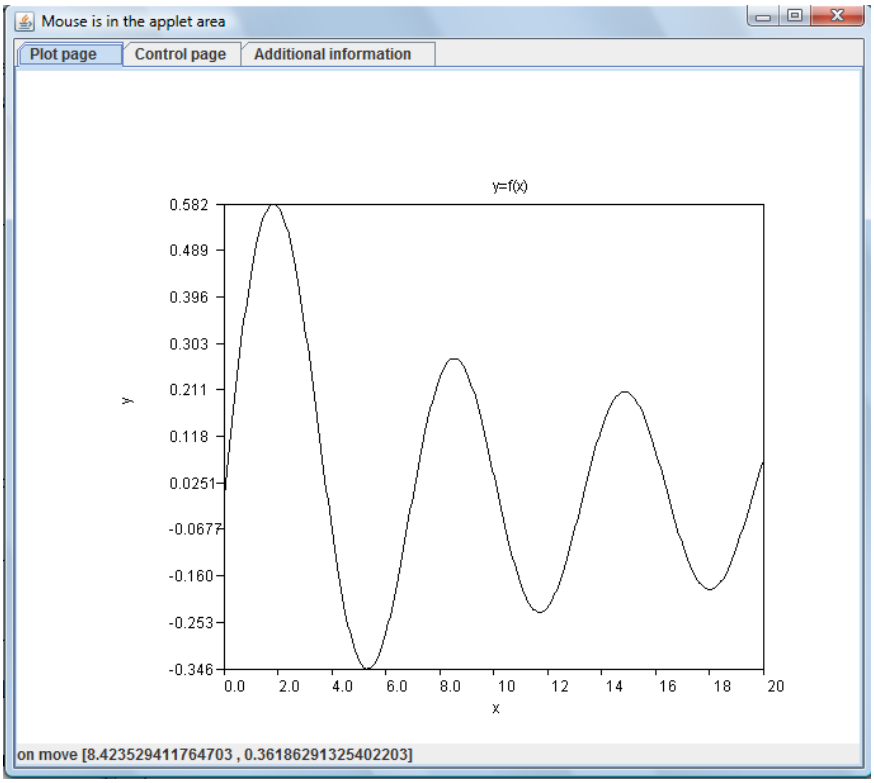


As an exercise re-write the program to plot function $f(x)=x^2-1$

As a second example Bessel function of order 1 is Plotted by using Mathd function library method J(1,x)

PROGRAM 21.1-17 Plot class exampleNA0A

```
import java.io.*;

class fa1 extends f_x
{  // Bessel function
   double func (double x)
   { return Mathd.J(1,x); }
}

class NA0A
{
     public static void main (String args[]) throws IOException
     {
     fa1 ff1=new fa1();
     Plot pp=new Plot(ff1,0.0,20.0,400,0);
     pp.plot();
     }
}
```



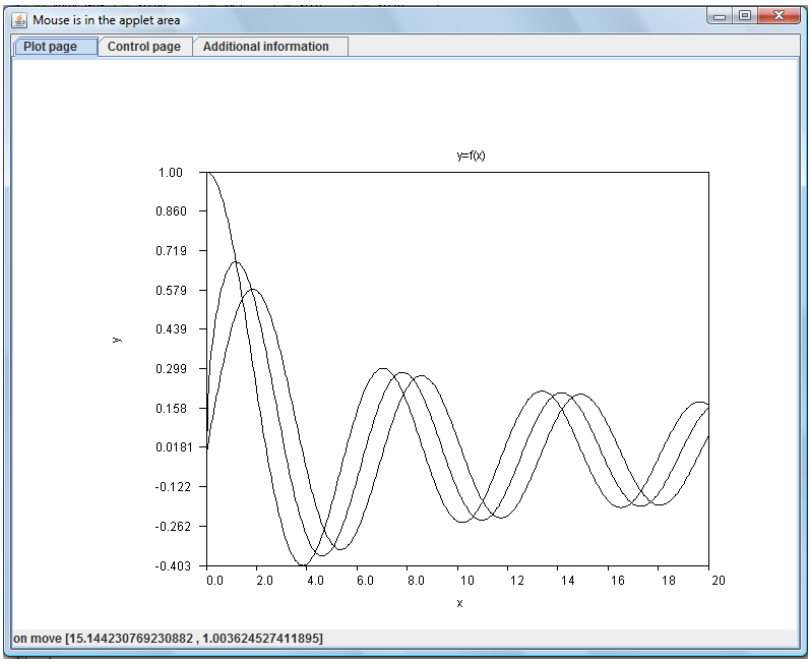
Re-write the program to plot function f(x)=sin(x)

If the Bessel function of order 0, order 0.5 and order 1 is to be plotted:

PROGRAM 21.1-18 Plot class exampleNA0C

```
İmport java.io.*;

class fa1 extends fi_x
{
   double [] func (double x)
   {
 double ff[]=new double[3];
    ff[0]=J(0,x);
    ff[1]=J(0.5,x);
```

```
   ff[2]=J(1,x);
   return ff;
  }
}

class NA0C
{
    public static void main (String args[]) throws IOException
    {
    fa1 ff1=new fa1();
    Plot pp=new Plot(ff1,0.0,20.0,400,3);
    pp.plot();
    }
}
```

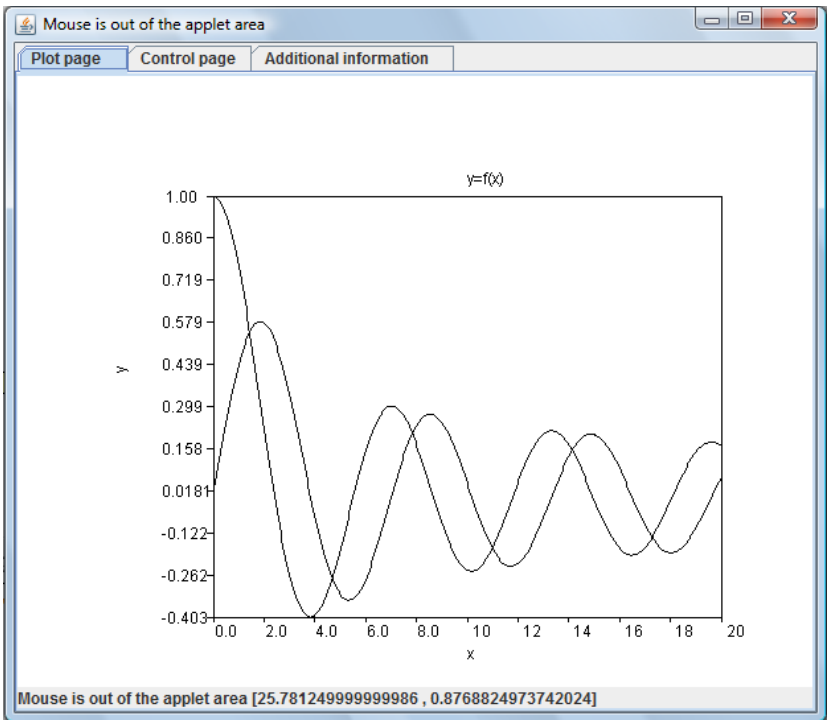


The same kind of plot can also be achived by reading different function and add second(third etc) function by using addFunction method.

PROGRAM 21.1-19 Plot class exampleNA0D

```
import java.io.*;

class fa0 extends f_x
{
  double func (double x)
  {  return J(0,x);}
}

class fa1 extends f_x
{
  double func (double x)
  { return J(1,x);}
}

class NA0D
{
```

```
    public static void main (String args[]) throws IOException
    {
    fa0 ff0=new fa0();
    fa1 ff1=new fa1();
    Plot pp=new Plot(ff0,0.0,20.0);
    pp.addFunction(ff1,0.0,20.0);
    pp.plot();
    }
}
```

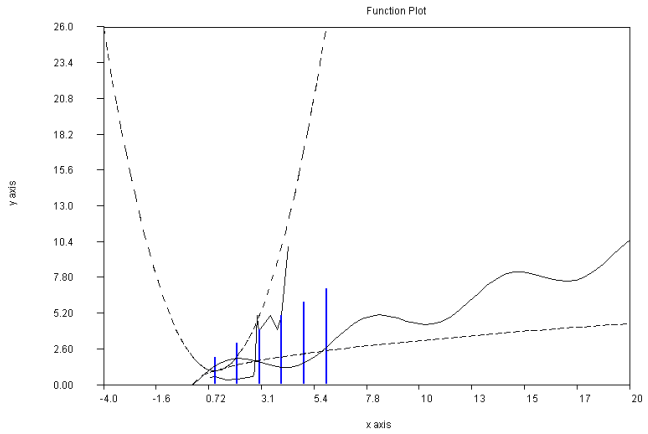


Last example combines different methods of the plot class together

PROGRAM 21.1-20 Plot class examplePlotT3

```
import java.io.*;
 class f3 extends fi_x
 {
  double[] func(double x)
  {
   double f[]=new double[2];
   f[0]=Math.sin(x)+x/2;
   f[1]=Math.sqrt(x);
   return f;
  }}
 class PlotT3
 {
     public static void main (String args[]) throws IOException
     {//Plot pp=new Plot();
     String fi[]={"in.txt","out.txt"};
     f3 ff=new f3();
     int ipt[]={0,1};
     int ir[]={0,0};
     int ig[]={0,0};
     int ib[]={0,2};
```

```
double xx[]={1,2,3,4,5,6};
double yy[]={2,3,4,5,6,7};
Plot pp=new Plot(fi); //read data from fi (in.txt,out.txt data are added up)
//pp.addData(fi);
pp.setPlotType(0,0);//
pp.setPlotType(1,1);//
pp.addData(xx,yy,0,0,0,0); //one set of addional data
pp.setPlotType(2,41);//bar chart
pp.setColor(2,0,0,255); //blue
pp.addFunction(ff,0.0,20.0,50,2,ipt,ir,ig,ib); //two additional function
pp.setPlabel("Function Plot");
pp.setXlabel("x axis");
pp.setYlabel("y axis");
pp.plot();}}
```



Grafik Package basic components :
Plot.java constructors, input and output methods
PlotW.java : basic graphic and control algorithms
PlotShapesSW: defines different plot shapes such as line, star,triangle
Plot2D : Converts shapes defined in the PlotShapesSW class into plot screen format

Constructor classes : (defines the graphic window and prepare the data to be plotted)

public Plot() throws IOException
public Plot(String pl,String xl,String yl,int xnt,int ynt,int xgo,int ygo,String fn[],int ipt[],int ir[],int ig[] ,int ib[]) throws IOException
public Plot(String fn[],int ipt[],int ir[],int ig[],int ib[]) throws IOException
public Plot(String fn[]) throws IOException : Read file names from String fn[] and plot them
public Plot(double xi[][],double yi[][],int ipt[],int ir[],int ig[],int ib[]) : read plot data from xi[][] and yi[][]
public Plot(double xiyi[][]) : read plot data xi[] and yi[] from xiyi[][] string
public Plot(double xi[][],double yi[][]) : read plot data from xi[][] and yi[][]
public Plot(double xi[],double yi[]) : read plot data from xi[] and yi[]
public Plot(f_x f,double xm,double xma,int N,int ipt,int ir,int ig,int ib)
public Plot(f_x f,double xm,double xma,int N) : create plot data from a simple function f_x f
public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function)

Static method definitions : (give short descriptions for one step plotting)

public static double[] data(double xmin,double xmax,double dx)
public static double[] data(double xmin,double xmax,int n)
public static Plot func(f_x ff1,double xmin,double xmax,int n,int pnumber)
public static Plot func(f_x ff1,double xmin,double xmax,int n)
public static Plot func(f_x ff1,double xmin,double xmax)
public static Plot func(fi_x ff1,double xmin,double xmax,int n,int pnumber)
public static Plot func(fi_x ff1,double xmin,double xmax,int n)
public static Plot func(fi_x ff1,double xmin,double xmax)

adding x,y and heading label

public void setPlabel(String ip)
public void setXlabel(String ix)
public void setYlabel(String iy)
public void setXYlabel(String ix,String iy)

Adding additional data into the graphic window

public void addData(double xi[],double yi[])
public void addData(double xi[],double yi[],int iplottype)
public void addData(double xi[],double yi[],int iplottype,int ir,int ig,int ib)
public void addData(double xi[][],double yi[][])
public void addData(double xi[][],double yi[][],int ip[])
public void addData(double xi[][],double yi[][],int ip[],int ir[],int ig[],int ib[])
public void addFunction(f_x f,double xmi,double xma,int N,int ipt)
public void addFunction(f_x f,double xmi,double xma,int N)
public void addFunction(f_x f,double xmi,double xma,int N,int ipt,int ir,int ig,int ib)
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function)
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,int ipt[])
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function, ,int ipt[])

adding data with curve fit

public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi)
public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)
public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi)
public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)
public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype)
public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addLagrange(double xi[],double yi[],int aradegersayisi)
public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype)
public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addLagrange(int linenumber,int aradegersayisi,int iplottype)
public void addLagrange(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)

public void addB_Spline(double xi[],double yi[],int nSpline)
public void addB_Spline(double ai[][],int nSpline)
public void addB_Spline(double xi[],double yi[],int nSpline,int iplottype)
public void addB_Spline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addB_Spline(int linenumber,int nSpline)
public void addB_Spline(int linenumber,int nSpline,int iplottype)
public void addB_Spline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addSpline(double xi[],double yi[],int nSpline)
public void addSpline(double ai[][],int nSpline)
public void addSpline(double xi[],double yi[],int nSpline,int iplottype)
public void addSpline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
public void addSpline(int linenumber,int nSpline)
public void addSpline(int linenumber,int nSpline,int iplottype)
public void addSpline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)

changing the graphic properties
==========Graphic type==========
public void setPlotType(int plot_type[])
public void setPlotType(int dataset,int plot_no)
public void setPlotType(int dataset,char plot_char)
==========x and y grid ==========
public void setXgrid(int igx)
public void setYgrid(int igy)
public void setGrid(int igx,int igy)
public void setXtic(int ixt)
public void setYtic(int iyt)
public void setXYtic(int ixt,int iyt)
==========Logarithmic scale on/off ==========
public void setXlogScaleOn()
public void setYlogScaleOn()
public void setXlogScaleOff()
public void setYlogScaleOff()
==========setColor==========
public void setColor(int dataset,int ired,int igreen,int iblue)
public void setColor(int dataset,Color c)
public void setColor(int ired[],int igreen[],int iblue[])
public void setColor(Color c[])
========== Setting maximum and minimum values of the plot window manually ====
public void setMinMax()
public void setMinMax(double xi[][],double yi[][])
public void setMinMax(double iminx,double imaxx,double iminy,double imaxy)
========== Formatting graphic window ==========
public void set_plotwindow(int width,int height,double xip,double yip,double dxp,double dyp)
public void setabsMaxMin(int ixmin,int iymin,int idx,int idy)

Drawing the data in graphic format
public void plot()
Complete Plot graphic codes are given belove:

PROGRAM 21.1-21 Plot class

```
//=====================================================
// Numerical Analysis package in java
// Plot class to plot 2D data
// Dr. Turhan Coban
// EGE University, Engineering School, Mechanical Eng. Div.
// turhan.coban@ege.edu.tr
// ====================================================
import java.io.*;
import java.awt.Color;
import static java.lang.Math.*;

//This class organises data and reqired plot classes for actual plot
//===== classes used by this class=======
// PlotShapesSW // define shapes to draw in plot windows in real coordinate system
// Plot2D       // Plot and coordinates data
// PLotW        // defines actual graphic screens and pages
//===== Example classes
// PlotT0,PlotT01,PlotT02,PlotT03,PlotT1,PlotT2,PlotT3,PlotT4

abstract class f_x extends Mathd
{
  // single function single independent variable
  // example f=x*x
  abstract double func(double x);
}

abstract class fi_x extends Mathd
{
  // multifunction single independent variable
  // vector of dependent variables are returned
  // example f[0]=x+sin(x)
  //         f[1]=x*x-x
  // func(x) returns the value of f[0] and f[1]
  // as a two dimensional vector
  abstract double[] func(double x);
}

abstract class f_xj extends Mathd
{
  // single function  multi independent variable
  // a single value is returned indiced to equation_ref
  // example f[0]=x[0]+sin(x[1])
  //         f[1]=x[0]*x[0]-x[1]
  // func(x,1) returns the value of f[1]
  // func(x,0) returns the value of f[0]

  abstract double func(double x[]);
}

public class Plot
{
  // This class reads plot input variables
  // and sets relations between absolute scale and relative scale plot windows
  public Color bg=Color.blue;
  public String label="y=f(x)";                // Plot Label
  public String xlabel="x";                    // X axis Label
  public String ylabel="y";                    // Y axis Label
  public String label_at_xy[];                 //additional label at x,y point
  public double label_x[];                     //additional label x points
  public double label_y[];                     //additional label x points
  public String filename;                      // file name
  public double xmin,ymin,xmax,ymax;           // real max,min scale
  public int xabsmin,yabsmin,abswidth,absheight; // absolute max,min scale in Pixel dimension
```

```
  public int nmax;                          // max number of data points
  public int ni;                          // data set number with nmax data
  public int nline;                          // max number of data sets
  public double x[][];                          // x data
  public double y[][];                          // y=f(x) data
  public int n[];                          // n : number of data in each line
  public int red[],green[],blue[];                // color code 0-255
  public int xgridon=0;                          // x axis grid line on
  public int ygridon=0;                          // y axis grid line on
  public int xlog=0;                          // x axis logaritmic scale
  public int ylog=0;                          // y axis logaritmic scale
  public int plottype[];
  public String plotst[]={"0 ______",
"1 __ __ ",
"2 . _ . ",
"3 . _ _ .",
"4 - - - ",
"5 bold _____",
"6 bold __ __",
"7 bold . _ . _",
"8 bold . _ _ . .",
"9 bold - - - ",
"10  * " ,
"11  # ",
"12  $ ",
"13  % ",
"14  & ",
"15  - ",
"16  ~ ",
"17  + ",
"18  x ",
"19  = ",
"20  square",
"21  filed square",
"22  circle",
"23  filled circle",
"24  triangle",
"25  diamond",
"26  pentagon",
"27  hexagon",
"28  filled triangle",
"28  filled diamond",
"30  filled pentagon",
"31  filled hexagon",
"32  3 sided star",
"33  4 sided star",
"34  5 sided star",
"35  6 sided star",
"36  filled 3 sided star",
"37  filled 4 sided star",
"38  filled 5 sided star",
"39  filled 6 sided star",
"40  bar graphic",
"41  filled bar graphic"};
  double xminmaxlog,yminmaxlog;
  // plottype = 0 continuous line
  // plottype = 1 dashed line
  // plottype = 2
  // plottype = 3
  // plottype = 10 variable character plot type
  // plottype = 20 rectangle
  // plottype = 21 filled rectangle
  // plottype = 22 circle
  // plottype = 23 filled circle
  // plottype = 24 triangle
  // plottype = 25 diamond
```

```
  // plottype = 26 pentagon
  // plottype = 27 hexagon
  // plottype = 28 filled triangle
  // plottype = 29 filled diamond
  // plottype = 30 filled pentagon
  // plottype = 31 filled hexagon
  // plottype = 32 tri sided star
  // plottype = 33 four sided  star
  // plottype = 34 five sided star
  // plottype = 35 six sided star
  // plottype = 36 filled tri sided  star
  // plottype = 37 filled four sided star
  // plottype = 38 filled five sided star
  // plottype = 39 filled six sided star
  // plottype = 40 bar plot
  // plottype = 41 filled bar plot

  public char ch[];    // plot character (used with plottype 10)
  public int xntic=10;    // number of tics in x axis
  public int yntic=10;    // number of tics in y axis
  BufferedReader fini;
  BufferedReader ffile;
  File plotFile;
  File lineDataFile;
  String fn[];
/*
========== constructors =======================
  public Plot() throws IOException   : Read actual data from file Plot.txt
                                       and plot filenames listed in this file
  public Plot(String pl,String xl,String yl,int xnt,int ynt
  ,int xgo,int ygo,String fn[],int ipt[],int ir[],int ig[]
  ,int ib[]) throws IOException      : Read file names from String fn[] and plot them
  public Plot(String fn[],int ipt[],int ir[],int ig[],int ib[]) throws IOException
  public Plot(String fn[]) throws IOException  : Read file names from String fn[] and plot them
  public Plot(double xi[][],double yi[][],int ipt[],int ir[],int ig[],int ib[]) : read plot data from xi[][] and yi[][]
  public Plot(double xiyi[][]) : read plot data xi[] and yi[] from xiyi[][] string
  public Plot(double xi[][],double yi[][]) : read plot data from xi[][] and yi[][]
  public Plot(double xi[],double yi[]) : read plot data from xi[] and yi[]
  public Plot(f_x f,double xm,double xma,int N,int ipt,int ir,int ig,int ib) : create plot data from a simple function f_x f
  public Plot(f_x f,double xm,double xma,int N) : create plot data from a simple function f_x f
  public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function) : create plot data from an array of simple
functions fi_x f

======== Label settings =======================
  public void setPlabel(String ip) {label=ip;}
  public void setXlabel(String ix) {xlabel=ix;}
  public void setYlabel(String iy) {ylabel=iy;}
  public void setXYlabel(String ix,String iy) {xlabel=ix;ylabel=iy;}

======== Setting additional data after initial construction =========
 This groups's functions are similar to constructor classes
public void addData(double xi[],double yi[])
public void addData(double xi[],double yi[],int iplottype)
public void addData(double xi[],double yi[],int iplottype,int ir,int ig,int ib)
public void addData(double xi[][],double yi[][])
public void addData(double xi[][],double yi[][],int ip[])
public void addData(double xi[][],double yi[][],int ip[],int ir[],int ig[],int ib[])
public void addFunction(f_x f,double xmi,double xma,int N,int ipt)
public void addFunction(f_x f,double xmi,double xma,int N)
public void addFunction(f_x f,double xmi,double xma,int N,int ipt,int ir,int ig,int ib)
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function)
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,int ipt[])
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,

================== Settings of additional plot information =================
----------Plot Type ----------------------
```

```
public void setPlotType(int plot_type[])
public void setPlotType(int dataset,int plot_no)
public void setPlotType(int dataset,char plot_char)
----------x y grid on-off ----------------
public void setXgrid(int igx)
public void setYgrid(int igy)
public void setGrid(int igx,int igy)
public void setXtic(int ixt)
public void setYtic(int iyt)
public void setXYtic(int ixt,int iyt)
public void setColor(int dataset,int ired,int igreen,int iblue)
public void setColor(int ired[],int igreen[],int iblue[])
public void setColor(Color c)
================ Setting of Minimum Maximum fuctions ===================
public void setMinMax()
public void setMinMax(double xi[][],double yi[][])
public void setMinMax(double iminx,double imaxx,double iminy,double imaxy)
=============== Plot Window Formatting ==================================
 public void set_plotwindow(int width,int height,double xip,double yip,double dxp,double dyp)
 public void setabsMaxMin(int ixmin,int iymin,int idx,int idy)
================ Actual Plotting of constructed data
public void plot()
*/

public static double[] data(double xmin,double xmax,double dx)
{
 int n=(int)((xmax-xmin)/dx)+1;
 double x[]=new double[n];
 for(int i=0;i<n-1;i++)
   {x[i]=xmin+dx*i;}
   x[n-1]=xmax;
   return x;
}

public static double[] data(double xmin,double xmax,int n)
{
 double dx=(xmax-xmin)/n;
 double x[]=new double[n];
 for(int i=0;i<n;i++)
   {x[i]=xmin+dx*i;}
   return x;
}

public Plot() throws IOException
{
                label_at_xy=new String[5];
                label_x=new double[5];
                label_y=new double[5];
                for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
                //read input variables from Plot.txt
   //adding max-min prompts and input fields
   //initial values of max and mins
   // Structure of input file
   // filename
   // xlabel
   // ylabel
   // nline
   // datafilename plottype redcolor greencolor bluecolor
   // ........
   // datafilename plottype redcolor greencolor bluecolor
   //
   // A sample data file input :
   //=========================
   // filename
   // xaxislabel
   // yaxislabel
```

```
// 4
// b.dat 22  0 0 0
// a.dat 21 0 255 0
// a.dat 0 0 255 0
// b.dat 0 0 0 255
//==========================
xmin=9.99e50;
xmax=-9.99e50;
ymin=9.99e50;
ymax=-9.99e50;
double xtemp,ytemp;
try{
  fini=new BufferedReader(new FileReader("Plot.txt"));
  } catch(IOException e)
  {
  System.err.println("Error Opening File \n"+e.toString());
  System.exit(1);
  }
label=Text.readStringLine(fini);
xlabel=Text.readStringLine(fini);
ylabel=Text.readStringLine(fini);
nline=Text.readInt(fini);
n=new int[500];
red=new int[500];
green=new int[500];
blue=new int[500];
plottype=new int[500];
ch=new char[500];
ch[10]='*';
ch[11]='#';
ch[12]='$';
ch[13]='%';
ch[14]='&';
ch[15]='@';
ch[16]='~';
ch[17]='+';
ch[18]='x';
ch[19]='=';
//read all data to determine limit values
String fn[];
fn=new String[nline];
int i,j;
for(i=0;i<nline;i++)
{
            plottype[i]=i;
  if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
  if(plottype[i] == 11) ch[i]='#';
  if(plottype[i] == 12) ch[i]='$';
  if(plottype[i] == 13) ch[i]='%';
  if(plottype[i] == 14) ch[i]='&';
  if(plottype[i] == 15) ch[i]='@';
  if(plottype[i] == 16) ch[i]='~';
  if(plottype[i] == 17) ch[i]='+';
  if(plottype[i] == 18) ch[i]='x';
  if(plottype[i] == 19) ch[i]='=';
  fn[i]=Text.readString(fini);
  plottype[i]=Text.readInt(fini);
  if((plottype[i] >= 10)&&(plottype[i] <= 19)) ch[i]=Text.readChar(fini);
  red[i]=Text.readInt(fini);
  green[i]=Text.readInt(fini);
  blue[i]=Text.readInt(fini);
  j=0;
  n[i]=0;
  //open ffile
     try{
  ffile=new BufferedReader(new FileReader(fn[i]));
```

```
      } catch(IOException e)
      {
      System.err.println("Error Opening File \n"+e.toString());
      System.exit(1);
      }
      try{
      while(ffile!=null)
      {
      xtemp=Text.readDouble(ffile);
      ytemp=Text.readDouble(ffile);
      if(xtemp<xmin) xmin=xtemp;
      if(xtemp>xmax) xmax=xtemp;
      if(ytemp<ymin) ymin=ytemp;
      if(ytemp>ymax) ymax=ytemp;
      n[i]++;
      }
        } catch(EOFException e_eof)
        {
        //close ffile
        try{
        ffile.close();
          } catch(IOException e)
          {
          System.err.println("Error Closing File\n"+e.toString());
          System.exit(1);
          }
        } //End of EOFException

      if(i==0) {nmax=n[i];ni=i;}
      else
        {
       if(n[i]>nmax) {nmax=n[i];ni=i;}
        }
    }
//close ffile
    try{
      ffile.close();
      } catch(IOException e)
      {
      System.err.println("Error Closing File\n"+e.toString());
      System.exit(1);
      }
    x=new double[nline][nmax];
    y=new double[nline][nmax];
    //re-read the data again to load it
    for(i=0;i<nline;i++)
    {
      //open ffile
          try{
      ffile=new BufferedReader(new FileReader(fn[i]));
      } catch(IOException e)
      {
      System.err.println("Error Opening File \n"+e.toString());
      System.exit(1);
      }
      for(j=0;j<n[i];j++)
      {
        x[i][j]=Text.readDouble(ffile);
        y[i][j]=Text.readDouble(ffile);

      }
    //close ffile
    try{
      ffile.close();
      } catch(IOException e)
      {
```

```
      System.err.println("Error Closing File\n"+e.toString());
      System.exit(1);
      }
    }
    set_plotwindow(500,400,0.1,0.1,0.9,0.9);
//close fini
    try{
      fini.close();
      } catch(IOException e)
      {
      System.err.println("Error Closing File\n"+e.toString());
      System.exit(1);
      }
  }


  public Plot(String pl,String xl,String yl,int xnt,int ynt
  ,int xgo,int ygo,String fn[],int ipt[],int ir[],int ig[]
  ,int ib[]) throws IOException
   {
   // String pl : plot label
   // String x1 : plot x axis label
   // String y1 : Plot y axis label
              label_at_xy=new String[5];
              label_x=new double[5];
              label_y=new double[5];
              for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
    xmin=9.99e50;
    xmax=-9.99e50;
    ymin=9.99e50;
    ymax=-9.99e50;
    setXtic(10);
    setYtic(10);
    setGrid(0,0);
    double xtemp,ytemp;
    setPlabel(pl);
    setXlabel(xl);
    setYlabel(yl);
    n=new int[500];
    red=new int[500];
    green=new int[500];
    blue=new int[500];
    plottype=new int[500];
    ch=new char[500];
    //read all data to determine limit values
    int i,j;
    nline=fn.length;
    for(i=0;i<nline;i++)
    {
      plottype[i]=ipt[i];
      if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
      if(plottype[i] == 11) ch[i]='#';
      if(plottype[i] == 12) ch[i]='$';
      if(plottype[i] == 13) ch[i]='%';
      if(plottype[i] == 14) ch[i]='&';
      if(plottype[i] == 15) ch[i]='@';
      if(plottype[i] == 16) ch[i]='~';
      if(plottype[i] == 17) ch[i]='+';
      if(plottype[i] == 18) ch[i]='x';
      if(plottype[i] == 19) ch[i]='=';
      red[i]=ir[i];
      green[i]=ig[i];
      blue[i]=ib[i];
      j=0;
      n[i]=0;
      //open ffile
```

```
    System.out.println("isim ="+fn[i]);
        try{
    ffile=new BufferedReader(new FileReader(fn[i]));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    try{
    while(ffile!=null)
    {
    xtemp=Text.readDouble(ffile);
    ytemp=Text.readDouble(ffile);
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[i]++;
    }
      } catch(EOFException e_eof)
      {
      //close ffile
      try{
      ffile.close();
        } catch(IOException e)
        {
        System.err.println("Error Closing File\n"+e.toString());
        System.exit(1);
        }
      } //End of EOFException

     if(i==0) {nmax=n[i];ni=i;}
    else
      {
      if(n[i]>nmax) {nmax=n[i];ni=i;}
      }
  }
  //System.out.println("nline = "+nline+"nmax="+nmax);
  x=new double[nline][nmax];
  y=new double[nline][nmax];
  //re-read the data again to load it
  for(i=0;i<nline;i++)
  {
    //open ffile
        try{
    ffile=new BufferedReader(new FileReader(fn[i]));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    for(j=0;j<n[i];j++)
    {
      x[i][j]=Text.readDouble(ffile);
      y[i][j]=Text.readDouble(ffile);
    }
  //close ffile
  try{
    ffile.close();
    } catch(IOException e)
    {
    System.err.println("Error Closing File\n"+e.toString());
    System.exit(1);
    }

  }
```

```
 set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}

public Plot(String fn[],int ipt[],int ir[],int ig[],int ib[]) throws IOException
 {
 // reading data from file and initilisation
 label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
 xmin=9.99e50;
 xmax=-9.99e50;
 ymin=9.99e50;
 ymax=-9.99e50;
 double xtemp,ytemp;
 n=new int[500];
 red=new int[500];
 green=new int[500];
 blue=new int[500];
 plottype=new int[500];
 ch=new char[500];
 //read all data to determine limit values
 int i,j;
 nline=fn.length;
 for(i=0;i<nline;i++)
 {
   plottype[i]=ipt[i];
   if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
   if(plottype[i] == 11) ch[i]='#';
   if(plottype[i] == 12) ch[i]='$';
   if(plottype[i] == 13) ch[i]='%';
   if(plottype[i] == 14) ch[i]='&';
   if(plottype[i] == 15) ch[i]='@';
   if(plottype[i] == 16) ch[i]='~';
   if(plottype[i] == 17) ch[i]='+';
   if(plottype[i] == 18) ch[i]='x';
   if(plottype[i] == 19) ch[i]='=';
   red[i]=ir[i];
   green[i]=ig[i];
   blue[i]=ib[i];
   j=0;
   n[i]=0;
   //open ffile
   //System.out.println("isim ="+fn[i]);
      try{
   ffile=new BufferedReader(new FileReader(fn[i]));
   } catch(IOException e)
   {
   System.err.println("Error Opening File \n"+e.toString());
   System.exit(1);
   }
   try{
   while(ffile!=null)
   {
   xtemp=Text.readDouble(ffile);
   ytemp=Text.readDouble(ffile);
   if(xtemp<xmin) xmin=xtemp;
   if(xtemp>xmax) xmax=xtemp;
   if(ytemp<ymin) ymin=ytemp;
   if(ytemp>ymax) ymax=ytemp;
   n[i]++;
   }
     } catch(EOFException e_eof)
     {
     //close ffile
     try{
```

```
        ffile.close();
          } catch(IOException e)
          {
          System.err.println("Error Closing File\n"+e.toString());
          System.exit(1);
          }
        } //End of EOFException

      if(i==0) {nmax=n[i];ni=i;}
     else
       {
      if(n[i]>nmax) {nmax=n[i];ni=i;}
       }
  }
  x=new double[nline][nmax];
  y=new double[nline][nmax];
  //re-read the data again to load it
  for(i=0;i<nline;i++)
  {
    //open ffile
        try{
    ffile=new BufferedReader(new FileReader(fn[i]));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    for(j=0;j<n[i];j++)
    {
      x[i][j]=Text.readDouble(ffile);
      y[i][j]=Text.readDouble(ffile);
    }
  //close ffile
  try{
    ffile.close();
    } catch(IOException e)
    {
    System.err.println("Error Closing File\n"+e.toString());
    System.exit(1);
    }
  }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
 }



 public Plot(String fn[]) throws IOException
  {
  //reading data from file(s)
              label_at_xy=new String[5];
              label_x=new double[5];
              label_y=new double[5];
              for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  nline=fn.length;
  int ipt[]=new int[nline];
  int ir[]=new int[nline];
  int ig[]=new int[nline];
  int ib[]=new int[nline];
              xmin=9.99e50;
   xmax=-9.99e50;
   ymin=9.99e50;
   ymax=-9.99e50;
   double xtemp,ytemp;
   n=new int[500];
   red=new int[500];
   green=new int[500];
```

```
blue=new int[500];
plottype=new int[500];
ch=new char[500];
//read all data to determine limit values
int i,j;
nline=fn.length;
for(i=0;i<nline;i++)
{
  plottype[i]=ipt[i];
  if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
  if(plottype[i] == 11) ch[i]='#';
  if(plottype[i] == 12) ch[i]='$';
  if(plottype[i] == 13) ch[i]='%';
  if(plottype[i] == 14) ch[i]='&';
  if(plottype[i] == 15) ch[i]='@';
  if(plottype[i] == 16) ch[i]='~';
  if(plottype[i] == 17) ch[i]='+';
  if(plottype[i] == 18) ch[i]='x';
  if(plottype[i] == 19) ch[i]='=';
  red[i]=ir[i];
  green[i]=ig[i];
  blue[i]=ib[i];
  j=0;
  n[i]=0;
  //open ffile
  //System.out.println("isim ="+fn[i]);
     try{
  ffile=new BufferedReader(new FileReader(fn[i]));
  } catch(IOException e)
  {
  System.err.println("Error Opening File \n"+e.toString());
  System.exit(1);
  }
  try{
  while(ffile!=null)
  {
  xtemp=Text.readDouble(ffile);
  ytemp=Text.readDouble(ffile);
  if(xtemp<xmin) xmin=xtemp;
  if(xtemp>xmax) xmax=xtemp;
  if(ytemp<ymin) ymin=ytemp;
  if(ytemp>ymax) ymax=ytemp;
  n[i]++;
  }
    } catch(EOFException e_eof)
    {
    //close ffile
    try{
    ffile.close();
      } catch(IOException e)
      {
      System.err.println("Error Closing File\n"+e.toString());
      System.exit(1);
      }
    } //End of EOFException

   if(i==0) {nmax=n[i];ni=i;}
  else
    {
   if(n[i]>nmax) {nmax=n[i];ni=i;}
    }
}
x=new double[nline][nmax];
y=new double[nline][nmax];
//re-read the data again to load it
for(i=0;i<nline;i++)
```

```
  {
    //open ffile
        try{
    ffile=new BufferedReader(new FileReader(fn[i]));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    for(j=0;j<n[i];j++)
    {
      x[i][j]=Text.readDouble(ffile);
      y[i][j]=Text.readDouble(ffile);
    }
 //close ffile
 try{
    ffile.close();
    } catch(IOException e)
    {
    System.err.println("Error Closing File\n"+e.toString());
    System.exit(1);
    }
 }
 set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}

public Plot(String fn) throws IOException
{
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
//reading data from file(s)
nline=1;
int ipt[]=new int[nline];
int ir[]=new int[nline];
int ig[]=new int[nline];
int ib[]=new int[nline];
            xmin=9.99e50;
 xmax=-9.99e50;
 ymin=9.99e50;
 ymax=-9.99e50;
 double xtemp,ytemp;
 n=new int[500];
 red=new int[500];
 green=new int[500];
 blue=new int[500];
 plottype=new int[500];
 ch=new char[500];
 //read all data to determine limit values
 int i,j;
 nline=1;
 for(i=0;i<nline;i++)
 {
    plottype[i]=ipt[i];
    if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
    if(plottype[i] == 11) ch[i]='#';
    if(plottype[i] == 12) ch[i]='$';
    if(plottype[i] == 13) ch[i]='%';
    if(plottype[i] == 14) ch[i]='&';
    if(plottype[i] == 15) ch[i]='@';
    if(plottype[i] == 16) ch[i]='~';
    if(plottype[i] == 17) ch[i]='+';
    if(plottype[i] == 18) ch[i]='x';
    if(plottype[i] == 19) ch[i]='=';
    red[i]=ir[i];
```

```
    green[i]=ig[i];
    blue[i]=ib[i];
    j=0;
    n[i]=0;
    //open ffile
    //System.out.println("isim ="+fn[i]);
        try{
    ffile=new BufferedReader(new FileReader(fn));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    try{
    while(ffile!=null)
    {
    xtemp=Text.readDouble(ffile);
    ytemp=Text.readDouble(ffile);
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[i]++;
    }
      } catch(EOFException e_eof)
      {
      //close ffile
      try{
      ffile.close();
        } catch(IOException e)
        {
        System.err.println("Error Closing File\n"+e.toString());
        System.exit(1);
        }
      } //End of EOFException

     if(i==0) {nmax=n[i];ni=i;}
    else
      {
     if(n[i]>nmax) {nmax=n[i];ni=i;}
      }
  }
  x=new double[nline][nmax];
  y=new double[nline][nmax];
  //re-read the data again to load it
  for(i=0;i<nline;i++)
  {
    //open ffile
        try{
    ffile=new BufferedReader(new FileReader(fn));
    } catch(IOException e)
    {
    System.err.println("Error Opening File \n"+e.toString());
    System.exit(1);
    }
    for(j=0;j<n[i];j++)
    {
      x[i][j]=Text.readDouble(ffile);
      y[i][j]=Text.readDouble(ffile);
    }
  //close ffile
  try{
    ffile.close();
    } catch(IOException e)
    {
    System.err.println("Error Closing File\n"+e.toString());
```

```
    System.exit(1);
    }
  }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
 }



public Plot(double xi[][],double yi[][],int ipt[],int ir[],int ig[],int ib[])
{
            pp2(xi,yi,ipt,ir,ig,ib);
}

public void pp2(double xi[][],double yi[][])
{
int ir[];
int ig[];
int ib[];
int ipt[];
int nn=xi.length;
ir=new int[nn];
ig=new int[nn];
ib=new int[nn];
ipt=new int[nn];
for(int i=0;i<nn;i++)
{
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
  ipt[i]=0;
}
pp2(xi,yi,ipt,ir,ig,ib);
}

public void pp2(double xi[][],double yi[][],int ipt[])
{
int ir[];
int ig[];
int ib[];
int nn=xi.length;
ir=new int[nn];
ig=new int[nn];
ib=new int[nn];
for(int i=0;i<nn;i++)
{
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
}
pp2(xi,yi,ipt,ir,ig,ib);
}

public void pp2(double xi[][],double yi[][],int ipt[],int ir[],int ig[],int ib[])
 {
  // reading data from initial array
                        label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  xmin=9.99e50;
  xmax=-9.99e50;
  ymin=9.99e50;
  ymax=-9.99e50;
  double xtemp,ytemp;
  n=new int[500];
  red=new int[500];
```

```
  green=new int[500];
  blue=new int[500];
  plottype=new int[500];
  ch=new char[500];
  ch=new char[500];
  //read all data to determine limit values
  int i,j;
  nline=xi.length;
  for(i=0;i<nline;i++)
  {
    plottype[i]=ipt[i];
    if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
    if(plottype[i] == 11) ch[i]='#';
    if(plottype[i] == 12) ch[i]='$';
    if(plottype[i] == 13) ch[i]='%';
    if(plottype[i] == 14) ch[i]='&';
    if(plottype[i] == 15) ch[i]='@';
    if(plottype[i] == 16) ch[i]='~';
    if(plottype[i] == 17) ch[i]='+';
    if(plottype[i] == 18) ch[i]='x';
    if(plottype[i] == 19) ch[i]='=';
    red[i]=ir[i];
    green[i]=ig[i];
    blue[i]=ib[i];
    nline=xi.length;
    n[i]=0;
    for(j=0;j<xi[i].length;j++)
    {
    xtemp=xi[i][j];
    ytemp=yi[i][j];
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[i]++;
    }
    if(i==0) {nmax=n[i];ni=i;}
    else
      {
      if(n[i]>nmax) {nmax=n[i];ni=i;}
      }
    } //end of for
    x=new double[nline][nmax];
    y=new double[nline][nmax];
    for(i=0;i<nline;i++)
    {
      for(j=0;j<n[i];j++)
      {
      x[i][j]=xi[i][j];
      y[i][j]=yi[i][j];
      }
    }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}

 public void pp2(double ai[][],int ipt[],int ir[],int ig[],int ib[])
 {
  // reading data from initial array
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  xmin=9.99e50;
  xmax=-9.99e50;
  ymin=9.99e50;
  ymax=-9.99e50;
```

```
  double xtemp,ytemp;
  n=new int[500];
  red=new int[500];
  green=new int[500];
  blue=new int[500];
  plottype=new int[500];
  ch=new char[500];
  ch=new char[500];
  //read all data to determine limit values
  int i,j;
  nline=ai.length-1;
  double xi[][]=new double[nline][ai[0].length];
  double yi[][]=new double[nline][ai[0].length];
  for(i=0;i<nline;i++)
  {
    plottype[i]=ipt[i];
    if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
    if(plottype[i] == 11) ch[i]='#';
    if(plottype[i] == 12) ch[i]='$';
    if(plottype[i] == 13) ch[i]='%';
    if(plottype[i] == 14) ch[i]='&';
    if(plottype[i] == 15) ch[i]='@';
    if(plottype[i] == 16) ch[i]='~';
    if(plottype[i] == 17) ch[i]='+';
    if(plottype[i] == 18) ch[i]='x';
    if(plottype[i] == 19) ch[i]='=';
    red[i]=ir[i];
    green[i]=ig[i];
    blue[i]=ib[i];
    nline=xi.length;
    n[i]=0;
    for(j=0;j<xi[i].length;j++)
    {
    xtemp=ai[0][j];
    ytemp=ai[i+1][j];
    xi[i][j]=xtemp;
    yi[i][j]=ytemp;
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[i]++;
    }
    if(i==0) {nmax=n[i];ni=i;}
    else
      {
      if(n[i]>nmax) {nmax=n[i];ni=i;}
      }
    } //end of for
    x=new double[nline][nmax];
    y=new double[nline][nmax];
    for(i=0;i<nline;i++)
    {
      for(j=0;j<n[i];j++)
      {
      x[i][j]=xi[i][j];
      y[i][j]=yi[i][j];
      }
    }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}

public void pp2(double ai[][],int ipt[])
{
  int ir[];
int ig[];
```

```
int ib[];
int nn=ai.length-1;
ir=new int[nn];
ig=new int[nn];
ib=new int[nn];
for(int i=0;i<nn;i++)
{
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
}
pp2(ai,ipt,ir,ig,ib);
}

public void pp2(double ai[][])
{
int ir[];
int ig[];
int ib[];
int ipt[];
int nn=ai.length-1;
ir=new int[nn];
ig=new int[nn];
ib=new int[nn];
ipt=new int[nn];
for(int i=0;i<nn;i++)
{
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
}
pp2(ai,ipt,ir,ig,ib);
}

public Plot(double ai[][])
{
pp2(ai);
}

public Plot(double xi[][],double yi[][])
{
pp2(xi,yi);
}


public void pp1( double xi[],double yi[])
{
pp1(xi,yi,0,0,0,0);
}


public void pp1( double xi[],double yi[],int ipt)
{
pp1(xi,yi,ipt,0,0,0);
}

public void pp1( double xi[],double yi[],int ipt,int ir,int ig,int ib)
{
  // reading data from initial array
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  xmin=9.99e50;
  xmax=-9.99e50;
  ymin=9.99e50;
```

```
  ymax=-9.99e50;
  double xtemp,ytemp;
  n=new int[500];
  red=new int[500];
  green=new int[500];
  blue=new int[500];
  plottype=new int[500];
  ch=new char[500];
  //read all data to determine limit values
  int i,j;
  nline=1;
  plottype[0]=ipt;
  red[0]=ir;
  green[0]=ig;
  blue[0]=ib;
  nline=1;
  n[0]=0;
    plottype[0]=ipt;
    if(plottype[0] == 10) ch[0]='*';//you can change character set by using setPlotType(0,'%');
    if(plottype[0] == 11) ch[0]='#';
    if(plottype[0] == 12) ch[0]='$';
    if(plottype[0] == 13) ch[0]='%';
    if(plottype[0] == 14) ch[0]='&';
    if(plottype[0] == 15) ch[0]='£';
    if(plottype[0] == 16) ch[0]='~';
    if(plottype[0] == 17) ch[0]='+';
    if(plottype[0] == 18) ch[0]='^';
    if(plottype[0] == 19) ch[0]='=';
    for(j=0;j<xi.length;j++)
    {
    xtemp=xi[j];
    ytemp=yi[j];
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[0]++;
    }
    nmax=n[0];ni=0;
    x=new double[nline][nmax];
    y=new double[nline][nmax];
    i=0;
      for(j=0;j<n[0];j++)
      {
      x[i][j]=xi[j];
      y[i][j]=yi[j];
      }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}

public void pp1( double yi[],int ipt,int ir,int ig,int ib)
{
  // reading data from initial array
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  xmin=9.99e50;
  xmax=-9.99e50;
  ymin=9.99e50;
  ymax=-9.99e50;
  double xtemp,ytemp;
  n=new int[500];
  n[0]=yi.length;
  double xi[]=new double[n[0]];
  for(int i=0;i<n[0];i++)
```

```
  {xi[i]=i+1;}
  red=new int[500];
  green=new int[500];
  blue=new int[500];
  plottype=new int[500];
  ch=new char[500];
  //read all data to determine limit values
  int i,j;
  nline=1;
  plottype[0]=ipt;
  red[0]=ir;
  green[0]=ig;
  blue[0]=ib;
  nline=1;
  n[0]=0;
    plottype[0]=ipt;
    if(plottype[0] == 10) ch[0]='*';//you can change character set by using setPlotType(0,'%');
    if(plottype[0] == 11) ch[0]='#';
    if(plottype[0] == 12) ch[0]='$';
    if(plottype[0] == 13) ch[0]='%';
    if(plottype[0] == 14) ch[0]='&';
    if(plottype[0] == 15) ch[0]='£';
    if(plottype[0] == 16) ch[0]='~';
    if(plottype[0] == 17) ch[0]='+';
    if(plottype[0] == 18) ch[0]='^';
    if(plottype[0] == 19) ch[0]='=';
    for(j=0;j<xi.length;j++)
    {
    xtemp=xi[j];
    ytemp=yi[j];
    if(xtemp<xmin) xmin=xtemp;
    if(xtemp>xmax) xmax=xtemp;
    if(ytemp<ymin) ymin=ytemp;
    if(ytemp>ymax) ymax=ytemp;
    n[0]++;
    }
    nmax=n[0];ni=0;
    x=new double[nline][nmax];
    y=new double[nline][nmax];
    i=0;
      for(j=0;j<n[0];j++)
      {
      x[i][j]=xi[j];
      y[i][j]=yi[j];
      }
  set_plotwindow(500,500,0.1,0.1,0.9,0.9);
}
public void pp1( double yi[],int ipt)
{pp1(yi,ipt,0,0,0);}

public void pp1( double yi[])
{pp1(yi,0,0,0,0);}

public Plot(double xi[],double yi[])
{
  pp1(xi,yi);
}

public Plot(double yi[])
 {
  pp1(yi);
}
 public Plot(double yi[],int ipt)
 {
  pp1(yi,ipt);
 }
```

```
public Plot(double yi[],int ipt,int ir,int ig,int ib)
 {
  pp1(yi,ipt,ir,ig,ib);
 }

public void pp10(f_xj f,double xmin,double xmax,double ymin,double ymax,int Nx,int Ny, int Nz,int ipt,int ir,int ig,int ib)
 {
 double z[][]=new double[Nx][Ny];
 double Zc[]=new double[Nz];
 double xc[][];
 double yc[][];
 double zmax=-9e99;
 double zmin=9e99;
 double xx=0;
 double yy=0;
 double dx=(xmax-xmin)/Nx;
 double dy=(ymax-ymin)/Ny;
 double x[]=new double[2];
 for(int i=0;i<Nx;i++)
 {for(int j=0;j<Ny;j++)
   {x[0]=xmin+dx*i;x[1]=ymin+dy*j;z[i][j]=f.func(x);
    if(z[i][j]<zmin) zmin=z[i][j];
    if(z[i][j]>zmax) zmax=z[i][j];
   }
 }
 double dz=(zmax-zmin)/Nz;
  for(int i=0;i<Nz;i++)
  {Zc[i]=zmin+i*dz;}


 }

public void pp3(f_x f,double xm,double xma,int N,int ipt,int ir,int ig,int ib)
 {
  // reading data from a given function//one data set only
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
  xmin=xm;
  xmax=xma;
  nmax=N;
  ni=0;
  ymin=9.99e50;
  ymax=-9.99e50;
  double xtemp,ytemp;
  n=new int[500];
  red=new int[500];
  green=new int[500];
  blue=new int[500];
  plottype=new int[500];
  ch=new char[500];
  double xi[]=new double[N];
  double yi[]=new double[N];
  double dx=(xmax-xmin)/(N-1);
  double xx=xmin;
  for(int j=0;j<N;j++)
  {
            xi[j]=xx;
            yi[j]=f.func(xx);
            xx+=dx;
  }
  pp1(xi,yi,ipt,ir,ig,ib);
}

public Plot(f_x f,double xm,double xma,int N,int ipt,int ir,int ig,int ib)
```

```
{
pp3(f,xm,xma,N,ipt,ir,ig,ib);
}

public Plot(f_x f,double xm,double xma,int N,int ipt)
{
 // reading data from a given function//one data set only
            int ir=0;
            int ig=0;
            int ib=0;
 pp3(f,xm,xma,N,ipt,ir,ig,ib);
}

public Plot(f_x f,double xm,double xma,int N)
 {
 // reading data from a given function//one data set only
            int ipt=0;
            int ir=0;
            int ig=0;
            int ib=0;
 pp3(f,xm,xma,N,ipt,ir,ig,ib);
}

 public Plot(f_x f,double xm,double xma)
 {
 // reading data from a given function//one data set only
            int ipt=0;
            int ir=0;
            int ig=0;
            int ib=0;
            int N=400;
 pp3(f,xm,xma,N,ipt,ir,ig,ib);
}
 public void pp4(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function,int ipt[],int ir[],int ig[],int ib[])
 {
 // reading data from a given functon set
 // same xmin xmax and Number_of_data for each function
 nline=Number_of_Function;
            label_at_xy=new String[5];
            label_x=new double[5];
            label_y=new double[5];
            for(int i=0;i<5;i++) {label_at_xy[i]="";label_x[i]=0.0;label_y[i]=0.0;}
            xmin=xm;
 xmax=xma;
 ymin=9.99e50;
 ymax=-9.99e50;
 double xtemp,ytemp;
 n=new int[500];
 red=new int[500];
 green=new int[500];
 blue=new int[500];
 plottype=new int[500];
 ch=new char[500];
 int N=Number_of_Data;
 double dx=(xmax-xmin)/N;
 double xx=xmin;
 int NF=Number_of_Function;
 nline=NF;
 double xi[][]=new double[NF][N];
 double yi[][]=new double[NF][N];
 double yy[]=new double[NF];
 for(int j=0;j<N;j++)
 {
            yy=f.func(xx);
            for(int i=0;i<NF;i++)
              {  yi[i][j]=yy[i];
```

```
                    xi[i][j]=xx;}
                xx+=dx;
  }
  pp2(xi,yi,ipt,ir,ig,ib);
}

 public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function,
 int ipt[],int ir[],int ig[],int ib[])
 {
 pp4(f,xm,xma,Number_of_Data,Number_of_Function,ipt,ir,ig,ib);
 }

 public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function,
 int ipt[])
 {
 int ir[];
 int ig[];
 int ib[];
 int nn=Number_of_Function;
 ir=new int[nn];
 ig=new int[nn];
 ib=new int[nn];
 for(int i=0;i<nn;i++)
 {
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
 }
 pp4(f,xm,xma,Number_of_Data,Number_of_Function,ipt,ir,ig,ib);
 }

 public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function)
 {
 int ir[];
 int ig[];
 int ib[];
 int ipt[];
 int nn=Number_of_Function;
 ir=new int[nn];
 ig=new int[nn];
 ib=new int[nn];
 ipt=new int[nn];
 for(int i=0;i<nn;i++)
 {
  ir[i]=0;
  ig[i]=0;
  ib[i]=0;
  ipt[i]=0;
 }
 pp4(f,xm,xma,Number_of_Data,Number_of_Function,ipt,ir,ig,ib);
 }

public void setPlabel(String ip) {label=ip;}
public void setXlabel(String ix) {xlabel=ix;}
public void setYlabel(String iy) {ylabel=iy;}
public void setXYlabel(String ix,String iy) {xlabel=ix;ylabel=iy;}

public void set_plotwindow(int width,int height,double xip,double yip,double dxp,double dyp)
{
  //sets rectangular plot window
  //function scale input variables :
  //real scale plot window width
  //real scale plot window height
  //xip: minimum real scale x
  //yip : minimum real scale y
  //dxp: real scale x width  xmax=xip+dxp
```

```
   //dyp: real scale y height ymax=yip+dyp

   // xabsmin : absolute (pixel) scale minimum x
   // yabsmin : absolute (pixel) scale minimum y
   //abswidth : absolute (pixel) scale x width (maximum x=minimum x+abswidth)
   //absheight: absolute (pixel) scale y height (maximum y=minimum y+absheight)

   xabsmin   =  (int)(width*(xip+0.2));
   yabsmin   =  (int)(height*(yip+0.2));
   abswidth  =  (int)(width*dxp);
   absheight =  (int)(height*dyp);
 }

public void setabsMaxMin(int ixmin,int iymin,int idx,int idy)
{
xabsmin=ixmin;
yabsmin=iymin;
abswidth=idx;
absheight=idy;
}

public void addData(String fn)
{ //adds data from file
 double x[];
 double y[];
 double a[][]=Text.readDoubleT(fn);
 x=a[0];
 y=a[1];
 addData(x,y);
}

public void addData(double xi[],double yi[])
{
addData(xi,yi,0,0,0,0);
}

public void addData(double ai[][])
{
pp2(ai);
}

public void addData(double ai[][],int iplottype[],int ir[],int ig[],int ib[])
{pp2(ai,iplottype,ir,ig,ib);}

public void addData(double ai[][],int iplottype[])
{pp2(ai,iplottype);}

public void addData(double xi[],double yi[],int iplottype)
{
addData(xi,yi,iplottype,0,0,0);
}

public void addData(double xi[],double yi[],int iplottype,int ir,int ig,int ib)
{
     int oldnmax=nmax;
     int oldnline=nline;
     nline+=1;
                n[oldnline]=xi.length;
                plottype[nline-1]=iplottype;
                if(plottype[nline-1] == 10) ch[nline-1]='*';//you can change character set by using setPlotType(0,'%');
     if(plottype[nline-1] == 11) ch[nline-1]='#';
     if(plottype[nline-1] == 12) ch[nline-1]='$';
     if(plottype[nline-1] == 13) ch[nline-1]='%';
     if(plottype[nline-1] == 14) ch[nline-1]='&';
     if(plottype[nline-1] == 15) ch[nline-1]='-';
     if(plottype[nline-1] == 16) ch[nline-1]='~';
```

```
        if(plottype[nline-1] == 17) ch[nline-1]='+';
        if(plottype[nline-1] == 18) ch[nline-1]='^';
        if(plottype[nline-1] == 19) ch[nline-1]='=';
                red[nline-1]=ir;
                green[nline-1]=ig;
                blue[nline-1]=ib;
                if(xi.length>nmax)
        {
        nmax=xi.length;
        ni=nline;
                }
                double xa[][]=new double[nline][nmax];
                double ya[][]=new double[nline][nmax];
        for(int i=0;i<oldnline;i++)
        {
                  for(int j=0;j<n[i];j++)
          {
          xa[i][j]=x[i][j];
          ya[i][j]=y[i][j];
          }
        }
        for(int j=0;j<n[nline-1];j++)
          {
          xa[nline-1][j]=xi[j];
          ya[nline-1][j]=yi[j];
          }
        x=xa;
        y=ya;
        setMinMax();
}

public void addData(double xi[][],double yi[][])
{
int ip[]=new int[0];
int ir[]=new int[0];
int ig[]=new int[0];
int ib[]=new int[0];
addData(xi,yi,ip,ir,ig,ib);
}

public void addData(double xi[][],double yi[][],int ip[])
{
int ir[]=new int[0];
int ig[]=new int[0];
int ib[]=new int[0];
addData(xi,yi,ip,ir,ig,ib);
}

public void addData(double xi[][],double yi[][],int ip[],int ir[],int ig[],int ib[])
 {
        int oldnmax=nmax;
        int oldnline=nline;
        nline+=xi.length;
                for(int i=oldnline;i<nline;i++)
                {
                plottype[i]=ip[i-oldnline];
                if(plottype[i] == 10) ch[i]='*';//you can change character set by using setPlotType(0,'%');
        if(plottype[i] == 11) ch[i]='#';
        if(plottype[i] == 12) ch[i]='$';
        if(plottype[i] == 13) ch[i]='%';
        if(plottype[i] == 14) ch[i]='&';
        if(plottype[i] == 15) ch[i]='£';
        if(plottype[i] == 16) ch[i]='~';
        if(plottype[i] == 17) ch[i]='+';
        if(plottype[i] == 18) ch[i]='^';
        if(plottype[i] == 19) ch[i]='=';
```

```
                red[i]=ir[i-oldnline];
                green[i]=ig[i-oldnline];
                blue[i]=ib[i-oldnline];
                n[i]=xi[i-oldnline].length;
                if(xi[i-oldnline].length>nmax)
        {
        nmax=xi[i-oldnline].length;
        ni=i;
                }
        }
                double xa[][]=new double[nline][nmax];
                double ya[][]=new double[nline][nmax];
        for(int i=0;i<oldnline;i++)
        {
                  for(int j=0;j<n[i];j++)
          {
          xa[i][j]=x[i][j];
          ya[i][j]=y[i][j];
          }
        }
        int k=0;
        for(int i=oldnline;i<nline;i++)
        {
        for(int j=0;j<n[i];j++)
          {
          xa[i][j]=xi[i-oldnline][j];
          ya[i][j]=yi[i-oldnline][j];
          }
        }
        x=xa;
        y=ya;
        setMinMax();
}


public void addFunction(f_x f,double xmi,double xma,int N,int ipt)
{
addFunction(f,xmi,xma,N,ipt,0,0,0);
}

public void addFunction(f_x f,double xmi,double xma,int N)
{
addFunction(f,xmi,xma,N,0);
}

public void addFunction(f_x f,double xmi,double xma)
{
int N=400;
addFunction(f,xmi,xma,N,0);
}

public void addFunction(f_x f,double xmi,double xma,int N,int ipt,int ir,int ig,int ib)
{
double xi[]=new double[N];
double yi[]=new double[N];
double dx=(xma-xmi)/N;
double xx=xmi;
for(int j=0;j<N;j++)
{
                xi[j]=xx;
                yi[j]=f.func(xx);
                xx+=dx;
}
addData(xi,yi,ipt,ir,ig,ib);
}
```

```
public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function)
{
int ipt[]=new int[Number_of_Function];
int ir[]=new int[Number_of_Function];
int ig[]=new int[Number_of_Function];
int ib[]=new int[Number_of_Function];
addFunction(f,xmin,xmax,Number_of_Data,Number_of_Function,ipt,ir,ig,ib);
}

public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,int ipt[])
{
int ir[]=new int[Number_of_Function];
int ig[]=new int[Number_of_Function];
int ib[]=new int[Number_of_Function];
addFunction(f,xmin,xmax,Number_of_Data,Number_of_Function,ipt,ir,ig,ib);
}

public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,
  int ipt[],int ir[],int ig[],int ib[])
{
   int N=Number_of_Data;
   double dx=(xmax-xmin)/N;
   double xx=xmin;
   int NF=Number_of_Function;
   double xi[][]=new double[NF][N];
   double yi[][]=new double[NF][N];
   double yy[]=new double[NF];
   for(int j=0;j<N;j++)
   {
                yy=f.func(xx);
                for(int i=0;i<NF;i++)
                  {  yi[i][j]=yy[i];
                     xi[i][j]=xx;}
                xx+=dx;
   }
addData(xi,yi,ipt,ir,ig,ib);
}


public void setPlotType(int plot_type[])
{
for(int i=0;i<plot_type.length;i++)
  {
  setPlotType(i,plot_type[i]);
  }
setMinMax();
}

public void setXlogScaleOn()
{xlog=1;xminmaxlog=Math.log10(xmax-xmin+10.0)-1.0;}

public void setYlogScaleOn()
{ylog=1;yminmaxlog=Math.log10(ymax-ymin+10.0)-1.0;}

public void setXlogScaleOff()
{xlog=0;}

public void setYlogScaleOff()
{ylog=0;}

public void setPlotType(int dataset,int plot_no)
{
  // plottype = 0 continuous line
  // plottype = 1 dashed line
  // plottype = 2
  // plottype = 3
```

```
  // plottype = 10 variable character plot type
        // plottype  = 10 ch[i]='*';//you can change character set by using setPlotType(0,'%');
        // plottype  = 11 ch[i]='#'
        // plottype  = 12 ch[i]='$'
        // plottype  = 13 ch[i]='%'
        // plottype  = 14 ch[i]='&'
        // plottype  = 15 ch[i]='£'
        // plottype  = 16 ch[i]='~'
        // plottype  = 17 ch[i]='+'
        //plottype   = 18 ch[i]='^'
        //plottype   = 19 ch[i]='='
  // plottype = 20 rectangle
  // plottype = 21 filled rectangle
  // plottype = 22 circle
  // plottype = 23 filled circle
  // plottype = 24 triangle
  // plottype = 25 diamond
  // plottype = 26 pentagon
  // plottype = 27 hexagon
  // plottype = 28 filled triangle
  // plottype = 29 filled diamond
  // plottype = 30 filled pentagon
  // plottype = 31 filled hexagon
  // plottype = 32 triangle star
  // plottype = 33 diamond star
  // plottype = 34 pentagon star
  // plottype = 35 hexagon star
  // plottype = 36 filled triangle star
  // plottype = 37 filled diamond star
  // plottype = 38 filled pentagon star
  // plottype = 39 filled hexagon star
  // plottype = 40 bar plot
  // plottype = 41 filled bar plot
  plottype[dataset]=plot_no;
  if(plot_no==10) ch[dataset]='*';   //user defined char is defined as *
}

public void setPlotType(int dataset,char plot_char)
{
  // plottype = 10 user defined character plot
  plottype[dataset]=10;
  ch[dataset]=plot_char;
}


public void setXgrid(int igx)
{
  xgridon=igx;
}

public void setYgrid(int igy)
{
  ygridon=igy;
}

public void setGrid(int igx,int igy)
{
 xgridon=igx;
 ygridon=igy;
}

public void setXtic(int ixt)
{
  xntic=ixt;
}
```

```
public void setYtic(int iyt)
{
  yntic=iyt;
}

public void setXYtic(int ixt,int iyt)
{
    xntic=ixt;
                yntic=iyt;
}

public void setColor(int dataset,int ired,int igreen,int iblue)
{
 red[dataset]=ired;
 green[dataset]=igreen;
 blue[dataset]=iblue;
}

public void setColor(int dataset,Color x)
{
 red[dataset]=x.getRed();
 green[dataset]=x.getGreen();
 blue[dataset]=x.getBlue();
}

public void setColor(int ired[],int igreen[],int iblue[])
{
  for(int i=0;i<nline;i++)
  {
   red[i]=ired[i];
   green[i]=igreen[i];
   blue[i]=iblue[i];
  }
}


public void setColor(Color x[])
{
  for(int i=0;i<nline;i++)
  {
   red[i]=x[i].getRed();
   green[i]=x[i].getGreen();
   blue[i]=x[i].getBlue();
  }
}

public void setMinMax()
{
                xmin=9.99e50;
    xmax=-9.99e50;
    ymin=9.99e50;
    ymax=-9.99e50;
    for(int i=0;i<nline;i++)
       {
                    for(int j=0;j<n[i];j++)
          {
          if(x[i][j]>xmax) xmax=x[i][j];
          if(x[i][j]<xmin) xmin=x[i][j];
          if(y[i][j]>ymax) ymax=y[i][j];
          if(y[i][j]<ymin) ymin=y[i][j];
          }
       }
}

public void setMinMax(double xi[][],double yi[][])
{
```

```
    xmin=9.99e50;
    xmax=-9.99e50;
    ymin=9.99e50;
    ymax=-9.99e50;
    for(int i=0;i<nline;i++)
        {
                if(xi[i].length>nmax) {nmax=xi[i].length;ni=i;}
                for(int j=0;j<xi[i].length;j++)
        {
        if(xi[i][j]>xmax) xmax=xi[i][j];
        if(xi[i][j]<xmin) xmin=xi[i][j];
        if(yi[i][j]>ymax) ymax=yi[i][j];
        if(yi[i][j]<ymin) ymin=yi[i][j];
        }
      }
}

public void setMinMax(double iminx,double imaxx,double iminy,double imaxy)
{
xmin=iminx;
ymin=iminy;
xmax=imaxx;
ymax=imaxy;
}

public void plot()
{
PlotW.plot(this);
}

//==== B-Spline intrepolation=============
  public static double[] hi(double ti[])
  {
  int n=ti.length-1;//ti 0..n
  int nn=n+2;
  double h[]=new double[nn];
  for(int i=1;i<=n;i++)
     {h[i]=ti[i]-ti[i-1];
     }
  h[0]=h[1];
  h[n+1]=h[n];
  return h;
  }

  public static double [] B_Spline(double ti[],double yi[])
  {
  int n=ti.length-1;
  int nn=n+2;
  int i;
  double delta,gamma,p,q,r;
  double a[]=new double[nn];
  double h[]=new double[nn];
  h=hi(ti);
  delta=-1;
  gamma=2.0*yi[0];
  p=delta*gamma;
  q=2.0;
  for(i=1;i<=n;i++)
  {r=h[i+1]/h[i];
  delta=-r*delta;
  gamma=-r*gamma+(r+1)*yi[i];
  p=p+gamma*delta;
  q=q+delta*delta;
  }
  a[0]=-p/q;
  for(i=1;i<=(n+1);i++)
```

```
 {a[i]=((h[i-1]+h[i])*yi[i-1]-h[i]*a[i-1])/h[i-1];}
 return a;
}

 public static double funcB_Spline(double ti[],double a[],double x)
 {
 double h[]=hi(ti);
 int i;
 int n=ti.length-1;
 double nn=n+2;
 double d,e;
 for(i=(n-1);i>=1;i--)
   {if((x-ti[i])>=0) break;}
 i++;
 d=(a[i+1]*(x-ti[i-1])+a[i]*(ti[i]-x+h[i+1]))/(h[i]+h[i+1]);
 e=(a[i]*(x-ti[i-1]+h[i-1])+a[i-1]*(ti[i-1]-x+h[i]))/(h[i-1]+h[i]);
 double bs=(d*(x-ti[i-1])+e*(ti[i]-x))/h[i];
 return bs;
 }

public static double[][]  funcB_Spline(double ti[],double yi[],int aradegersayisi)
{
            //aradegersayisi:  x--o--o--x--o--o--x zincirinde x deneysel noktalar ise
            // ara değer sayısı 2 dir
            int n=ti.length;
            int nn=(n-1)*(aradegersayisi+1)+1;
            double z[][]=new double[2][nn];
   double A[]=B_Spline(ti,yi);
   double dx=0;
   int k=0;
   int i;
   for(i=0;i<(n-1);i++)
      {  z[0][k]=ti[i];z[1][k]=funcB_Spline(ti,A,z[0][k]);k++;
         for(int j=0;j<aradegersayisi;j++)
            {dx=(ti[i+1]-ti[i])/((double)aradegersayisi+1.0);
         z[0][k]=z[0][k-1]+dx;z[1][k]=funcB_Spline(ti,A,z[0][k]);k++;}
      }
      z[0][k]=ti[i];z[1][k]=funcB_Spline(ti,A,z[0][k]);
   return z;
}
public void addB_Spline(double xi[],double yi[],int nSpline)
{
  addSpline(xi,yi,nSpline,0,0,0,0);
}

public void addB_Spline(double ai[][],int nSpline)
{
  addSpline(ai[0],ai[1],nSpline,0,0,0,0);
}

public void addB_Spline(double xi[],double yi[],int nSpline,int iplottype)
{
  addB_Spline(xi,yi,nSpline,iplottype,0,0,0);
}

public void addB_Spline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
{
   double S[][]=funcB_Spline(xi,yi,aradegersayisi);
   addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of spline function)
   }

public void addB_Spline(int linenumber,int nSpline)
{addB_Spline(linenumber,nSpline,0,0,0,0);}

public void addB_Spline(int linenumber,int nSpline,int iplottype)
{addB_Spline(linenumber,nSpline,iplottype,0,0,0);}
```

```
public void addB_Spline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)
{
 double x2[]=new double[n[linenumber]];
 double y2[]=new double[n[linenumber]];
 if(linenumber<0) linenumber=0;
 else if(linenumber>nline) linenumber=nline;
                    for(int j=0;j<n[linenumber];j++)
        { x2[j]=x[linenumber][j];
          y2[j]=y[linenumber][j];
        }
 double S[][]=funcB_Spline(x2,y2,aradegersayisi);
 addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of spline function)
 }
//==== spline interpolation===============
 public static double [] thomas(double a[][],double r[])
 {
 //
 int n=a.length;
 double f[]=new double[n];
 double e[]=new double[n];
 double g[]=new double[n];
 double x[]=new double[n];
 for(int i=0;i<n;i++)     {f[i]=a[i][i];}
 for(int i=0;i<(n-1);i++) {g[i]=a[i][i+1];}
 for(int i=0;i<(n-1);i++) {e[i+1]=a[i+1][i];}
 for(int k=1;k<n;k++)
    {e[k]=e[k]/f[k-1];
     f[k]=f[k]-e[k]*g[k-1];
    }
 for(int k=1;k<n;k++)
    {r[k]=r[k]-e[k]*r[k-1];
    }
 x[n-1]=r[n-1]/f[n-1];
 for(int k=(n-2);k>=0;k--)
    {x[k]=(r[k]-g[k]*x[k+1])/f[k];}
 return x;
 }

 public static double [] thomas(double f[],double e[],double g[],double r[])
 {
 int n=f.length;
 double x[]=new double[n];
 for(int k=1;k<n;k++)
    {e[k]=e[k]/f[k-1];
     f[k]=f[k]-e[k]*g[k-1];
    }
 for(int k=1;k<n;k++)
    {r[k]=r[k]-e[k]*r[k-1];
    }
 x[n-1]=r[n-1]/f[n-1];
 for(int k=(n-2);k>=0;k--)
    {x[k]=(r[k]-g[k]*x[k+1])/f[k];}
 return x;
 }


 public static double [][] cubic_spline(double xi[],double yi[],double c0,double cn)
 {
 int n=xi.length;
 double h[]=new double[n];
 double w[]=new double[n];
 double f[]=new double[n];
 double e[]=new double[n];
 double g[]=new double[n];
 double d[]=new double[n];
```

```
 double x[]=new double[n];
 double S[][]=new double[4][n];
 int k;
 for(k=0;k<(n-1);k++)
    {h[k]=xi[k+1]-xi[k];
     w[k]=(yi[k+1]-yi[k])/h[k];
                }
 d[0]=c0;
 d[n-1]=cn;
 for(k=1;k<(n-1);k++)
    {d[k]=6.0*(w[k]-w[k-1]);}
 f[0]=1.0;
 f[n-1]=1.0;
 g[0]=0.0;
 g[n-1]=0.0;
 e[0]=0.0;
 e[n-1]=0.0;
 for(k=1;k<(n-1);k++)
 {f[k]=2.0*(h[k]+h[k-1]);e[k]=h[k-1];g[k]=h[k];}
 S[2]=thomas(f,e,g,d);
 S[3]=xi;
 for(k=0;k<(n-1);k++)
   {S[0][k]=(6.*yi[k+1]-h[k]*h[k]*S[2][k+1])/(6.0*h[k]);
    S[1][k]=(6.*yi[k]-h[k]*h[k]*S[2][k])/(6.0*h[k]);
   }
 return S;
 }

 public static double funcSpline(double S[][],double x)
 {
 int n=S[0].length;
 double xx1=0;
 double xx2=0;
 double y=0;
 double hk=0;
 for(int k=0;k<(n-1);k++)
    {if(S[3][k]<=x && x<=S[3][k+1])
     {hk=(S[3][k+1]-S[3][k]);
      xx1=(x-S[3][k]);
      xx2=(S[3][k+1]-x);
      y=S[0][k]*xx1+S[1][k]*xx2+(xx1*xx1*xx1*S[2][k+1]+xx2*xx2*xx2*S[2][k])/(6.0*hk);
      break;
      }
    }
      if(y==0 && S[3][n-2]<=x )
      {
                int k=n-2;
                hk=(S[3][k+1]-S[3][k]);
      xx1=(x-S[3][k]);
      xx2=(S[3][k+1]-x);
      y=S[0][k]*xx1+S[1][k]*xx2+(xx1*xx1*xx1*S[2][k+1]+xx2*xx2*xx2*S[2][k])/(6.0*hk);
      }
 return y;
 }

public static double[][]  funcSpline(double xi[],double yi[],int aradegersayisi)
{
              //aradegersayisi:  x--o--o--x--o--o--x zincirinde x deneysel noktalar ise
              // ara değer sayısı 2 dir
              int n=xi.length;
              int nn=(n-1)*(aradegersayisi+1)+1;
              double z[][]=new double[2][nn];
   double S[][]=cubic_spline(xi,yi,0,0);
   double dx=0;
   int k=0;
   int i;
```

```
    for(i=0;i<(n-1);i++)
      {  z[0][k]=xi[i];z[1][k]=funcSpline(S,z[0][k]);k++;
         for(int j=0;j<aradegersayisi;j++)
            {dx=(xi[i+1]-xi[i])/((double)aradegersayisi+1.0);
         z[0][k]=z[0][k-1]+dx;z[1][k]=funcSpline(S,z[0][k]);k++;}
      }
      z[0][k]=xi[i];z[1][k]=funcSpline(S,z[0][k]);
    return z;
  }


 public void addSpline(double xi[],double yi[],int nSpline)
  {
   addSpline(xi,yi,nSpline,0,0,0,0);
  }

  public void addSpline(double ai[][],int nSpline)
  {
   addSpline(ai[0],ai[1],nSpline,0,0,0,0);
  }

  public void addSpline(double xi[],double yi[],int nSpline,int iplottype)
  {
   addSpline(xi,yi,nSpline,iplottype,0,0,0);
  }

  public void addSpline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
  {
    double S[][]=funcSpline(xi,yi,aradegersayisi);
    addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of spline function)
    }

  public void addSpline(int linenumber,int nSpline)
 {addSpline(linenumber,nSpline,0,0,0,0);}

  public void addSpline(int linenumber,int nSpline,int iplottype)
  {addSpline(linenumber,nSpline,iplottype,0,0,0);}

  public void addSpline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)
  {
   double x2[]=new double[n[linenumber]];
   double y2[]=new double[n[linenumber]];
   if(linenumber<0) linenumber=0;
   else if(linenumber>nline) linenumber=nline;
                    for(int j=0;j<n[linenumber];j++)
          { x2[j]=x[linenumber][j];
            y2[j]=y[linenumber][j];
          }
   double S[][]=funcSpline(x2,y2,aradegersayisi);
   addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of spline function)
   }
//==========orthogonal polynomial interpolation (curve fitting)
 public static double[][] OPEKK(double xi[],double fi[],int m)
 //ortogonal polinom en küçük kareler metodu
 //Referans : A First Course in Numerical Analysis
 // Anthony Ralston,Philip Rabinowitz, Mc Graw Hill ISBN 0-07-051158-6
 //m polinom derecesi xi fi girdi verisi
 {
 int i,j,k;
 int n=xi.length;
 int mp2=n+2;
 int mp1=n+1;
 double p[][]=new double[mp2][n];
 double gamma[]=new double[mp1];
 double beta[]=new double[mp1];
 double omega[]=new double[mp1];
```

```
double alpha[]=new double[mp1];
double b[]=new double[mp1];
double wi[]=new double[n];
double a[][]=new double[3][mp1];
for(i=0;i<n;i++)
  {
   p[1][i]=1.0;
   p[0][i]=0.0;
   wi[i]=1.0;
  }
gamma[0]=0;
for(i=0;i<n;i++)
  {
  gamma[0]+=wi[i];
  }
beta[0]=0.0;
for(j=0;j<m+1;j++)
  {
  omega[j]=0;
  for(i=0;i<n;i++)
    {omega[j]+=wi[i]*fi[i]*p[j+1][i];}
  b[j]=omega[j]/gamma[j];
if( j != m)
    {
    alpha[j+1]=0;
    for(i=0;i<n;i++)
      { alpha[j+1]+=wi[i]*xi[i]*p[j+1][i]*p[j+1][i]/gamma[j];}
    for(i=0;i<n;i++)
      {p[j+2][i]=(xi[i]-alpha[j+1])*p[j+1][i]-beta[j]*p[j][i];}
      gamma[j+1]=0;
    for(i=0;i<n;i++)
      {gamma[j+1]+=wi[i]*p[j+2][i]*p[j+2][i];}
    beta[j+1]=gamma[j+1]/gamma[j];
    }
}//end of j
for(j=0;j<m+1;j++)
  {
  a[0][j]=b[j];
  a[1][j]=alpha[j];
  a[2][j]=beta[j];
  }
return a;
}

public static double funcOPEKK(double a[][],double x)
{
// polinom değerleri hesaplama fonksiyonu
double yy=0;
int k;
int m=a[0].length-1;
int mp2=m+2;
double q[];
q=new double[mp2];
//vector<double> q(m+2,0.0);
for(k=m-1;k>=0;k--)
  {
  q[k]=a[0][k]+(x-a[1][k+1])*q[k+1]-a[2][k+1]*q[k+2];
  yy=q[k];
  }
return yy;
}

public static double[][]  funcOPEKK(double xi[],double yi[],int polinomkatsayisi,int aradegersayisi)
 {
            //aradegersayisi:  x--o--o--x--o--o--x zincirinde x deneysel noktalar ise
            // ara değer sayısı 2 dir
```

```
                int n=xi.length;
                int nn=(n-1)*(aradegersayisi+1)+1;
                double z[][]=new double[2][nn];
    double E[][]=OPEKK(xi,yi,polinomkatsayisi);
    double dx=0;
    int k=0;
    int i;
    for(i=0;i<(n-1);i++)
       {z[0][k]=xi[i];z[1][k]=funcOPEKK(E,z[0][k]);k++;
       for(int j=0;j<aradegersayisi;j++)
       {dx=(xi[i+1]-xi[i])/((double)aradegersayisi+1.0);
       z[0][k]=z[0][k-1]+dx;z[1][k]=funcOPEKK(E,z[0][k]);k++;}
       }
       z[0][k]=xi[i];z[1][k]=funcOPEKK(E,z[0][k]);
    return z;
  }

 public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi)
  {
   addOPEKK(xi,yi,npolinom,aradegersayisi,0,0,0,0);
  }

  public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)
  {
   addOPEKK(xi,yi,npolinom,npolinom,iplottype,0,0,0);
  }

  public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
  {
    double S[][]=funcOPEKK(xi,yi,npolinom,aradegersayisi);
    addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of OPEKK function)
    }

  public void addOPEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype)
  {addOPEKK(linenumber,npolinom,aradegersayisi,iplottype,0,0,0);}

  public void addOPEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
  {
   double x2[]=new double[n[linenumber]];
   double y2[]=new double[n[linenumber]];
   if(linenumber<0) linenumber=0;
   else if(linenumber>nline) linenumber=nline;
                    for(int j=0;j<n[linenumber];j++)
          { x2[j]=x[linenumber][j];
            y2[j]=y[linenumber][j];
          }
   double S[][]=funcOPEKK(x2,y2,npolinom,aradegersayisi);
   addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of OPEKK function)
   }
// en kucuk kareler
  public static double[] pivotlugauss(double a[][],double b[])
  { //kısmi pivotlu gauss eleme yöntemi
  int n=b.length;
  double x[]=new double[n];
  double carpan=0;
  double toplam=0;
  double buyuk;
  double dummy=0;
  //gauss eleme
  int i,j,k,p,ii,jj;
  for(k=0;k<(n-1);k++)
  { //pivotlama
                p=k;
                buyuk=Math.abs(a[k][k]);
                for(ii=k+1;ii<n;ii++)
                {  dummy=Math.abs(a[ii][k]);
```

```
                if(dummy > buyuk) {buyuk=dummy;p=ii;}
              }
              if(p!=k)
              { for(jj=k;jj<n;jj++)
                { dummy=a[p][jj];
                  a[p][jj]=a[k][jj];
                  a[k][jj]=dummy;
                }
                dummy=b[p];
                b[p]=b[k];
                b[k]=dummy;
              }
              //gauss elemeyi çözme
    for(i=k+1;i<n;i++)
    {  carpan=a[i][k]/a[k][k];
       a[i][k]=0;
       for(j=k+1;j<n;j++)
       {   a[i][j]-=carpan*a[k][j]; }
           b[i]   =b[i]   -carpan*b[k];
              }
  }
  //geriye doğru yerine koyma
  x[n-1]=b[n-1]/a[n-1][n-1];
  for(i=n-2;i>=0;i--)
  {
    toplam=0;
    for(j=i+1;j<n;j++)
    {  toplam+=a[i][j]*x[j];}
    x[i]=(b[i]-toplam)/a[i][i];
  }
  return x;
}

public static double[] EKK(double xi[],double yi[],int n)
{
int l=xi.length;
int i,j,k;
int np1=n+1;
double A[][];
A=new double[np1][np1];
double B[];
B=new double[np1];
double X[];
X=new double[np1];
for(i=0;i<n+1;i++)
  {  for(j=0;j<n+1;j++)
     {if(i==0 && j==0) A[i][j]=l;
     else  for(k=0;k<l;k++) A[i][j] += Math.pow(xi[k],(i+j));
     }
     for(k=0;k<l;k++) { if(i==0)  B[i]+= yi[k];
                    else      B[i] += Math.pow(xi[k],i)*yi[k];}
  }
System.out.println(Matrix.toString(A));
System.out.println(Matrix.toStringT(B));
X=pivotlugauss(A,B);
//X=B/A;
double max=0;
for(i=0;i<n+1;i++)
 if(Math.abs(X[i]) > max) max = Math.abs(X[i]);
for(i=0;i<n+1;i++)
 if((Math.abs(X[i]/max) > 0) && (Math.abs(X[i]/max) < 1.0e-100)) X[i]=0;
return X;
}

public static double funcEKK(double e[],double x)
{
```

```
// this function calculates the value of
// least square curve fitting function
int n=e.length;
double ff;
if(n!=0.0)
  { ff=e[n-1];
  for(int i=n-2;i>=0;i--)
   { ff=ff*x+e[i]; }
  }
else
  ff=0;
return ff;
}

public static double hata(double x[],double y[],double e[])
{
//calculates absolute square root error of a least square approach
double n=x.length;
 int k;
 double total=0;
 for(k=0;k<n;k++)
 {
 total+=(y[k]-funcEKK(e,x[k]))*(y[k]-funcEKK(e,x[k]));
 }
total=Math.sqrt(total);
return total;
}

public static double[][]  funcEKK(double xi[],double yi[],int polinomkatsayisi,int aradegersayisi)
 {
                //aradegersayisi:  x--o--o--x--o--o--x zincirinde x deneysel noktalar ise
                // ara değer sayısı 2 dir
                 int n=xi.length;
                int nn=(n-1)*(aradegersayisi+1)+1;
                double z[][]=new double[2][nn];
   double E[]=EKK(xi,yi,polinomkatsayisi);
   double dx=0;
   int k=0;
   int i;
   for(i=0;i<(n-1);i++)
      {z[0][k]=xi[i];z[1][k]=funcEKK(E,z[0][k]);k++;
      for(int j=0;j<aradegersayisi;j++)
      {dx=(xi[i+1]-xi[i])/((double)aradegersayisi+1.0);
      z[0][k]=z[0][k-1]+dx;z[1][k]=funcEKK(E,z[0][k]);k++;}
      }
      z[0][k]=xi[i];z[1][k]=funcEKK(E,z[0][k]);
   return z;
 }
public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi)
 {
  addEKK(xi,yi,npolinom,aradegersayisi,0,0,0,0);
 }

 public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)
 {
  addEKK(xi,yi,npolinom,npolinom,iplottype,0,0,0);
 }

 public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
 {
   double S[][]=funcEKK(xi,yi,npolinom,aradegersayisi);
   addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of EKK function)
   }

 public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype)
 {addEKK(linenumber,npolinom,aradegersayisi,iplottype,0,0,0);}
```

```
 public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)
 {
  double x2[]=new double[n[linenumber]];
  double y2[]=new double[n[linenumber]];
  if(linenumber<0) linenumber=0;
  else if(linenumber>nline) linenumber=nline;
                    for(int j=0;j<n[linenumber];j++)
         { x2[j]=x[linenumber][j];
           y2[j]=y[linenumber][j];
         }
  double S[][]=funcEKK(x2,y2,npolinom,aradegersayisi);
  addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of EKK function)
  }
//=====Lagrange interpolasyon
  public static double[][]  Lagrange(double xi[],double yi[])
  {
                int n=xi.length;
    double L[][]=new double[2][n];
    for(int i=0;i<n;i++)
    {L[0][i]=yi[i];
    for(int j=0;j<n;j++)
    {if(i!=j) L[0][i]/=(xi[i]-xi[j]);}
    }
    for(int i=0;i<n;i++)
    {L[1][i]=xi[i];}
    return L;
  }

  public static double  funcLagrange(double L[][],double x)
  {
                int n=L[0].length;
    double carpim=1.0;
    double toplam=0;
    for(int i=0;i<n;i++)
    {carpim=L[0][i];
    for(int j=0;j<n;j++)
    {if(i!=j) carpim*=(x-L[1][j]);}
    toplam+=carpim;
    }
    return toplam;
  }

 public static double[][]  funcLagrange(double xi[],double yi[],int aradegersayisi)
 {
               //aradegersayisi:  x--o--o--x--o--o--x zincirinde x deneysel noktalar ise
               // ara değer sayısı 2 dir
               int n=xi.length;
               int nn=(n-1)*(aradegersayisi+1)+1;
               double z[][]=new double[2][nn];
   double Q[][]=Lagrange(xi,yi);
   double dx=0;
   int k=0;
   int i;
   for(i=0;i<(n-1);i++)
      {  z[0][k]=xi[i];z[1][k]=funcLagrange(Q,z[0][k]);k++;
         for(int j=0;j<aradegersayisi;j++)
            {dx=(xi[i+1]-xi[i])/((double)aradegersayisi+1.0);
         z[0][k]=z[0][k-1]+dx;z[1][k]=funcLagrange(Q,z[0][k]);k++;}
      }
      z[0][k]=xi[i];z[1][k]=funcLagrange(Q,z[0][k]);
   return z;
 }

public void addLagrange(double xi[],double yi[],int aradegersayisi)
 {
```

```
  addLagrange(xi,yi,aradegersayisi,0,0,0,0);
 }

 public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype)
 {
  addLagrange(xi,yi,aradegersayisi,iplottype,0,0,0);
 }

 public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)
 {
   double S[][]=funcLagrange(xi,yi,aradegersayisi);
   addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of Lagrange function)
   }

 public void addLagrange(int linenumber,int aradegersayisi,int iplottype)
 {addLagrange(linenumber,aradegersayisi,iplottype,0,0,0);}

 public void addLagrange(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)
 {
  double x2[]=new double[n[linenumber]];
  double y2[]=new double[n[linenumber]];
  if(linenumber<0) linenumber=0;
  else if(linenumber>nline) linenumber=nline;
                    for(int j=0;j<n[linenumber];j++)
         { x2[j]=x[linenumber][j];
           y2[j]=y[linenumber][j];
         }
  double S[][]=funcLagrange(x2,y2,aradegersayisi);
  addData(S[0],S[1],iplottype,ir,ig,ib); //one set of additional data (of Lagrange function)
  }

  public void setBackground(Color  b1)
  {bg=b1;}

  public void setBackground(int ir,int ig,int ib)
  {bg=new Color(ir,ig,ib);}
    //Static plot functions
    //===========

    public static Plot func(f_x ff1,double xmin,double xmax,int n,int pnumber)
    {
    Plot pp=new Plot(ff1,xmin,xmax,n,pnumber);
    pp.plot();
    return pp;
    }
    public static Plot func(f_x ff1,double xmin,double xmax,int n)
    {
    Plot pp=new Plot(ff1,xmin,xmax,n,0);
    pp.plot();
    return pp;
    }

    public static Plot func(f_x ff1,double xmin,double xmax)
    {
    Plot pp=new Plot(ff1,xmin,xmax,300,0);
    pp.plot();
    return pp;
    }

    public static Plot func(fi_x ff1,double xmin,double xmax,int n,int pnumber)
    {
    Plot pp=new Plot(ff1,xmin,xmax,n,pnumber);
    pp.plot();
    return pp;
    }
```

```
public static Plot func(fi_x ff1,double xmin,double xmax,int n)
{
Plot pp=new Plot(ff1,xmin,xmax,n,0);
pp.plot();
return pp;
}

public static Plot func(fi_x ff1,double xmin,double xmax)
{
Plot pp=new Plot(ff1,xmin,xmax,300,0);
pp.plot();
return pp;
}

public static Plot func(f_x ff1,f_x ff2,double xmin,double xmax,int n)
{
Plot pp=new Plot(ff1,xmin,xmax,n);
pp.addFunction(ff2,xmin,xmax,n);
pp.plot();
return pp;
}

public static Plot func(f_x ff1,f_x ff2,double xmin,double xmax)
{
Plot pp=new Plot(ff1,xmin,xmax,300);
pp.addFunction(ff2,xmin,xmax,300);
pp.plot();
return pp;
}

public static Plot file(String ff1)throws IOException
{
Plot pp=new Plot(ff1);
pp.plot();
return pp;
}

public static Plot file()throws IOException
{
Plot pp=new Plot();
pp.plot();
return pp;
}

public static Plot file(String ff1,String ff2)throws IOException
{
        String ff3[]={ff1,ff2};
Plot pp=new Plot(ff3);
pp.plot();
return pp;
}

public static Plot file(String ff1,String ff2,String ff3)throws IOException
{
        String ff4[]={ff1,ff2,ff3};
Plot pp=new Plot(ff4);
pp.plot();
return pp;
}

public static Plot file(String ff1,String ff2,String ff4,String ff3)throws IOException
{
        String ff5[]={ff1,ff2,ff3,ff4};
Plot pp=new Plot(ff5);
pp.plot();
return pp;
```

```
    }

    public static Plot file(String ff1[])throws IOException
    {
    Plot pp=new Plot(ff1);
    pp.plot();
    return pp;
    }

    public static Plot data(double x[],double y[])
    {
    Plot pp=new Plot(x,y);
    pp.plot();
    return pp;
    }

    public static Plot data(double x[],double y1[],double y2[])
    {
    Plot pp=new Plot(x,y1);
    pp.addData(x,y2);
    pp.plot();
    return pp;
    }

    public static Plot data(double x[],double y1[],double y2[],double y3[])
    {
    Plot pp=new Plot(x,y1);
    pp.addData(x,y2);
    pp.addData(x,y3);
    pp.plot();
    return pp;
    }

    public static Plot data(double a[][])
    {
    Plot pp=new Plot(a);
    pp.plot();
    return pp;
    }
}
```

PROGRAM 21.1-21 PlotW class

```
//=====================================================
// Numerical Analysis package in java
// Plot package PlotW
// draws the plot window
// Dr. Turhan Coban
// EGE Üniversitesi Mühendislik Fakultesi, Makina Bölümü
// turhan.coban@ege.edu.tr
// ====================================================
/*
 * Swing version.
 */
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.io.*;
```

```
// Plot2D;
// PlotShapesSW;
// PlotW
// BasicWindowMonitor;

public class PlotW extends JFrame implements ItemListener,ActionListener, MouseListener,MouseMotionListener
{
   private String s="";
            //f2 ff=new f2();
            boolean inAnApplet = true;
            BufferedReader fbilgi;
   final static String KONTROLPANEL = "Control page";
   final static String    PLOTPANEL = "Plot page   ";
    final static String   INFOPANEL = "Additional information   ";
   Plot2D jta;
   double xi1,yi1,xi2,yi2,xi3,yi3;
   int n;
   JLabel altbar;         //writing area
   JLabel promptXmin;      // Label prompt in Xmin field
   JLabel promptXmax;      // Label prompt in Xmax field
   JLabel promptYmin;      // Label prompt in Ymin field
   JLabel promptYmax;      // Label prompt in Ymax field
   JLabel promptLabel;     // Label prompt Plot Label
   JLabel promptXLabel;    // Label prompt Plot XLabel
   JLabel promptYLabel;    // Label prompt Plot YLabel
   JLabel promptXntic;      // Label prompt in Xmin field
   JLabel promptYntic;      // Label prompt in Xmax field
   JLabel promptXgridon;    // Label prompt in Ymin field
   JLabel promptYgridon;    // Label prompt in Ymax field
   JLabel promptXlogon;     // x axis log scale on
   JLabel promptYlogon;     // y axis log scale on
   JLabel pr[];
   JTextField inputXmin;    // input field Xmin
   JTextField inputXmax;    // input field Xmax
   JTextField inputYmin;    // input field Ymin
   JTextField inputYmax;    // input field Ymax
   JTextField inputLabel;   // input field Label
   JTextField inputXLabel;  // input field XLabel
   JTextField inputYLabel;  // input field YLabel
   JTextField inputXntic;   // input field xntic
   JTextField inputYntic;   // input field yntic
   JTextArea bilgi;
   JCheckBox inputXgridon;  // input field xgridon
   JCheckBox inputYgridon;  // input field ygridon
   JCheckBox inputXlogon;  // check field xlogon
   JCheckBox inputYlogon;  // check field ylogon
   JLabel promptAdditionalLabel[];
   //to add additional label to the graphic window
   JTextField inputAdditionalLabel[];
   JTextField inputAdditionalXLabel[];
   JTextField inputAdditionalYLabel[];
   JButton printButton;
            JLabel arkarenk1;
            JButton arkarenk2;
   JLabel lab1[];
            JLabel lab2[];
            JLabel xymin[];
   JButton xyset[];
```

```
JComboBox c1[];
JButton col1[];
JButton but1[];
JScrollPane skrolPane;
Color renk1;
Color renk2;

public PlotW()
{
super("Plot çizimi");
   try{Plot pi=new Plot();plotWkur(pi); } catch(IOException e)
  {
  System.err.println("Error Opening File \n"+e.toString());
  System.exit(1);
  }
}

public PlotW(Plot pi)
{
super(pi.label);
plotWkur(pi);
}

public void plotWkur(Plot pi)
{
        renk2=Color.white;
n=pi.nline;
lab1=new JLabel[n];
lab2=new JLabel[6];
c1=new JComboBox[n];
col1=new JButton[n];
xymin=new JLabel[2];
xyset=new JButton[2];
bilgi=new JTextArea();
promptAdditionalLabel=new JLabel[5];
inputAdditionalLabel= new JTextField[5];
inputAdditionalXLabel= new JTextField[5];
inputAdditionalYLabel=new JTextField[5];
arkarenk1=new JLabel("Background color");
arkarenk2=new JButton("");
arkarenk2.setBackground(renk2);
pr=new JLabel[5];
but1=new JButton[5];
for(int i=0;i<5;i++)
{promptAdditionalLabel[i]=new JLabel("additional heading #"+i+" at the x,y coordinates");
inputAdditionalLabel[i]=new JTextField("");
inputAdditionalXLabel[i]=new JTextField("");
inputAdditionalYLabel[i]=new JTextField("");
but1[i]=new JButton("enter x,y coordinate");
}
pr[0]=new JLabel("");
pr[1]=new JLabel("x coordinate");
pr[2]=new JLabel("y coordinate");
pr[3]=new JLabel("Heading");
pr[4]=new JLabel("last clicked point");
xymin[0]=new JLabel("last right click : ");
xymin[1]=new JLabel("last left click  : ");
xyset[0]=new JButton("set as xmin, ymin");
```

```
xyset[1]=new JButton("set as xmax, ymax");
lab2[0]=new JLabel("");
lab2[1]=new JLabel("line type");
lab2[2]=new JLabel("Color selection table");
for(int i=0;i<n;i++)
{
        Color color1=new Color(pi.red[i],pi.green[i],pi.blue[i]);
        lab1[i]=new JLabel("line "+i+" ");
c1[i]=new JComboBox(pi.plotst);
c1[i].setSelectedIndex(pi.plottype[i]);
col1[i]=new JButton(" ");
col1[i].setBackground(color1);
}

Container contentPane = getContentPane();
renk1=contentPane.getBackground();
JTabbedPane tabbedPane = new JTabbedPane();
   promptXmin=new JLabel("Xmin ");
   inputXmin=new JTextField(10);
   promptXmax=new JLabel("Xmax ");
   inputXmax=new JTextField(10);
   promptYmin=new JLabel("Ymin ");
   inputYmin=new JTextField(10);
   promptYmax=new JLabel("Ymax ");
   inputYmax=new JTextField(10);
   //*******
   promptLabel=new  JLabel("Plot label     : ");
   promptXLabel=new JLabel("x axes label : ");
   promptYLabel=new JLabel("y axes label : ");
   inputLabel=new JTextField(40);
   inputXLabel=new JTextField(40);
   inputYLabel=new JTextField(40);
   inputLabel.setBackground(renk1);
   inputXLabel.setBackground(renk1);
   inputYLabel.setBackground(renk1);
   //*******
   promptXntic=new JLabel("X tick no");
   inputXntic=new JTextField(10);
   promptYntic=new  JLabel("Y tick no");
   inputYntic=new JTextField(10);

   promptXgridon=new JLabel("X grid ");
   if(pi.xgridon!=0) inputXgridon=new JCheckBox("   ",true);
   else              inputXgridon=new JCheckBox("   ",false);
   promptYgridon=new JLabel("Y grid ");
   if(pi.ygridon!=0) inputYgridon=new JCheckBox("   ",true);
   else              inputYgridon=new JCheckBox("   ",false);
   promptXlogon=new JLabel("x log scale ");
   promptYlogon=new JLabel("y log scale ");
   if(pi.xlog!=0) inputXlogon=new JCheckBox("   ",true);
   else           inputXlogon=new JCheckBox("   ",false);
   if(pi.ylog!=0) inputYlogon=new JCheckBox("   ",true);
   else           inputYlogon=new JCheckBox("   ",false);
    // check field xlogon
   JPanel pane1 = new JPanel();
   JPanel pane3 = new JPanel();
   JPanel mpane=new JPanel();
   bilgi.setText(bilgioku());
```

```
bilgi.setBackground(renk1);
pane3.add(bilgi);

mpane.setLayout(new GridLayout(7,4));
JPanel p1=new JPanel();

p1.setLayout(new GridLayout(6,5));
inputXmin.setBackground(renk1);
inputXmax.setBackground(renk1);
inputYmin.setBackground(renk1);
inputYmax.setBackground(renk1);
inputXntic.setBackground(renk1);
inputYntic.setBackground(renk1);
mpane.add(xymin[0]);
mpane.add(xyset[0]);
mpane.add(xymin[1]);
mpane.add(xyset[1]);
mpane.add(promptXmin);
mpane.add(inputXmin);
mpane.add(promptXmax);
mpane.add(inputXmax);
mpane.add(promptYmin);
mpane.add(inputYmin);
mpane.add(promptYmax);
mpane.add(inputYmax);
mpane.add(promptXntic);
mpane.add(inputXntic);
mpane.add(promptYntic);
mpane.add(inputYntic);
mpane.add(promptXgridon);
mpane.add(inputXgridon);
mpane.add(promptYgridon);
mpane.add(inputYgridon);
mpane.add(promptXlogon);
mpane.add(inputXlogon);
mpane.add(promptYlogon);
mpane.add(inputYlogon);
mpane.add(arkarenk1);
mpane.add(arkarenk2);
pane1.add(mpane,BorderLayout.NORTH);
JPanel xpane=new JPanel();
xpane.setLayout(new GridLayout((n+1),3));
for(int i=0;i<3;i++)
{lab2[i].setBackground(renk1);
        xpane.add(lab2[i]);}
for(int i=0;i<n;i++)
{
xpane.add(lab1[i]);
xpane.add(c1[i]);
xpane.add(col1[i]);
c1[i].setBackground(renk1);
}
//********
JPanel lpane=new JPanel();
lpane.setLayout(new GridLayout(3,2));
lpane.add(promptLabel);
lpane.add(inputLabel);
lpane.add(promptXLabel);
```

```
        lpane.add(inputXLabel);
        lpane.add(promptYLabel);
        lpane.add(inputYLabel);
        //********
        p1.add(pr[0]);
        p1.add(pr[1]);
        p1.add(pr[2]);
        p1.add(pr[3]);
        p1.add(pr[4]);
        for(int i=0;i<5;i++)
        {inputAdditionalLabel[i].setBackground(renk1);
         inputAdditionalXLabel[i].setBackground(renk1);
         inputAdditionalYLabel[i].setBackground(renk1);
                p1.add(promptAdditionalLabel[i]);
         p1.add(inputAdditionalXLabel[i]);
         p1.add(inputAdditionalYLabel[i]);
         p1.add(inputAdditionalLabel[i]);
         p1.add(but1[i]);
        }
        //lpane.add(p1);
        skrolPane=new
JScrollPane(xpane,JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);
        pane1.add(lpane,BorderLayout.NORTH);
        pane1.add(p1,BorderLayout.CENTER);
        pane1.add(skrolPane,BorderLayout.SOUTH);

        //*********
        for(int i=0;i<5;i++)
        {inputAdditionalLabel[i].addActionListener(this);
        inputAdditionalXLabel[i].addActionListener(this);
        inputAdditionalYLabel[i].addActionListener(this);
        but1[i].addActionListener(this);
        }
        xyset[0].addActionListener(this);
        xyset[1].addActionListener(this);
        for(int i=0;i<n;i++)
        { c1[i].addItemListener(this);
                col1[i].addActionListener(this);
        }
        //**********
        inputXmin.addActionListener(this);
        inputXmax.addActionListener(this);
        inputYmin.addActionListener(this);
        inputYmax.addActionListener(this);
        inputLabel.addActionListener(this);
        inputXLabel.addActionListener(this);
        inputYLabel.addActionListener(this);
        inputXntic.addActionListener(this);
        inputYntic.addActionListener(this);
        inputXlogon.addItemListener(this);
        inputYlogon.addItemListener(this);
        inputXgridon.addItemListener(this);
        inputYgridon.addItemListener(this);
        arkarenk2.addActionListener(this);
        jta=new Plot2D(pi);
        inputXmin.setText(Double.toString(jta.p1.xmin));
        inputXmax.setText(Double.toString(jta.p1.xmax));
        inputYmin.setText(Double.toString(jta.p1.ymin));
```

```
        inputYmax.setText(Double.toString(jta.p1.ymax));
        inputXntic.setText(Integer.toString(jta.p1.xntic));
        inputYntic.setText(Integer.toString(jta.p1.yntic));
        inputLabel.setText(jta.p1.label);
        inputXLabel.setText(jta.p1.xlabel);
        inputYLabel.setText(jta.p1.ylabel);

        JPanel pane2 = new JPanel();
        pane2.setLayout(new BorderLayout());
        altbar=new JLabel();
        pane2.add(altbar,BorderLayout.SOUTH);
        pane2.addMouseListener(this);
        pane2.addMouseMotionListener(this);
        pane2.add(jta);
        tabbedPane.addTab(PLOTPANEL,    pane2);
        tabbedPane.addTab(KONTROLPANEL, pane1);
        tabbedPane.addTab(INFOPANEL,new
JScrollPane(pane3,JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,JScrollPane.HORIZONTAL_SCROLLBAR_NEVER));
        contentPane.add(  tabbedPane, BorderLayout.CENTER);
              }

    public void itemStateChanged(ItemEvent e)
    {
    inputXmin.setText(Double.toString(jta.p1.xmin));
    inputXmax.setText(Double.toString(jta.p1.xmax));
    inputYmin.setText(Double.toString(jta.p1.ymin));
    inputYmax.setText(Double.toString(jta.p1.ymax));
    inputXntic.setText(Integer.toString(jta.p1.xntic));
    inputYntic.setText(Integer.toString(jta.p1.yntic));
    Object source=e.getItemSelectable();

    if(source==inputXgridon)
    {
        if (e.getStateChange() == ItemEvent.DESELECTED)
        {jta.p1.xgridon=0;}
        else
        {jta.p1.xgridon=1;}
    }
    else if(source==inputYgridon)
    {
        if (e.getStateChange() == ItemEvent.DESELECTED)
        {jta.p1.ygridon=0;}
        else
        {jta.p1.ygridon=1;}
    }
    else if(source==inputXlogon)
    {
        if (e.getStateChange() == ItemEvent.DESELECTED)
        {jta.p1.setXlogScaleOff();}
        else
        {jta.p1.setXlogScaleOn();}
    }
    else if(source==inputYlogon)
    {
       if (e.getStateChange() == ItemEvent.DESELECTED)
       {jta.p1.setYlogScaleOff();}
       else
       {jta.p1.setYlogScaleOn();}
```

```
    }
    else
    {
    for(int i=0;i<n;i++)
    {   if(source==c1[i])
                {jta.p1.plottype[i]=c1[i].getSelectedIndex();
                 if(jta.p1.plottype[i] == 10) jta.p1.ch[i]='*';//you can change character set by using setPlotType(0,'%');
        if(jta.p1.plottype[i] == 11) jta.p1.ch[i]='#';
        if(jta.p1.plottype[i] == 12) jta.p1.ch[i]='$';
        if(jta.p1.plottype[i] == 13) jta.p1.ch[i]='%';
        if(jta.p1.plottype[i] == 14) jta.p1.ch[i]='&';
        if(jta.p1.plottype[i] == 15) jta.p1.ch[i]='-';
        if(jta.p1.plottype[i] == 16) jta.p1.ch[i]='~';
        if(jta.p1.plottype[i] == 17) jta.p1.ch[i]='+';
        if(jta.p1.plottype[i] == 18) jta.p1.ch[i]='x';
        if(jta.p1.plottype[i] == 19) jta.p1.ch[i]='=';     }
    }
    inputLabel.setText(jta.p1.label);
    inputXLabel.setText(jta.p1.xlabel);
    inputYLabel.setText(jta.p1.ylabel);
    jta.yenidenciz();
    }
}
    public void actionPerformed(ActionEvent e)
    {
            if(e.getSource()==xyset[0])
            {inputXmin.setText(""+xi2);
             inputYmin.setText(""+yi2);
             jta.p1.xmin=xi2;
             jta.p1.ymin=yi2;
            }
    else if(e.getSource()==xyset[1])
            {inputXmax.setText(""+xi3);
             inputYmax.setText(""+yi3);
             jta.p1.xmax=xi3;
             jta.p1.ymax=yi3;
            }
    else if (e.getSource()==inputXmin )
    {Double valXmin=new Double(inputXmin.getText());
    jta.p1.xmin=valXmin.doubleValue();}
    else if (e.getSource()==inputXmax )
    {Double valXmax=new Double(inputXmax.getText());
    jta.p1.xmax=valXmax.doubleValue();}
    else if (e.getSource()==inputYmin )
    {Double valYmin=new Double(inputYmin.getText());
    jta.p1.ymin=valYmin.doubleValue();}
    else if (e.getSource()==inputYmax )
    {Double valYmax=new Double(inputYmax.getText());
    jta.p1.ymax=valYmax.doubleValue();}
    else if (e.getSource()==inputXntic )
    {Integer valXntic=new Integer(inputXntic.getText());
    jta.p1.xntic=valXntic.intValue();}
    else if (e.getSource()==inputYntic )
    {Integer valYntic=new Integer(inputYntic.getText());
    jta.p1.yntic=valYntic.intValue();}
    else if (e.getSource()==inputLabel )
    {jta.p1.label=inputLabel.getText();}
    else if (e.getSource()==inputXLabel )
```

```
{jta.p1.xlabel=inputXLabel.getText();}
else if (e.getSource()==inputYLabel )
{jta.p1.ylabel=inputYLabel.getText();}
else if(e.getSource()==arkarenk2)
{ renk2=JColorChooser.showDialog(null,"graphic background color ",renk2 );
  jta.setBackground(renk2);
  arkarenk2.setBackground(renk2);
}
String s1;
for(int i=0;i<5;i++)
{
            if(e.getSource()==but1[i])
            { jta.p1.label_x[i]=xi2;
              jta.p1.label_y[i]=yi2;
              inputAdditionalXLabel[i].setText(""+xi2);
              inputAdditionalYLabel[i].setText(""+yi2);
            }
            if(e.getSource()==inputAdditionalLabel[i])
  {
   jta.p1.label_at_xy[i]=inputAdditionalLabel[i].getText();
   s1=inputAdditionalXLabel[i].getText();
   if(s1!="") jta.p1.label_x[i]=Double.parseDouble(s1);
   else jta.p1.label_x[i]=0.0;
   s1=inputAdditionalYLabel[i].getText();
   if(s1!="") jta.p1.label_y[i]=Double.parseDouble(s1);
   else jta.p1.label_y[i]=0.0;
   s1=jta.p1.label_at_xy[i];
  }
  else if(e.getSource()==inputAdditionalXLabel[i])
  {s1=inputAdditionalXLabel[i].getText();
   if(s1!="") jta.p1.label_x[i]=Double.parseDouble(s1);
   else jta.p1.label_x[i]=0.0;
  }
  else if(e.getSource()==inputAdditionalYLabel[i])
  {s1=inputAdditionalYLabel[i].getText();
   if(s1!="") jta.p1.label_y[i]=Double.parseDouble(s1);
   else jta.p1.label_y[i]=0.0;}
  }
for(int i=0;i<n;i++)
{
          Color color1=new Color(jta.p1.red[i],jta.p1.green[i],jta.p1.blue[i]);
          Color color2;
if (e.getSource()==col1[i] )
{ color2=JColorChooser.showDialog(null,"line  "+i,color1 );
jta.p1.red[i]=color2.getRed();jta.p1.green[i]=color2.getGreen();jta.p1.blue[i]=color2.getBlue();
 col1[i].setBackground(color2);
}
}
jta.yenidenciz();
}

  public void degerleriGir(String e,int x,int y)
  {
  s=e;
  if(jta.p1.xlog!=0)
  {xi1=jta.p1.xmin+Math.pow(10.0,(double)((x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(double)jta.p1.xminmaxlog+1.0))-
10.0; }
  else
```

```
    {xi1=jta.p1.xmin+(double)(x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(jta.p1.xmax-jta.p1.xmin);}
    if(jta.p1.ylog!=0)
    {yi1=jta.p1.ymin-10.0+Math.pow(10.0,((double)(1.0-(double)(y-jta.p1.yabsmin-
jta.p1.absheight)/(double)jta.p1.absheight*(double)jta.p1.yminmaxlog))); }
    else
    {yi1=jta.p1.ymin-(double)(y-jta.p1.yabsmin-jta.p1.absheight)/(double)jta.p1.absheight*(jta.p1.ymax-jta.p1.ymin);}
    altbar.setText(e+" ["+xi1+" , "+yi1+"] ");
    //jta.yenidenciz();
    }

    public void degerleriGir1(String e,int x,int y)
    {
    s=e;
    if(jta.p1.xlog!=0)
    {xi2=jta.p1.xmin+Math.pow(10.0,(double)((x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(double)jta.p1.xminmaxlog+1.0))-
10.0; }
    else
    {xi2=jta.p1.xmin+(double)(x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(jta.p1.xmax-jta.p1.xmin);}
    if(jta.p1.ylog!=0)
    {yi2=jta.p1.ymin-10.0+Math.pow(10.0,((double)(1.0-(double)(y-jta.p1.yabsmin-
jta.p1.absheight)/(double)jta.p1.absheight*(double)jta.p1.yminmaxlog))); }
    else
    {yi2=jta.p1.ymin-(double)(y-jta.p1.yabsmin-jta.p1.absheight)/(double)jta.p1.absheight*(jta.p1.ymax-jta.p1.ymin);}
    System.out.println(xi2+" "+yi2);
    }
    public void degerleriGir2(String e,int x,int y)
    {
    s=e;
    if(jta.p1.xlog!=0)
    {xi3=jta.p1.xmin+Math.pow(10.0,(double)((x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(double)jta.p1.xminmaxlog+1.0))-
10.0; }
    else
    {xi3=jta.p1.xmin+(double)(x-jta.p1.xabsmin)/(double)jta.p1.abswidth*(jta.p1.xmax-jta.p1.xmin);}
    if(jta.p1.ylog!=0)
    {yi3=jta.p1.ymin-10.0+Math.pow(10.0,((double)(1.0-(double)(y-jta.p1.yabsmin-
jta.p1.absheight)/(double)jta.p1.absheight*(double)jta.p1.yminmaxlog))); }
    else
    {yi3=jta.p1.ymin-(double)(y-jta.p1.yabsmin-jta.p1.absheight)/(double)jta.p1.absheight*(jta.p1.ymax-jta.p1.ymin);}
    System.out.println(xi3+" "+yi3);
    }

    public void mousePressed(MouseEvent e)
    {  }

    public void mouseClicked(MouseEvent e)
    {if(e.isMetaDown()) {degerleriGir2("Mouse values are recorded",e.getX(),e.getY());} //sağ tuş basıldığında
      else if(e.isAltDown())            //orta tuş basıldığında
          {degerleriGir1("Mouse values are recorded",e.getX(),e.getY());} //sağ tuş basıldığında}
      else { degerleriGir1("Mouse values are recorded",e.getX(),e.getY());}  //sol tuş basıldığında
    }

    public void mouseReleased(MouseEvent e)
    {}

    public void mouseEntered(MouseEvent e)
    { setTitle("Mouse is in the applet area"); }

    public void mouseExited(MouseEvent e)
```

```
    { setTitle("Mouse is out of the applet area");
    }
    //MouseMotionListener (fare hareket dinleyicisi) metotlari

    public void mouseDragged(MouseEvent e)
    {
            degerleriGir("pushed and dragged",e.getX(),e.getY());
            }

    public void mouseMoved(MouseEvent e)
    { degerleriGir("on move",e.getX(),e.getY());
    }

    public static void plot()
    {
            //takes plot data from Plot.txt file
    PlotW pencere= new PlotW();
    pencere.addWindowListener(new BasicWindowMonitor());
    pencere.setSize(1000,700);
    pencere.setVisible(true);
    pencere.setDefaultCloseOperation(pencere.DISPOSE_ON_CLOSE);
    }

    public static void plot(Plot pi)
    {
    //take plot data from pi object
            PlotW pencere= new PlotW(pi);
    pencere.addWindowListener(new BasicWindowMonitor());
    pencere.setSize(1000,700);
    pencere.setVisible(true);
    pencere.setDefaultCloseOperation(pencere.DISPOSE_ON_CLOSE);
    }

    public static void main(String[] args)
    {
    PlotW pencere= new PlotW();
    pencere.addWindowListener(new BasicWindowMonitor());
    pencere.setSize(1000,700);
    pencere.setVisible(true);
    pencere.setDefaultCloseOperation(pencere.DISPOSE_ON_CLOSE);
    }

    public String bilgioku()
    {String s;
s= " 1.6.13 JAVA 2 DIMENSIONAL GRAPHIC OUTPUT CLASS : PLOT   \n ";
s+=" 2 dimensional scientific plot program to be \n";
s+= "=====================================================  \n ";
s+= " Plot program aims a simple yet sophisticated set of plots  \n ";
s+= " to be drawn for two dimensional engineering plot applications.\n";
s+= " Seeing the data is always easier for evaluating the results.\n";
s+= " There are commercially available plot programs, but the purpose of\n";
s+= " this package is to create an absolutely free and open code supporting plot\n";
s+= " definitions. Most of the subprograms in the package is add on methods from \n";
s+= " the actual requirements arised in numerical analysis and engineering applications \n";
s+= " If you would like to add into the package you are welcome, but please send me the \n";
s+= " add ups, if they will be usefull, I can make them permanent in the package \n";
s+= " my current email : turhan_coban@yahoo.com. Have a happy plotting\n";
s+= "=====================================================  \n ";
```

```
s+= "Program 1.6.13-1 PlotT1.java SAMPLE PROGRAM  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT1  \n ";
s+= "{  public static void main(String args[]) throws IOException  \n ";
s+= "  {Plot pp=new Plot(\"in.txt\");  \n ";
s+= "   pp.plot();}}  \n ";
s+= "  \n ";
s+= "Our first example code reads data from file in.txt and plot it  \n ";
s+= "The first line is the constructor line     \n ";
s+= "Plot pp=new Plot(\"in.txt\"); it inputs data from file in.txt and   pp.plot();   \n ";
s+= "term is actually draw the plot window. This term will always be used to   \n ";
s+= "open plot window\n ";
s+= "  \n ";
s+= "In the second page of the graphic window several editing features are available\n ";
s+= "such as plot shapes, headings, colors, grid, logarithmic scale, manual maximum\n ";
s+= "and minimum x an y scales, additional heading space in any desired location \n ";
s+= "of the plot \n ";
s+= "Page can be grided into a desired size, Graphic color can be changed by using\n ";
s+= "standard java JColorChooser class graphic editor   \n ";
s+= "   \n ";
s+= "In our second example program data in file in.txt is drawn by a single static definition   \n ";
s+= "  \n ";
s+= "Program 1.6.13-1A PlotT1.java sample program  \n ";
s+= "import java.io.*;  \n ";
s+= "class plotT1  \n ";
s+= "{    \n ";
s+= "  public static void main(String args[]) throws IOException  \n ";
s+= "  {Plot.file(\"in.txt\");}  \n ";
s+= "}  \n ";
s+= "  \n ";
s+= "Static definitions are usually preferred for one step plot drawing  \n ";
s+= "If several different plot jobs will be defined, dynamic definitions    \n ";
s+= "should be preferrred. In the next example data is given inside of the program\n ";
s+= "as two double vectors  \n ";
s+= "Program 1.6.13-2  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2A  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "     double yy[]={1,4,9,16,25};  \n ";
s+= "     Plot pp=new Plot(xx,yy);  \n ";
s+= "     pp.plot();}}  \n ";
s+= "   \n ";
s+= "The static version of the same program  :  \n ";
s+= "   \n ";
s+= "Program 1.6.13-2A  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2A  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot.data(xx,yy);}}  \n ";
s+= " The next program first defines the data and then add up cubic spline  \n ";
s+= " curve fittings to the given data (2 plot lines are to be drawn)  \n ";
s+= "Program 1.6.13-3  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2B  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
```

```
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addSpline(xx,yy,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "  \n ";
s+= "  \n ";
s+= "In order to do Cubic spline curve fitting adding term pp.addSpline(xx,yy,10); was enough\n ";
s+= "10 additional cubic spline fit data is added in between every point\n ";
s+= "ara değerleri kübik şerit interpolasyonuyla hesaplamamız anlamına gelmektedir.   \n ";
s+= " If the original curve index number is known, it can also be used to curve feet additional data \n ";
s+= " index number start from zero and goes upwards for evry new curve to be added \n ";
s+= "Program 1.6.13-3A  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2B2  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addSpline(0,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "Static definitions can not be applied in the prot drawings with more than one command such   \n ";
s+= "as in the above program, but you can always write your user specific static method  \n ";
s+= "to collect your own group of commands\n ";
s+= "  \n ";
s+= "Program 1.6.13-3B  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2B  \n ";
s+= "{      \n ";
s+= "     public static void plot(double xx[],double yy[])  \n ";
s+= "     {Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addSpline(xx,yy,10);  \n ";
s+= "      pp.plot();}  \n ";
s+= "        \n ";
s+= "     public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      plot(xx,yy);}}  \n ";
s+= "  \n ";
s+= "The same interpolation calculations can also be carried out by B spline\n";
s+= "interpolation with term  pp.addB_Spline(xx,yy,10);    \n ";
s+= "  \n ";
s+= "Program 1.6.13-3C  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2B1  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addB_Spline(xx,yy,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "  \n ";
s+= " If second degree orthogonal polynomial to be used for curve fitting   \n ";
s+= " (10 data points in between original points)  \n ";
s+= "  \n ";
s+= "Program 1.6.13-4  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2C  \n ";
```

```
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addOPEKK(xx,yy,2,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "  \n ";
s+= "  \n ";
s+= "if a second degree polynomial is desired fro corve fitting (fx=a0+a1*x+a2*x2) \n";
s+= " (10 data points in between original points)  \n ";
s+= "Program 1.6.13-5  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2D  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addEKK(xx,yy,2,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "  \n ";
s+= "  \n ";
s+= "Lagrange interpolation can also be used.  \n ";
s+= "Program 1.6.13-6  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2E  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {double xx[]={1,2,3,4,5};  \n ";
s+= "      double yy[]={1,4,9,16,25};  \n ";
s+= "      Plot pp=new Plot(xx,yy);  \n ";
s+= "      pp.addLagrange(xx,yy,10);  \n ";
s+= "      pp.plot();}}  \n ";
s+= "  \n ";
s+= "  \n ";
s+= "The detailed information about interpolation routines are given in \n ";
s+= "Numerical Analysis by using java programs text book  \n ";
s+= " \n";
s+= "Constructor in the plot program can be given as completely empty.   \n ";
s+= "Program 1.6.13-7 (Dynamic call)  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT01  \n ";
s+= "{public static void main (String args[]) throws IOException  \n ";
s+= "  {Plot pp=new Plot();  \n ";
s+= "   pp.plot();}}  \n ";
s+= "  \n ";
s+= "Program 1.6.13-7A (Static call)  \n ";
s+= "import java.io.*;  \n ";
s+= "class PlotT2B1  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {Plot.file();}}  \n ";
s+= "  \n ";
s+= "In this case program will look the data file  Plot.txt for data.\n ";
s+= "Data is given in the following form in this data\n ";
s+= "Plot label         \n ";
s+= "x axis label  \n ";
s+= "y axis label        \n ";
s+= "2   \n ";
s+= "in.txt  0 0 0 0    \n ";
s+= "out.txt 3 0 0 255   \n ";
```

```
s+= "  \n ";
s+= "The meaning of this data is as follows \"Plot label\"  \"x axis label\" \n";
s+= " \"y axis label\"   \n are defines graphic label headings \n";
s+= "2 indicates number of data files data to be taken from in.txt and out.txt\n ";
s+= "in this case, the next number is the plot line type and the last 3 numbers are plot color\n";
s+= "(red green and blue 0 to 255).  \n ";
s+= "Plot line type codes are as follows :   \n ";
s+= "\"1 __ __ __ __ \",  \n ";
s+= "\"2 . _ . _ . _ \",  \n ";
s+= "\"3 . _ _ . . _ _ .\",  \n ";
s+= "\"4 - - - - - - - - \",  \n ";
s+= "\"5  _______\",  \n ";
s+= "\"6  __ __ __ __ \",  \n ";
s+= "\"7 . _ . _ . _ \",  \n ";
s+= "\"8  . _ _ . . _ _ .\",  \n ";
s+= "\"9 - - - - - \",  \n ";
s+= "\"10  *      \" ,  \n ";
s+= "\"11  #      \",  \n ";
s+= "\"12  $      \",  \n ";
s+= "\"13  %      \",  \n ";
s+= "\"14  &      \",  \n ";
s+= "\"15  -      \",  \n ";
s+= "\"16  ~      \",  \n ";
s+= "\"17  +      \",  \n ";
s+= "\"18  x      \",  \n ";
s+= "\"19  =      \",  \n ";
s+= "\"20  square\",    \n ";
s+= "\"21  filled square\",  \n ";
s+= "\"22  circle\",  \n ";
s+= "\"23  filled circle\",  \n ";
s+= "\"24  triangle\",  \n ";
s+= "\"25  diamond \",  \n ";
s+= "\"26  pentagon\",  \n ";
s+= "\"27  hexagon\",  \n ";
s+= "\"28  filled triangle\",  \n ";
s+= "\"28  filled diamond\",  \n ";
s+= "\"30  filled pentagon\",  \n ";
s+= "\"31  filled hexagon\",  \n ";
s+= "\"32  3 sided star\",  \n ";
s+= "\"33  4 sided star\",  \n ";
s+= "\"34  5 sided star\",  \n ";
s+= "\"35  6 sided star\",  \n ";
s+= "\"36  filled 3 sided star\",  \n ";
s+= "\"37  filled 4 sided star\",  \n ";
s+= "\"38  filled 5 sided star\",  \n ";
s+= "\"39  filled 6 sided star\",  \n ";
s+= "\"40  bar graphic\",  \n ";
s+= "\"41  filled bar graphic\"};  \n ";
s+= ".   \n ";
s+= "  \n ";
s+= "Plot class is also draw a given function   \n ";
s+= "functions to be drawn should be extended to abstract class f_x.  \n ";
s+= "  \n ";
s+= "Program 1.6.13-8 (Dynamic version)  \n ";
s+= "import java.io.*;  \n ";
s+= "class f3 extends f_x  \n ";
s+= "{double func(double x)  \n ";
s+= " {return x*x*x+3.6*x*x-36.4;}}  \n ";
```

```
s+= "   \n ";
s+= "class PlotT06  \n ";
s+= "{public static void main (String args[]) throws IOException  \n ";
s+= " {f3 ff2=new f3();      \n ";
s+= "  Plot pp=new Plot(ff2,0.0,7.0,400,2);  \n ";
s+= "  pp.plot();}}  \n ";
s+= "  \n ";
s+= "Program 1.6.13-8A (Static version)  \n ";
s+= "import java.io.*;  \n ";
s+= "  \n ";
s+= "class f3 extends f_x  \n ";
s+= "{double func(double x)  \n ";
s+= " { return x*x*Math.abs(Math.sin(Math.sqrt(x)))-5.0;}}  \n ";
s+= "   \n ";
s+= "class PlotT06  \n ";
s+= "{    public static void main (String args[]) throws IOException  \n ";
s+= "     {Plot.func(new f3(),0.0,10.0,400,0);}  \n ";
s+= "  \n ";
s+= "The function given in the previous program is : f(x)=x3+3.6x2-36.4   \n ";
s+= "function is required to be plotted in between 0 and 7 with 400 data points\n ";
s+= "  \n ";
s+= "More than one function can be defined in the same plot window with one  \n ";
s+= "function class  \n ";
s+= "Program 1.6.13-9 (Dynamic version)  \n ";
s+= "import java.io.*;  \n ";
s+= "class f3 extends fi_x  \n ";
s+= "{  \n ";
s+= " double[] func(double x)  \n ";
s+= " { double f[]=new double[2];           \n ";
s+= "   f[0]=x*Math.sin(10.0*Math.PI*x)+1.0;  \n ";
s+= "   f[1]=x*x;  \n ";
s+= "   return f;}}  \n ";
s+= "   \n ";
s+= "class PlotT03  \n ";
s+= "{public static void main (String args[]) throws IOException  \n ";
s+= "  { f3 ff2=new f3();      \n ";
s+= "    Plot pp=new Plot(ff2,-2.0,2.0,400,2);  \n ";
s+= "    pp.plot();}}  \n ";
s+= "  \n ";
s+= "Program 1.6.13-9A (Statik versiyon)  \n ";
s+= "import java.io.*;  \n ";
s+= "  \n ";
s+= "class f3 extends fi_x  \n ";
s+= "{  \n ";
s+= " double[] func(double x)  \n ";
s+= " { double f[]=new double[2];           \n ";
s+= "   f[0]=x*Math.sin(10.0*Math.PI*x)+1.0;  \n ";
s+= "   f[1]=x*x;  \n ";
s+= "   return f;}  \n ";
s+= "}  \n ";
s+= "  \n ";
s+= "class PlotT03  \n ";
s+= "{public static void main (String args[]) throws IOException  \n ";
s+= "  { Plot.func(new f3(),-2.0,2.0,400,2);}}  \n ";
s+= "  \n ";

s+= "or the same plot can be achieved by using different plot function classes.\n ";
s+= "  \n ";
```

```
s+= "Program 1.6.13-10  \n ";
s+= "import java.io.*;  \n ";
s+= "class f3 extends f_x  \n ";
s+= "{double func(double x)  \n ";
s+= " {return x*x;}}  \n ";
s+= "  \n ";
s+= "class f4 extends f_x  \n ";
s+= "{double func(double x)  \n ";
s+= " {return x*Math.sin(10.0*Math.PI*x)+1.0;}}  \n ";
s+= "   \n ";
s+= "class PlotT07  \n ";
s+= "{public static void main (String args[]) throws IOException  \n ";
s+= " {f3 ff3=new f3();  \n ";
s+= "  f4 ff4=new f4();      \n ";
s+= "  Plot pp=new Plot(ff3,-3.0,2.0,400);  \n ";
s+= "  pp.addFunction(ff4,-2.0,2.0,400);  \n ";
s+= "  pp.plot();}}  \n ";
s+= "  \n ";
s+= "Different plot zones can be defined when functions called seperately.   \n ";
s+= "For example in the previous program second function is called for -2,2  \n ";
s+= "zone  \n ";
s+= "  \n ";
s+= "pp.addFunction(ff4,-2.0,2.0,400);  \n ";
s+= "   \n ";
s+= "method addData is also available it adds additional data points \n ";
s+= "  \n ";
s+= "pp.addData(xx,yy,0,0,0,0);  \n ";
s+= "  \n ";
s+= "Last example combines different methods of the plot class together  \n ";
s+= "  \n ";
s+= "Program 1.6.13-11  \n ";
s+= "import java.io.*;  \n ";
s+= "   \n ";
s+= "class f3 extends fi_x  \n ";
s+= "{  \n ";
s+= " double[] func(double x)  \n ";
s+= " {  \n ";
s+= "   double f[]=new double[2];            \n ";
s+= "   f[0]=Math.sin(x)+x/2;  \n ";
s+= "   f[1]=Math.sqrt(x);  \n ";
s+= "   return f;  \n ";
s+= " }}  \n ";
s+= "class PlotT3  \n ";
s+= "{  \n ";
s+= "     public static void main (String args[]) throws IOException  \n ";
s+= "     {//Plot pp=new Plot();  \n ";
s+= "     String fi[]={\"in.txt\",\"out.txt\"};  \n ";
s+= "     f3 ff=new f3();  \n ";
s+= "     int ipt[]={0,1};  \n ";
s+= "     int ir[]={0,0};  \n ";
s+= "     int ig[]={0,0};  \n ";
s+= "     int ib[]={0,2};  \n ";
s+= "     double xx[]={1,2,3,4,5,6};  \n ";
s+= "     double yy[]={2,3,4,5,6,7};  \n ";
s+= "     Plot pp=new Plot(fi); //read data from fi (in.txt,out.txt data are added up)  \n ";
s+= "     //pp.addData(fi);  \n ";
s+= "     pp.setPlotType(0,0);//  \n ";
s+= "     pp.setPlotType(1,1);//   \n ";
```

```
s+= "     pp.addData(xx,yy,0,0,0,0); //one set of addional data  \n ";
s+= "     pp.setPlotType(2,41);//bar chart  \n ";
s+= "     pp.setColor(2,0,0,255); //blue  \n ";
s+= "     pp.addFunction(ff,0.0,20.0,50,2,ipt,ir,ig,ib); //two additional function  \n ";
s+= "     pp.setPlabel(\"Fonksiyon Plot\");  \n ";
s+= "     pp.setXlabel(\"x ekseni\");  \n ";
s+= "     pp.setYlabel(\"y ekseni\");  \n ";
s+= "     pp.plot();}}  \n ";
s+= "  \n ";
s+= "Grafik Package basic components :   \n ";
s+= "  \n ";
s+= "Plot.java constructors, input and output methods  \n ";
s+= "PlotW.java  : basic graphic and control algorithms  \n ";
s+= "PlotShapesSW: defines different plot shapes such as line, star,triangle   \n ";
s+= "Plot2D : Converts shapes defined in the PlotShapesSW class into plot screen   \n ";
s+= "format   \n ";
s+= "\n";
s+= "The list of class Plot constructors  \n ";
s+= "  \n ";
s+= "Constructor classes : (defines the graphic window and prepare the data to be plotted)  \n ";
s+= "public Plot() throws IOException     \n ";
s+= "public Plot(String pl,String xl,String yl,int xnt,int ynt,int xgo,int ygo,String fn[],int ipt[],int ir[],int ig[]                    \n ";
s+= " ,int ib[]) throws IOException       \n ";
s+= "  public Plot(String fn[],int ipt[],int ir[],int ig[],int ib[]) throws IOException              \n ";
s+= "  public Plot(String fn[]) throws IOException  : Read file names from String fn[] and plot them   \n ";
s+= "  public Plot(double xi[][],double yi[][],int ipt[],int ir[],int ig[],int ib[]) : read plot data from xi[][] and yi[][]  \n ";
s+= "  public Plot(double xiyi[][]) : read plot data xi[] and yi[] from xiyi[][] string     \n ";
s+= "  public Plot(double xi[][],double yi[][]) : read plot data from xi[][] and yi[][]  \n ";
s+= "  public Plot(double xi[],double yi[]) : read plot data from xi[] and yi[]  \n ";
s+= "  public Plot(f_x f,double xm,double xma,int N,int ipt,int ir,int ig,int ib)    \n ";
s+= " public Plot(f_x f,double xm,double xma,int N) : create plot data from a simple function f_x f  \n ";
s+= " public Plot(fi_x f,double xm,double xma,int Number_of_Data,int Number_of_Function)   \n ";
s+= "   \n ";
s+= "Static method definitions : (give short descriptions for one step plotting)  \n ";
s+= " public static double[] data(double xmin,double xmax,double dx)   \n ";
s+= " public static double[] data(double xmin,double xmax,int n)  \n ";
s+= " public static Plot func(f_x ff1,double xmin,double xmax,int n,int pnumber)  \n ";
s+= " public static Plot func(f_x ff1,double xmin,double xmax,int n)  \n ";
s+= " public static Plot func(f_x ff1,double xmin,double xmax)  \n ";
s+= " public static Plot func(fi_x ff1,double xmin,double xmax,int n,int pnumber)  \n ";
s+= " public static Plot func(fi_x ff1,double xmin,double xmax,int n)  \n ";
s+= " public static Plot func(fi_x ff1,double xmin,double xmax)  \n ";
s+= "  \n ";
s+= " adding x,y and heading label  \n ";
s+= "  public void setPlabel(String ip) {label=ip;}  \n ";
s+= "  public void setXlabel(String ix) {xlabel=ix;}  \n ";
s+= "  public void setYlabel(String iy) {ylabel=iy;}  \n ";
s+= "  public void setXYlabel(String ix,String iy) {xlabel=ix;ylabel=iy;}  \n ";
s+= "  \n ";
s+= "Adding additional data into the graphic window \n ";
s+= "public void addData(double xi[],double yi[])  \n ";
s+= "public void addData(double xi[],double yi[],int iplottype)  \n ";
s+= "public void addData(double xi[],double yi[],int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addData(double xi[][],double yi[][])  \n ";
s+= "public void addData(double xi[][],double yi[][],int ip[])  \n ";
s+= "public void addData(double xi[][],double yi[][],int ip[],int ir[],int ig[],int ib[])  \n ";
s+= "public void addFunction(f_x f,double xmi,double xma,int N,int ipt)  \n ";
s+= "public void addFunction(f_x f,double xmi,double xma,int N)  \n ";
```

```
s+= "public void addFunction(f_x f,double xmi,double xma,int N,int ipt,int ir,int ig,int ib)              \n ";
s+= "public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function)  \n ";
s+= "public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function,int ipt[])  \n ";
s+= "public void addFunction(fi_x f,double xmin,double xmax,int Number_of_Data,int Number_of_Function, ,int ipt[])  \n ";
s+= "  \n ";
s+= "adding data with curve fit  \n ";
s+= "public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi)  \n ";
s+= "public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)  \n ";
s+= "public void addOPEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi)  \n ";
s+= "public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype)  \n ";
s+= "public void addEKK(double xi[],double yi[],int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype)  \n ";
s+= "public void addEKK(int linenumber,int npolinom,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addLagrange(double xi[],double yi[],int aradegersayisi)  \n ";
s+= "public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype)  \n ";
s+= "public void addLagrange(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addLagrange(int linenumber,int aradegersayisi,int iplottype)  \n ";
s+= "public void addLagrange(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addB_Spline(double xi[],double yi[],int nSpline)  \n ";
s+= "public void addB_Spline(double ai[][],int nSpline)  \n ";
s+= "public void addB_Spline(double xi[],double yi[],int nSpline,int iplottype)  \n ";
s+= "public void addB_Spline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addB_Spline(int linenumber,int nSpline)  \n ";
s+= "public void addB_Spline(int linenumber,int nSpline,int iplottype)  \n ";
s+= "public void addB_Spline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addSpline(double xi[],double yi[],int nSpline)  \n ";
s+= "public void addSpline(double ai[][],int nSpline)  \n ";
s+= "public void addSpline(double xi[],double yi[],int nSpline,int iplottype)  \n ";
s+= "public void addSpline(double xi[],double yi[],int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "public void addSpline(int linenumber,int nSpline)  \n ";
s+= "public void addSpline(int linenumber,int nSpline,int iplottype)  \n ";
s+= "public void addSpline(int linenumber,int aradegersayisi,int iplottype,int ir,int ig,int ib)  \n ";
s+= "  \n ";
s+= "changing the graphic properties  \n ";
s+= "===========Grafik türü  =====================  \n ";
s+= "public void setPlotType(int plot_type[])        \n ";
s+= "public void setPlotType(int dataset,int plot_no)  \n ";
s+= "public void setPlotType(int dataset,char plot_char)  \n ";
s+= "===========x ve y grid  ==========================  \n ";
s+= "public void setXgrid(int igx)  \n ";
s+= "public void setYgrid(int igy)  \n ";
s+= "public void setGrid(int igx,int igy)  \n ";
s+= "public void setXtic(int ixt)  \n ";
s+= "public void setYtic(int iyt)  \n ";
s+= "public void setXYtic(int ixt,int iyt)  \n ";
s+= "==========logaritmik graphic scale openning and closing ==========  \n ";
s+= "public void setXlogScaleOn()  \n ";
s+= "public void setYlogScaleOn()  \n ";
s+= "public void setXlogScaleOff()  \n ";
s+= "public void setYlogScaleOff()  \n ";
s+= "==========setColor==========  \n ";
s+= "public void setColor(int dataset,int ired,int igreen,int iblue)  \n ";
s+= "public void setColor(int dataset,Color c)  \n ";
s+= "public void setColor(int ired[],int igreen[],int iblue[])  \n ";
s+= "public void setColor(Color c[])  \n ";
s+= "================ Setting maximum and minimum values of the plot window manually ====  \n ";
s+= "public void setMinMax()  \n ";
```

```
s+= "public void setMinMax(double xi[][],double yi[][])  \n ";
s+= "public void setMinMax(double iminx,double imaxx,double iminy,double imaxy)  \n ";
s+= "=============== Formatting graphic window ================================  \n ";
s+= " public void set_plotwindow(int width,int height,double xip,double yip,double dxp,double dyp)   \n ";
s+= " public void setabsMaxMin(int ixmin,int iymin,int idx,int idy)  \n ";
s+= "  \n ";
s+= "Drawing the data in graphic format  \n ";
s+= "public void plot()  \n ";
return s;}
     }
```

PROGRAM 21.1-22 Plot2D class

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
import javax.swing.*;
import java.awt.print.PrinterJob;
import java.awt.print.*;

public class Plot2D extends JPanel implements Printable
{
            public Graphics2D g;
            public Plot p1;
   final static Color bg = Color.white;
   final static Color fg = Color.black;
   final static Color red = Color.red;
   final static Color white = Color.white;
   final static BasicStroke stroke = new BasicStroke(1.0f);
   final static BasicStroke boldStroke = new BasicStroke(2.0f);
   PlotShapesSW ps;
   double xx[],yy[];
   int xn;

   public Plot2D(Plot pi)
   {
            //Initialize drawing colors
   setBackground(Color.white);
   setForeground(Color.black);
   //read data from file Plot data previously defined
   p1=pi;
   xx=new double[200];
   yy=new double[200];
   xn=0;
            }
            public Plot2D()
   {
            //Initialize drawing colors
   setBackground(Color.white);
   setForeground(Color.black);
   //read data from file Plot data previously defined
   try{p1=new Plot();
     } catch(IOException ioe) {System.err.println("IOExceptionin opening plot");}
   xx=new double[200];
   yy=new double[200];
   xn=0;
            }
```

```
public Plot2D(String pl,String xl,String yl,int xnt,int ynt
,int xgo,int ygo,String ifn[],int ipt[],int ir[],int ig[]
,int ib[])
{
 //input variables
 // pl : plot label
 // xl : x axis label
 // yl : y axis label
 //xnt : x axis tic (0=off, any other:on)
 //ynt : y axis tic (0=off, any other:on)
 //xgo : x axis grid on (0=off, any other:on)
 //ygo : y axis grid on (0=off, any other:on)
 //ifn :file name array variable
 //ipt : plot type array variable
 //ir  : color red component array variable
 //ig  : color green componenet array variable
    //Initialize drawing colors
    setBackground(Color.white);
    setForeground(Color.black);
    try{
         p1=new Plot(pl,xl,yl,xnt,ynt,xgo,ygo,ifn,ipt,ir,ig,ib);
       } catch(IOException ioe) {System.err.println("IOExceptionin opening plot");}
xx=new double[200];
yy=new double[200];
xn=0;
}

public void setXY(double ixx[],double iyy[],int ixn)
{
xn=ixn;
           for(int i=0;i<xn;i++)
  {xx[i]=ixx[i];yy[i]=iyy[i];}
           }

public void yenidenPlotDatasiOku()
{
    try{
         p1=new Plot();
       } catch(IOException ioe) {System.err.println("IOExceptionin opening plot");}
}

public void yenidenPlotDatasiOku(Plot pi)
{
         p1=pi;
 }

public void plotVerisiEkle(double xi[],double yi[],int iplottype,int ir,int ig,int ib)
{
   p1.addData(xi,yi,iplottype,ir,ig,ib);
}

public void plotVerisiEkle(double xi[][],double yi[][],int iplottype[],int ir[],int ig[],int ib[])
{
   p1.addData(xi,yi,iplottype,ir,ig,ib);
}
public void plotVerisiEkle(f_x f,double xmin,double xmax,int N,int iplottype,int ir,int ig,int ib)
{
   p1.addFunction(f,xmin,xmax,N,iplottype,ir,ig,ib);
```

```
}

public void yenidenPlotDatasiOku(String pl,String xl,String yl,int xnt,int ynt
,int xgo,int ygo,String ifn[],int ipt[],int ir[],int ig[]
,int ib[])
{
 //input variables
 // pl : plot label
 // xl : x axis label
 // yl : y axis label
 //xnt : x axis tic (0=off, any other:on)
 //ynt : y axis tic (0=off, any other:on)
 //xgo : x axis grid on (0=off, any other:on)
 //ygo : y axis grid on (0=off, any other:on)
 //ifn :file name array variable
 //ipt : plot type array variable
 //ir  : color red component array variable
 //ig  : color green componenet array variable
 //ib  : color blue componenet
   //Initialize drawing colors
   setBackground(Color.white);
   setForeground(Color.black);
   try{
       p1=new Plot(pl,xl,yl,xnt,ynt,xgo,ygo,ifn,ipt,ir,ig,ib);
      } catch(IOException ioe) {System.err.println("IOExceptionin opening plot");}
}

public void yenidenciz()
{
repaint();
}

public void yazdir()
{
   PrinterJob printJob = PrinterJob.getPrinterJob();
   printJob.setPrintable(this);
   if (printJob.printDialog()) {
       try { printJob.print();} catch (Exception ex) {ex.printStackTrace();}                                    }
}

public int print(Graphics g, PageFormat pf, int pi) throws PrinterException {
   if (pi >= 1) {
              return Printable.NO_SUCH_PAGE;
              }
   super.paintComponent(g);
   Graphics2D g2 = (Graphics2D) g;
   drawShapes(g2);
   return Printable.PAGE_EXISTS;
}


public void drawShapes(Graphics2D g2)
{
   g=g2;
             //draw Plot
   Dimension d=getSize();
   g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
   RenderingHints.VALUE_ANTIALIAS_ON);
```

```
        int gridWidth = d.width;
        int gridHeight = d.height;
        int rowspacing=(int)(d.width/100.0);
        int columnspacing=(int)(d.height/50.0);
        int rectWidth=gridWidth-columnspacing;
        int rectHeight=gridHeight-rowspacing;
        p1.xabsmin=(int)(0.25*rectWidth);
        p1.yabsmin=(int)(0.2*rectHeight);
        p1.abswidth=(int)(0.65*rectWidth);
        p1.absheight=(int)(0.7*rectHeight);
        g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_OFF);
        g2.setStroke(stroke);
        g2.setPaint(fg);
        g2.drawRect(p1.xabsmin,p1.yabsmin,p1.abswidth,p1.absheight);
        ps=new PlotShapesSW(g2,p1.xabsmin,p1.yabsmin,
        p1.absheight,p1.abswidth,p1.xmin,p1.xmax,p1.ymin,p1.ymax,p1.xlog,p1.ylog);
        if(p1.xgridon!=0)
          ps.drawXGrid(p1.xntic);
        if(p1.ygridon!=0)
          ps.drawYGrid(p1.yntic);
        int i,j;
        ps.drawMouseTic(xx,yy,xn);
        for(i=0;i<p1.nline;i++)
          {
          //  Select plot colors
          g2.setPaint(new Color(p1.red[i],p1.green[i],p1.blue[i]));
          ps.drawPlotLines(i,p1.plottype,p1.x,p1.y,p1.n,p1.ch);
          }//end of for(i=0
        for(i=0;i<5;i++)
        {if(p1.label_at_xy[i]!="") ps.drawAdditionalLabel(p1.label_at_xy[i],p1.label_x[i],p1.label_y[i]);}
        g2.setPaint(fg);
        ps.drawXTic(p1.xntic,(p1.ymax-p1.ymin)/80.0,0);
        ps.drawYTic(p1.yntic,(p1.xmax-p1.xmin)/80.0,0);
        ps.drawXNumbers(p1.xntic);
        ps.drawYNumbers(p1.yntic);
        ps.drawLabel(p1.label);
        ps.drawXLabel(p1.xlabel);
        ps.drawYLabel(p1.ylabel);
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D) g;
        drawShapes(g2);
    }
}
```

PROGRAM 21.1-23 PlotShapesSW class

```
//=====================================================
// Numerical Analysis package in java
// PlotShapes class
// This class convert graphic draw methods to
// plot coordinates and gives additional plotting methods
// Dr. Turhan Coban
// =====================================================
import java.awt.*;
import java.awt.event.*;
```

```
import java.awt.geom.*;
import javax.swing.*;
import java.awt.font.*;
import java.io.*;
import java.util.Locale;

public class PlotShapesSW
{
public Graphics2D g;
int xabsmin,yabsmin;
int absheight,abswidth;
double xmin,xmax,ymin,ymax;
public int xlog,ylog;
double xminmaxlog,yminmaxlog;
Font f;
final static float dash1[] = {10.0f};
final static float dash2[] = {10.0f,3.0f,3.0f};
final static float dash3[] = {10.0f,3.0f,3.0f};
final static float dash4[] = {5.0f,5.0f,5.0f};

final static BasicStroke d1 = new BasicStroke(1.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash1, 2.0f);
final static BasicStroke d2 = new BasicStroke(1.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash2, 2.0f);
final static BasicStroke d3 = new BasicStroke(1.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash3, 2.0f);
final static BasicStroke d4 = new BasicStroke(1.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash4, 2.0f);
final static BasicStroke d5 = new BasicStroke(2.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash1, 2.0f);
final static BasicStroke d6 = new BasicStroke(2.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash2, 2.0f);
final static BasicStroke d7 = new BasicStroke(2.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash3, 2.0f);
final static BasicStroke d8 = new BasicStroke(2.0f,BasicStroke.CAP_BUTT,
                                    BasicStroke.JOIN_MITER,
                                    3.0f, dash4, 2.0f);


public PlotShapesSW(Graphics2D gi,int xabsmini  ,int yabsmini,
                      int absheighti,int abswidthi,
                    double xmini,double xmaxi,
                    double ymini,double ymaxi, int xlogi,int ylogi)
{
// xabsmin   : absolute(pixel) starting point x axis
// yabsmin   : absolute(pixel) starting point y axis
// absheight : absolute(pixel) height of plotting window
// abswidth  : absolute(pixel) width of plotting window
// xmin      : minimum x value (real number)
// xmax      : maximum x value (real number)
// ymin      : minimum y value (real number)
```

```
// ymax      : maximum y value (real number)
// g         : graphic object that actual drawing is done through
g=gi;
//Font fonts[]=GraphicsEnvironment.getLocalGraphicsEnvironment().getAllFonts();
//f=fonts[2].deriveFont(Font.BOLD,14);//You can change Fony from here
//g.setFont(f);
xabsmin=xabsmini;
yabsmin=yabsmini;
absheight=absheighti;
abswidth=abswidthi;
xmin=xmini;
xmax=xmaxi;
ymin=ymini;
ymax=ymaxi;
xlog=xlogi;
ylog=ylogi;
if(xlog!=0) {xminmaxlog=Math.log10(xmax-xmin+10.0)-1.0;}
if(ylog!=0) {yminmaxlog=Math.log10(ymax-ymin+10.0)-1.0;}
}

public void drawLine(int plottype,double x1,double y1,double x2,double y2)
{
// draw a line from (x1,y1) to (x2,y2)
        int xx1,xx2,yy1,yy2;
        if(xlog!=0)
        {
        xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth));
        xx2=(int)(xabsmin+((Math.log10(x2-xmin+10.0)-1.0)/xminmaxlog*abswidth));
                }
        else
        {
                xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);
        xx2=(int)(xabsmin+(x2-xmin)/(xmax-xmin)*abswidth);
                }
        if(ylog!=0)
        {
        yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));
        yy2=(int)(yabsmin+absheight-((Math.log10(y2-ymin+10.0)-1.0)/yminmaxlog*absheight));
                }
        else
        {
                yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);
        yy2=(int)(yabsmin+absheight-(y2-ymin)/(ymax-ymin)*absheight);
                }
        g.draw(new Line2D.Double( xx1,yy1,xx2,yy2));
}  //end of drawLine


public void drawPolyline(int plottype,double x1[],double y1[])
{
// draw a line from (x1,y1) to (x2,y2)
        int points=y1.length;
        int xPoints[]=new int[points];
        int yPoints[]=new int[points];
        int xx1,yy1;
        for(int i=0;i<points;i++)
        {
                  if(xlog!=0)
```

```
          {xx1=(int)(xabsmin+(Math.log10(x1[i]-xmin+10.0)-1.0)/xminmaxlog*abswidth);}
          else
          {xx1=(int)(xabsmin+(x1[i]-xmin)/(xmax-xmin)*abswidth);}
          if(ylog!=0)
          {yy1=(int)(yabsmin+absheight-(Math.log10(y1[i]-ymin+10.0)-1.0)/yminmaxlog*absheight);}
          else
          {yy1=(int)(yabsmin+absheight-(y1[i]-ymin)/(ymax-ymin)*absheight);}
       xPoints[i]=xx1;
       yPoints[i]=yy1;
                }
       g.drawPolyline(xPoints,yPoints,points);
}  //end of drawPolyLine

public  void drawChar(char ch,double x1,double y1)
{
   // draws a single character at (x1,y1)
       char ch1[]=new char[1];
       ch1[0]=ch;
       int h=g.getFontMetrics().getHeight();
       int w=h-2;
       int xx1,yy1;
          if(xlog!=0)
          {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-abswidth/200.0));}
          else
          {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}//-abswidth/200.0);}
          if(ylog!=0)
          {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight))+h/2;}
          else
          {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);}//+h/2;}
       g.drawChars(ch1,0,1,xx1,yy1);
}  //end of PlotShapes.drawChar

public void drawChars(char ch[],int firstspace,int numberofchars,
                           double x1,double y1)
{
       int h=g.getFontMetrics().getHeight();
       int w=h-2;
       int xx1,yy1;
       if(xlog!=0)
          {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-abswidth/80.0));}
       else
          {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}//-abswidth/80.0);}
       if(ylog!=0)
          {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));}
       else
          {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);}
       g.drawChars(ch,firstspace,numberofchars,xx1,yy1);
}  //end of PlotShapes.drawChars

public void drawString(String s,double x1,double y1)
{
   //draws a String at (x1,y1)
       int h=g.getFontMetrics().getHeight();
       int w=h-2;
          int xx1,yy1;
          if(xlog!=0)
          {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth));}
          else
```

```
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);}
      g.drawString(s,xx1,yy1);

}  //end of PlotShapes.drawStrings

public void drawRect(double x1,double y1,int rectwidth,int rectheight)
{
// draw a rectangle starting at (x1,y1)
// with dimensions of (rectwidth,rectheight)
        int xx1,yy1;
        if(xlog!=0)
        {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-rectwidth/2));}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-rectwidth/2);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-rectheight/2));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-rectheight/2);}

      g.draw(new Rectangle2D.Double(xx1,yy1,rectwidth,rectheight));

}//end of drawRect

public void drawBar(double x1,double y1)
{
// draw a rectangle starting at (x1,y1)
// with dimensions of (rectwidth,rectheight)
        int xx1,yy1;
        double barheight;
        double barwidth=abswidth/300;
        if(xlog!=0)
        {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-barwidth/2.0));}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-barwidth/2.0);
        }
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));
        barheight =(int)((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight);
        }
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);
        barheight =(int)((y1-ymin)/(ymax-ymin)*absheight);}
        g.draw(new Rectangle2D.Double(xx1,yy1,barwidth,barheight));
}//end of drawBar

public void fillBar(double x1,double y1)
{
       int xx1,yy1;
        double barheight;
        double barwidth=abswidth/300;
        if(xlog!=0)
        {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-barwidth/2.0));}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-barwidth/2.0);
```

```
        }
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));
        barheight =(int)((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight);
        }
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);
        barheight =(int)((y1-ymin)/(ymax-ymin)*absheight);}
        g.fill(new Rectangle2D.Double(xx1,yy1,barwidth,barheight));
}//end of fillBar

public void fillRect(double x1,double y1,int rectwidth,int rectheight)
{
// draw a rectangle starting at (x1,y1)
// with dimensions of (rectwidth,rectheight)
// draw a rectangle starting at (x1,y1)
// with dimensions of (rectwidth,rectheight)
        int xx1,yy1;
        if(xlog!=0)
        {xx1=(int)(xabsmin+(Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-rectwidth/2);}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-rectwidth/2);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-rectheight/2));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-rectheight/2);}
      g.fill(new Rectangle2D.Double(xx1,yy1,rectwidth,rectheight));
}

public void drawEllipse(double x1,double y1,int width,int height)
{
//      draw an oval with the centre of (x1,y1)
//      with dimension of (width,height)
        int xx1,yy1;
        if(xlog!=0)
        {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-width/2));}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-width/2);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-height/2));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-height/2);}
      g.draw(new Ellipse2D.Double(xx1,yy1,width,height));
}//end of drawRect

public void fillEllipse(double x1,double y1,int width,int height)
{
//      draw an oval with the centre of (x1,y1)
//      with dimension of (width,height)
        int xx1,yy1;
        if(xlog!=0)
        {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-width/2));}
        else
        {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-width/2);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-height/2));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-height/2);}
```

```
        g.fill(new Ellipse2D.Double(xx1,yy1,width,height));
}//end of drawRect

public void drawPolygon(double x1,double y1,int radius,int side)
{
//draw a polygon of n sides n=3(trinagle),n=4(dimond)....
        double Pi=Math.PI;
        int xvalues[]=new int[side+1];
        int yvalues[]=new int[side+1];
        GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
        xvalues.length);
        double angle_increase;
        double angle;
        angle_increase=2.0*Pi/side;
        angle=Pi/2.0;
        for(int i=0;i<=side;i++)
        {
                if(xlog!=0)
          {xvalues[i]=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth+radius*Math.cos(angle)));}
          else
          {xvalues[i]=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth+radius*Math.cos(angle));}
          if(ylog!=0)
          {yvalues[i]=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-
radius*Math.sin(angle)));}
          else
          {yvalues[i]=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-radius*Math.sin(angle));}
         angle+=angle_increase;
         if(i==0)
         {
         xvalues[side]=xvalues[i];yvalues[side]=yvalues[i];
         polygon.moveTo(xvalues[0],yvalues[0]);
         }
         else
         {
          polygon.lineTo(xvalues[i],yvalues[i]);
         }
        }
        g.draw(polygon);
}//end of drawPolygon

public void fillPolygon(double x1,double y1,int radius,int side)
{
//draw a polygon of n sides n=3(trinagle),n=4(dimond)....
//draw a polygon of n sides n=3(trinagle),n=4(dimond)....
        double Pi=Math.PI;
        int xvalues[]=new int[side+1];
        int yvalues[]=new int[side+1];
        GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
        xvalues.length);
        double angle_increase;
        double angle;
        angle_increase=2.0*Pi/side;
        angle=Pi/2.0;
        for(int i=0;i<=side;i++)
        {
                if(xlog!=0)
          {xvalues[i]=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth+radius*Math.cos(angle)));}
          else
```

```
          {xvalues[i]=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth+radius*Math.cos(angle));}
          if(ylog!=0)
          {yvalues[i]=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight-
radius*Math.sin(angle)));}
          else
          {yvalues[i]=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight-radius*Math.sin(angle));}
         angle+=angle_increase;
         if(i==0)
         {
         xvalues[side]=xvalues[i];yvalues[side]=yvalues[i];
         polygon.moveTo(xvalues[0],yvalues[0]);
         }
         else
         {
          polygon.lineTo(xvalues[i],yvalues[i]);
         }
        }
       g.fill(polygon);
}//end of fillPolygon

public void drawStar(double x1,double y1,int radius,int side)
{
  double teta=2.0*Math.PI/side;
  double R=radius;
  double r=radius*0.35;
  int xvalues[]=new int[2*side+2];
  int yvalues[]=new int[2*side+2];
  GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
  xvalues.length);
  double teta1,teta2;
  for(int i=0;i<=side;i++)
  {
   teta1=teta*i+Math.PI/2.0;
   teta2=teta/2.0+teta1;
   if(xlog!=0)
   {
   xvalues[2*i]= (int)(Math.floor(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/
       xminmaxlog*abswidth+R*Math.cos(teta1))));
   xvalues[2*i+1]= (int)(Math.floor(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/
       xminmaxlog*abswidth+r*Math.cos(teta2))));
   }
   else
   {
   xvalues[2*i]= (int)(Math.floor(xabsmin+(x1-xmin)/
       (xmax-xmin)*abswidth+R*Math.cos(teta1)));
   xvalues[2*i+1]= (int)(Math.floor(xabsmin+(x1-xmin)/
       (xmax-xmin)*abswidth+r*Math.cos(teta2)));
   }
   if(ylog!=0)
   {
   yvalues[2*i]= (int)(Math.floor(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/
       yminmaxlog*absheight-R*Math.sin(teta1))));
    yvalues[2*i+1]= (int)(Math.floor(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/
       yminmaxlog*absheight-r*Math.sin(teta2))));
   }
   else
   {
   yvalues[2*i]= (int)(Math.floor(yabsmin+absheight-(y1-ymin)/
```

```
        (ymax-ymin)*absheight-R*Math.sin(teta1)));
    yvalues[2*i+1]= (int)(Math.floor(yabsmin+absheight-(y1-ymin)/
        (ymax-ymin)*absheight-r*Math.sin(teta2)));
    }

    if(i==side)
    {
    xvalues[2*i]=xvalues[0];
    yvalues[2*i]=yvalues[0];
    }
  }

  for(int i=0;i<=2*side;i++)
  {
        if(i==0)
        {
        polygon.moveTo(xvalues[0],yvalues[0]);
        }
        else
        {
         polygon.lineTo(xvalues[i],yvalues[i]);
        }
  }
  g.draw(polygon);

}//end of drawStar

public void fillStar(double x1,double y1,int radius,int side)
{
  double teta=2.0*Math.PI/side;
  double R=radius;
  double r=radius*0.35;
  int xvalues[]=new int[2*side+2];
  int yvalues[]=new int[2*side+2];
  GeneralPath polygon=new GeneralPath(GeneralPath.WIND_EVEN_ODD,
  xvalues.length);
  double teta1,teta2;
  for(int i=0;i<=side;i++)
  {
    teta1=teta*i+Math.PI/2.0;
    teta2=teta/2.0+teta1;
    if(xlog!=0)
    {
    xvalues[2*i]= (int)(Math.floor(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/
        xminmaxlog*abswidth+R*Math.cos(teta1))));
    xvalues[2*i+1]= (int)(Math.floor(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/
        xminmaxlog*abswidth+r*Math.cos(teta2))));
    }
    else
    {
    xvalues[2*i]= (int)(Math.floor(xabsmin+(x1-xmin)/
        (xmax-xmin)*abswidth+R*Math.cos(teta1)));
    xvalues[2*i+1]= (int)(Math.floor(xabsmin+(x1-xmin)/
        (xmax-xmin)*abswidth+r*Math.cos(teta2)));
    }
    if(ylog!=0)
    {
    yvalues[2*i]= (int)(Math.floor(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/
```

```
        yminmaxlog*absheight-R*Math.sin(teta1))));
     yvalues[2*i+1]= (int)(Math.floor(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/
        yminmaxlog*absheight-r*Math.sin(teta2))));
    }
    else
    {
    yvalues[2*i]= (int)(Math.floor(yabsmin+absheight-(y1-ymin)/
        (ymax-ymin)*absheight-R*Math.sin(teta1)));
    yvalues[2*i+1]= (int)(Math.floor(yabsmin+absheight-(y1-ymin)/
        (ymax-ymin)*absheight-r*Math.sin(teta2)));
    }

    if(i==side)
    {
    xvalues[2*i]=xvalues[0];
    yvalues[2*i]=yvalues[0];
    }
   }

   for(int i=0;i<=2*side;i++)
   {
        if(i==0)
        {
        polygon.moveTo(xvalues[0],yvalues[0]);
        }
        else
        {
         polygon.lineTo(xvalues[i],yvalues[i]);
        }
   }
   g.fill(polygon);
}//end of drawStar


public void drawXTic(int ticNumber,double ticHeight,int ticSide)
{
       //draw a series of x axis tics
       double dtic=(xmax-xmin)/ticNumber;
       double x1=xmin;
       double y1=ymin;
       double dy;
       dy=ticHeight;
       int xx1,yy1,yy2,yy3;
       for(int i=0;i<=ticNumber;i++)
       {
                if(xlog!=0)
       {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth));}
       else
       {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}
                yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);
                yy2=(int)(yabsmin+absheight-(y1-dy-ymin)/(ymax-ymin)*absheight);
                yy3=(int)(yabsmin+absheight-(y1+dy-ymin)/(ymax-ymin)*absheight);
                if(ticSide==0)
           {g.draw(new Line2D.Double( xx1,yy1,xx1,yy2));}
        else
           {g.draw(new Line2D.Double( xx1,yy1,xx1,yy3));}
           x1=x1+dtic;
       }
```

```
}


public void drawYTic(int ticNumber,double ticWidth,int ticSide)
{
        //draw a series of y axis tics
        double dtic=(ymax-ymin)/ticNumber;
        double x1=xmin;
        double y1=ymin;
        double dx;
        dx=ticWidth;
        double xx1,xx2,xx3,yy1;
        for(int i=0;i<=ticNumber;i++)
        {if(ylog!=0) {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);}
                xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);
                xx2=(int)(xabsmin+(x1-dx-xmin)/(xmax-xmin)*abswidth);
                xx3=(int)(xabsmin+(x1+dx-xmin)/(xmax-xmin)*abswidth);
                  if(ticSide==0)
         {g.draw(new Line2D.Double( xx1,yy1,xx2,yy1));}
         else
         {g.draw(new Line2D.Double( xx1,yy1,xx3,yy1));}
         y1=y1+dtic;
        }
}

public void drawXGrid(int ticNumber)
{
        //draw x gridlies ||||||
        double dtic=(xmax-xmin)/ticNumber;
        double x1=xmin;
        for(int i=0;i<=ticNumber;i++)
        {
         drawLine(0,x1,ymin,x1,ymax);
         x1=x1+dtic;
        }
}

public void drawYGrid(int ticNumber)
{
        //draw y gridlines =====
        double dtic=(ymax-ymin)/ticNumber;
        double y1=ymin;
        for(int i=0;i<=ticNumber;i++)
        {
         drawLine(0,xmin,y1,xmax,y1);
         y1=y1+dtic;
        }
}

public void drawXNumbers(int ticNumber)
{
        //draw x numbers
        Locale en=new Locale("US");
        double dtic=(xmax-xmin)/ticNumber;
        double x1=xmin;
        double y1=ymin;
```

```
        int xx1,yy1;

        String s;
        s="  ";
        for(int i=0;i<=ticNumber;i++)
        {  if(xlog!=0)
           {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-abswidth/200.0));}
           else
           {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth-abswidth/200.0);}
           if(ylog!=0)
           {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight)+absheight/20.0);}
           else
           {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight+absheight/20.0);}
           s=String.format(en,"%4.2g",x1);
          g.drawString(s,xx1,yy1);
          x1=x1+dtic;
        }
}


public void drawYNumbers(int ticNumber)
{
        // draw y  numbers
        Locale en=new Locale("US");
        double dtic=(ymax-ymin)/ticNumber;
        double x1=xmin;
        double y1=ymin;
         int xx1,yy1;
        String s;
        s="  ";
        for(int i=0;i<=ticNumber;i++)
        {
          if(xlog!=0)
           {xx1=(int)(xabsmin+((Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth-abswidth/200.0)-abswidth/10.0);}
           else
           {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth/200.0-abswidth/10.0);}
           if(ylog!=0)
           {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));}
           else
           {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight)+5;}
           s=String.format(en,"%5.3g",y1);
           g.drawString(s,xx1,yy1);
          y1=y1+dtic;
        }
}

public void drawXLabel(String xLabel)
{
    // draw x labels
    double x1=xmin+(xmax-xmin)/2.0;
    double y1=ymin;
     int xx1,yy1;

    if(xlog!=0)
           {xx1=(int)(xabsmin+(Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth);}
           else
           {xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}
           if(ylog!=0)
```

```
        {yy1=(int)(yabsmin+absheight-(Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight+abswidth/12.0);}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight+abswidth/12.0);}
    g.drawString(xLabel,xx1,yy1);
}

public void drawYLabel(String yLabel)
{
    // draw y labels
    double x1;
    double y1=ymax-(ymax-ymin)/2.0;
     int xx1,yy1;
        if(xlog!=0)
        {x1=xmin-2.7;xx1=(int)(xabsmin+(Math.log10(x1-xmin+10.0)-1.0)/xminmaxlog*abswidth);}
        else
        {x1=xmin-(xmax-xmin)/6.0;xx1=(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth);}
        if(ylog!=0)
        {yy1=(int)(yabsmin+absheight-((Math.log10(y1-ymin+10.0)-1.0)/yminmaxlog*absheight));}
        else
        {yy1=(int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight);}
    int n=yLabel.length();
    char ch[]=new char[n];
    yLabel.getChars(0,n,ch,0);
    TextLayout t=new TextLayout(yLabel,g.getFont(),g.getFontRenderContext());
    AffineTransform at=AffineTransform.getRotateInstance(3.0*Math.PI/2.0,xx1,yy1);
    g.setTransform(at);
    t.draw(g,xx1,yy1);
    at=AffineTransform.getRotateInstance(0,xx1,yy1);
    g.setTransform(at);
}

public void drawLabel(String Label)
{
    //draw graphic label
    double x1=xmin+(xmax-xmin)/2.0;
    double y1=ymax;
    g.drawString(Label,(int)(xabsmin+(x1-xmin)/(xmax-xmin)*abswidth),
         (int)(yabsmin+absheight-(y1-ymin)/(ymax-ymin)*absheight)-(int)(abswidth/40.0));
}

public void drawAdditionalLabel(String s,double x1,double y1)
{
    //draw Additional graphic label
   drawString(s,x1,y1);
}

    public void drawMouseTic(double xi[],double yi[],int xn)
    {
                for(int i=0;i<xn;i++)
                {
                        if((xi[i]>=xmin && xi[i]<=xmax) && (yi[i]>=ymin && yi[i]<=ymax)) {
drawStar(xi[i],yi[i],(int)(abswidth/80.0),5); }
        }
    }

public void drawPlotLines(int i,int plottype[],double x[][],double y[][],int n[],char ch[])
{
    int j;
```

```
//draw lines
    if((plottype[i] >= 0) && (plottype[i] < 10) )
    {
                switch ( plottype[i] )
      {
       case 0 : g.setStroke(new BasicStroke(1.0f));break;
       case 1 : g.setStroke(d1);break;
                 case 2 : g.setStroke(d2);break;
       case 3 : g.setStroke(d3);break;
       case 4 : g.setStroke(d4);break;
       case 5 : g.setStroke(new BasicStroke(2.0f));break;
       case 6 : g.setStroke(d5);break;
       case 7 : g.setStroke(d6);break;
       case 8 : g.setStroke(d7);break;
       case 9 : g.setStroke(d8);break;
       };
      for(j=0;j<n[i]-1;j++)
       {
       if((x[i][j]>=xmin && x[i][j]<=xmax )
         && (y[i][j]>=ymin && y[i][j]<=ymax))
       {
       if((x[i][j+1]>=xmin && x[i][j+1]<=xmax) &&
          (y[i][j+1]>=ymin && y[i][j+1]<=ymax))
          {
          drawLine(0,x[i][j],y[i][j],x[i][j+1],y[i][j+1]);
          }
       else if(x[i][j+1]>xmax)
          {
             double b=(y[i][j+1]-y[i][j])/(x[i][j+1]-x[i][j]);
             double  a=y[i][j]-b*x[i][j];
             drawLine(0,x[i][j],y[i][j],xmax,(a+b*xmax));
          }
       else if(y[i][j+1]>ymax)
          {
             double b=(y[i][j+1]-y[i][j])/(x[i][j+1]-x[i][j]);
             double a=y[i][j]-b*x[i][j];
             drawLine(0,x[i][j],y[i][j],(ymax-a)/b,ymax);
          }
       else if(x[i][j+1]>xmax && y[i][j+1]>ymax)
          {
             double b=(y[i][j+1]-y[i][j])/(x[i][j+1]-x[i][j]);
             double a=y[i][j]-b*x[i][j];
             drawLine(0,x[i][j],y[i][j],(ymax-a)/b,(a+b*xmax));
          }
       }
       else if((x[i][j+1]>=xmin && x[i][j+1]<=xmax )
         && (y[i][j+1]>=ymin && y[i][j+1]<=ymax))
       {
          if(x[i][j]<xmin)
          {
             double b=(y[i][j+1]-y[i][j])/(x[i][j+1]-x[i][j]);
             double  a=y[i][j+1]-b*x[i][j+1];
             drawLine(0,xmin,(a+b*xmin),x[i][j+1],y[i][j+1]);
          }
          if(y[i][j]<ymin)
          {
             double b=(y[i][j+1]-y[i][j])/(x[i][j+1]-x[i][j]);
             double  a=y[i][j+1]-b*x[i][j+1];
```

```
            drawLine(0,(ymin-a)/b,ymin,x[i][j+1],y[i][j+1]);
          }
        }
        }//end of for(j=0
      g.setStroke(new BasicStroke(1.0f));
      }//end of if(plottype
//draw characters
    else if(plottype[i]>=10 && plottype[i]<=19)
      {
        for(j=0;j<n[i];j++)
        {
          if((x[i][j]>=xmin && x[i][j]<=xmax )
          && (y[i][j]>=ymin && y[i][j]<=ymax))
            { drawChar(ch[i],x[i][j],y[i][j]); }
        }//end of for(j=0;
      }//end else if(plottype[i]==10..19)
//draw rectangles
    else if(plottype[i]==20)
      {
        for(j=0;j<n[i];j++)
        {
          if((x[i][j]>=xmin && x[i][j]<=xmax )
          && (y[i][j]>=ymin && y[i][j]<=ymax))
              { drawRect(x[i][j],y[i][j],abswidth/100,abswidth/80); }
        }//end of for(j=0;
      }//end else if(plottype[i]==20)
      else if(plottype[i]==21)
      {
        for(j=0;j<n[i];j++)
        {
          if((x[i][j]>=xmin && x[i][j]<=xmax )
          && (y[i][j]>=ymin && y[i][j]<=ymax))
              { fillRect(x[i][j],y[i][j],abswidth/100,abswidth/80); }
        }//end of for(j=0;
      }//end else if(plottype[i]==20)
//draw circle
    else if(plottype[i]==22)
      {
        for(j=0;j<n[i];j++)
        {
          if((x[i][j]>=xmin && x[i][j]<=xmax )
          && (y[i][j]>=ymin && y[i][j]<=ymax))
            { drawEllipse(x[i][j],y[i][j],abswidth/100,abswidth/80); }
        }//end of for(j=0;
      }//end else if(plottype[i]==22)
    else if(plottype[i]==23)
      {
        for(j=0;j<n[i];j++)
        {
          if((x[i][j]>=xmin && x[i][j]<=xmax )
          && (y[i][j]>=ymin && y[i][j]<=ymax))
            { fillEllipse(x[i][j],y[i][j],abswidth/100,abswidth/80); }
        }//end of for(j=0;
      }//end else if(plottype[i]==21)
    else if(plottype[i]>=24 && plottype[i]<=27)
      {

        for(j=0;j<n[i];j++)
```

```
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
                    drawPolygon(x[i][j],y[i][j],abswidth/100,(plottype[i]-21));
          }
      }//end of for(j=0;
    }//end else if(plottype[i]==24..27)
    else if(plottype[i]>=28 && plottype[i]<=31)
    {

      for(j=0;j<n[i];j++)
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
                    fillPolygon(x[i][j],y[i][j],abswidth/100,(plottype[i]-25));
          }
      }//end of for(j=0;
    }//end else if(plottype[i]==24..27)
  else if(plottype[i]>=32 && plottype[i]<=35)
    {

      for(j=0;j<n[i];j++)
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
                    drawStar(x[i][j],y[i][j],abswidth/100,(plottype[i]-29));
          }
      }//end of for(j=0;
    }
  else if(plottype[i]>=36 && plottype[i]<=39)
    {

      for(j=0;j<n[i];j++)
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
                    fillStar(x[i][j],y[i][j],abswidth/100,(plottype[i]-33));
          }
      }//end of for(j=0;
    }
  else if(plottype[i]==40)
    {

      for(j=0;j<n[i];j++)
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
                    drawBar(x[i][j],y[i][j]);
          }
      }//end of for(j=0;
    }
  else if(plottype[i]==41)
    {

      for(j=0;j<n[i];j++)
      { if((x[i][j]>=xmin && x[i][j]<=xmax )
       && (y[i][j]>=ymin && y[i][j]<=ymax))
          {
```

```
                    fillBar(x[i][j],y[i][j]);
            }
        }//end of for(j=0;
      }
}

}
```

21.2 PLOTTING THREE DIMENSIONAL DATA AND FUNCTIONS
3D PLOTS BY USING VISAD PACKAGE

In order to create 3 dimensional plots in java a free distributed graphic package VisAD is used. VisAD is a professional level graphic package for 3 dimensional graphic applications. You can download the latest VisAD package from internet adres http://www.ssec.wisc.edu/~billh/visad.html**.** Open codes are available if you desire the compile the package yourself or you can used precompiled jar file. For your compiler to see and use the package copy visad.jar file into jre/lib/ext directory of your java compiler. To utilise this program you also need java 3D graphic package. This package can be obtained from http://www.java.sun.com**.** Java 3D package use openGL graphic library as default. In some older computers, your graphic card may not support openGL library, if your card is not supporting openGL, you can use DirectX graphic system. In this case you should tell java compiler you are using directX. You achive this by using java command as :

java -Dj3d.rend=d3d classname

You can find the latest directX version from the adres http://www.microsoft.com/windows/directx/. An interphase program to run our standart 3D function definitions in VisAD graphic environment, an interphase program is developed.

Program 21.2-1 Plot3D VisAD graphic package interphase

```
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

class f1 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=(x[1]-x[0]*x[0])*(x[1]-x[0]*x[0])+0.01*(x[0]-1.0)*(x[0]-1.0);
return -ff;
}
}

public class Plot3D{

  public static double[] grid(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int NROWS)
  {  //create 1D Z grid for Z value from a given function value
            float[][] zi = new float[2][NCOLS * NROWS];
   float xx[][]=new float[2][NCOLS * NROWS];
   double zz[]=new double[NCOLS * NROWS];
   double x[]=new double[2];
   for(int c = 0; c < NCOLS; c++)
     for(int r = 0; r < NROWS; r++)
     { xx[0][ c * NROWS + r ]=(float)(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
       x[0]=(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
```

```
        xx[1][ c * NROWS + r ]=(float)(x2min+(x2max-x2min)/(float)(NROWS-1)*r);
        x[1]=(x2min+(x2max-x2min)*r/(float)(NROWS-1));
        zz[ c * NROWS + r ]=f.func(x);
      }
    return zz;
   }

   public static double[][] grid2D(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int NROWS)
   {  //create 2D Z grid for Z value from a given function value

   float[][] zi = new float[2][NCOLS * NROWS];
    float xx[][]=new float[2][NCOLS * NROWS];
    double zz[][]=new double[NCOLS][NROWS];
    double x[]=new double[2];
       for(int c = 0; c < NCOLS; c++)
          for(int r = 0; r < NROWS; r++)
      { xx[0][ c * NROWS + r ]=(float)(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
        x[0]=(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
        xx[1][ c * NROWS + r ]=(float)(x2min+(x2max-x2min)/(float)(NROWS-1)*r);
        x[1]=(x2min+(x2max-x2min)*r/(float)(NROWS-1));
        zz[c][r]=f.func(x);
      }
    return zz;
   }

   public static void plot3D_CD2(double zz[][],double x1min,double x1max,double x2min,double x2max,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   {
   RealType longitude, latitude;
   RealType altitude;
  // Tuple to pack longitude and latitude together
   RealTupleType domain_tuple;
  // The function (domain_tuple -> altitude )
   FunctionType func_domain_alt;
   // Our Data values for the domain are represented by the Set
   Set domain_set;
  // The Data class FlatField
   FlatField vals_ff;
  // The DataReference from data to display
   DataReferenceImpl data_ref;
  // The 2D display, and its the maps
   DisplayImpl display;
   ScalarMap latMap, lonMap;
   ScalarMap altMap, altRGBMap;
   ScalarMap altAlphaMap;

   ColorControl colCont;

  // Our color table

   float[][] myColorTable;

    // Create the quantities
    // Use RealType(String name, Unit unit, Set set);

    latitude = RealType.getRealType(sx, SI.meter, null);
    longitude = RealType.getRealType(sy, SI.meter, null);
    altitude = RealType.getRealType("z", SI.meter, null);
```

```
domain_tuple = new RealTupleType(latitude, longitude);
// Create a FunctionType (domain_tuple -> range_tuple )
// Use FunctionType(MathType domain, MathType range)
func_domain_alt = new FunctionType( domain_tuple, altitude);
// Create the domain Set, with 12 columns and 6 rows, using an
// LinearDSet(MathType type, double first1, double last1, int lengthX,
//                                          double first2, double last2, int lengthY)
// note the "inverted" first and last values of latitude
int NCOLS = zz[0].length;;
int NROWS = zz.length;
domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                        x2min, x2max, NCOLS);
// The actal altitudes values
// ordered as float[ NROWS ][ NCOLS ]
double[][] alt_samples = zz;
// Our 'flat' array
double[][] flat_samples = new double[1][NCOLS * NROWS];
// Fill our 'flat' array with the altitude values
// by looping over NCOLS and NROWS
// Note the use of an index variable, indicating the order of the samples
int index = 0;
double zmin=2.0e60,zmax=-2e60;;
for(int c = 0; c < NCOLS; c++)
  for(int r = 0; r < NROWS; r++)
  {
            flat_samples[0][ index ] = alt_samples[c][r];
            if(flat_samples[0][ index ]<zmin) zmin=flat_samples[0][ index ];
            if(flat_samples[0][ index ]>zmax) zmax=flat_samples[0][ index ];
            index++;
  }
// Create a FlatField
// Use FlatField(FunctionType type, Set domain_set)
vals_ff = new FlatField( func_domain_alt, domain_set);
// ...and put the altitude values above into it
// Note the argument false, meaning that the array won't be copied
vals_ff.setSamples( flat_samples , false );
// Create Display and its maps
display = new DisplayImplJ3D("display1");
// Get display's graphics mode control and draw scales
GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
dispGMC.setScaleEnable(true);
// Also enable Texture
//dispGMC.setTextureEnable(false);
dispGMC.setTextureEnable(true);
// Create the ScalarMaps: latitude to XAxis, longitude to YAxis and
// altitude to ZAxis and to RGB
// Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
latMap = new ScalarMap( latitude,    Display.XAxis );
lonMap = new ScalarMap( longitude, Display.YAxis );
altRGBMap = new ScalarMap( altitude,  Display.RGB );
altMap = new ScalarMap( altitude,  Display.ZAxis );
altAlphaMap = new ScalarMap( altitude,  Display.Alpha );
// Add maps to display
display.addMap( latMap );
display.addMap( lonMap );

display.addMap( altMap );
display.addMap( altRGBMap );
```

```
    display.addMap( altAlphaMap );
    // Create a different color table
    // Note: table has red, green and blue components
    // and is 10 units long, i.e float[3][10]
    int tableLength = 10;
    myColorTable = new float[3][tableLength];
    for(int i=0;i<tableLength;i++){
      myColorTable[0][i]= (float) 1.0f - (float)i / ((float)tableLength-1.0f); // red component
      myColorTable[1][i]= (float) (float)i / ((float)tableLength-1.0f); // green component
      myColorTable[2][i]= (float) 0.50f; // blue component
    }
    // Force top of table to be white
    myColorTable[0][9]=1.0f;
    myColorTable[1][9]=1.0f;
    myColorTable[2][9]=1.0f;
    // Get the ColorControl from the altitude RGB map
    colCont = (ColorControl) altRGBMap.getControl();
    // Set the table
    colCont.setTable(myColorTable );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    display.addReference( data_ref);
    // Set maps ranges
    latMap.setRange(x1min, x1max);
    altMap.setRange(zmin, zmax);
    // Create application window and add display to window
    JFrame jframe = new JFrame("Plot3D");
    jframe.getContentPane().add(display.getComponent());
    // Add the LabeledColorWidget to the frame
    // Set window size and make it visible
    jframe.setSize(300, 300);
    jframe.setVisible(true);
   }
   public static void plot3D_CD1(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int
NROWS,int NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   { plot3D_CD1(grid2D(f,x1min,x1max,x2min,x2max,NCOLS,
NROWS),x1min,x1max,x2min,x2max,NCONTOUR,sx,sy,sz);}

   public static void plot3D_CD1(double zz[][],double x1min,double x1max,double x2min,double x2max,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   {
    RealType longitude, latitude;
    RealType altitude;
   // Tuple to pack longitude and latitude together
    RealTupleType domain_tuple;
  // The function (domain_tuple -> altitude )
   FunctionType func_domain_alt;
   // Our Data values for the domain are represented by the Set
   Set domain_set;
  // The Data class FlatField
   FlatField vals_ff;
  // The DataReference from data to display
   DataReferenceImpl data_ref;
  // The 2D display, and its the maps

   DisplayImpl display;
   ScalarMap latMap, lonMap;
```

```
ScalarMap altMap, altRGBMap;
ColorControl colCont;
// Our color table
float[][] myColorTable;
 // Create the quantities
 // Use RealType(String name, Unit unit, Set set);
 latitude = RealType.getRealType(sx, SI.meter);
 longitude = RealType.getRealType(sy, SI.meter);
 altitude  = RealType.getRealType("z", SI.meter);
 domain_tuple = new RealTupleType(latitude, longitude);
 //String ss=sz;
 //altitude = RealType.getRealType("z", SI.meter);
 // Create a FunctionType (domain_tuple -> range_tuple )
 // Use FunctionType(MathType domain, MathType range)
 func_domain_alt = new FunctionType( domain_tuple, altitude);
 // Create the domain Set, with 12 columns and 6 rows, using an
 // LinearDSet(MathType type, double first1, double last1, int lengthX,
 //                                        double first2, double last2, int lengthY)
 // note the "inverted" first and last values of latitude
 int NCOLS = zz.length;
 int NROWS = zz[0].length;
 domain_set = new Linear2DSet(domain_tuple, x1min,x1max, NROWS,
                                                        x2min, x2max, NCOLS);
 // The actal altitudes values
 // ordered as float[ NROWS ][ NCOLS ]
 double[][] alt_samples = zz;
 double[][] flat_samples = new double[1][NCOLS * NROWS];
 int index = 0;
 double zmin=2.0e60,zmax=-2e60;;
 for(int c = 0; c < NCOLS; c++)
   for(int r = 0; r < NROWS; r++)
   {
            flat_samples[0][ index ] = alt_samples[c][r];
            if(flat_samples[0][ index ]<zmin) zmin=flat_samples[0][ index ];
            if(flat_samples[0][ index ]>zmax) zmax=flat_samples[0][ index ];
            index++;
   }
 // Create a FlatField
 // Use FlatField(FunctionType type, Set domain_set)
 vals_ff = new FlatField( func_domain_alt, domain_set);
 FlatField der_vals_ff ;//= new FlatField( func_domain_alt, domain_set);
 // ...and put the altitude values above into it
 // Note the argument false, meaning that the array won't be copied
 vals_ff.setSamples( flat_samples , false );
 der_vals_ff = (FlatField) vals_ff.derivative( longitude, Data.NO_ERRORS );//.setSamples( flat_samples , false );
 // Create Display and its maps
 display = new DisplayImplJ3D("display1");
 // Get display's graphics mode control and draw scales
 GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
 dispGMC.setScaleEnable(true);
 // Also enable Texture
 dispGMC.setTextureEnable(false);
 // Create the ScalarMaps: latitude to XAxis, longitude to YAxis and
 // altitude to ZAxis and to RGB
 // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
 latMap = new ScalarMap( latitude,    Display.XAxis );
 lonMap = new ScalarMap( longitude, Display.YAxis );
 altRGBMap = new ScalarMap( altitude,  Display.RGB );
```

```
    altMap = new ScalarMap( altitude,  Display.ZAxis );
    // Add maps to display
    display.addMap( latMap );
    display.addMap( lonMap );
    display.addMap( altMap );
    display.addMap( altRGBMap );
    // Create a color table
    // Note: table has red, green and blue components
    // and is 8 units long, i.e float[3][8]
    myColorTable = new float[][]{{0.0f, 1.0f, 0.0f, 0.0f, 0.0f, 1.0f, 1.0f, 1.0f},  // red component
                                 {0.0f, 0.0f, 1.0f, 0.0f, 1.0f, 0.0f, 1.0f, 1.0f},  // green component
                                 {0.0f, 0.0f, 0.0f, 1.0f, 1.0f, 1.0f, 0.0f, 1.0f}}; // blue component
    // Get the ColorControl from the altitude RGB map
    colCont = (ColorControl) altRGBMap.getControl();
    // Set the table
    colCont.setTable(myColorTable );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    display.addReference( data_ref);
    // Set maps ranges
    latMap.setRange(x1min, x1max);
    lonMap.setRange(x2min, x2max);
    altMap.setRange(zmin, zmax);
    // Create application window and add display to window
    JFrame jframe = new JFrame(sz);
    jframe.getContentPane().add(display.getComponent());
    // Add the LabeledColorWidget to the frame
    // Set window size and make it visible
    jframe.setSize(300, 300);
    jframe.setVisible(true);
   }

   public static void plot3D_CD(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int NROWS,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   { plot3D_CD(grid2D(f,x1min,x1max,x2min,x2max,NCOLS,
NROWS),x1min,x1max,x2min,x2max,NCONTOUR,sx,sy,sz);}


   public static void plot3D_CD(double zz[][],double x1min,double x1max,double x2min,double x2max,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   {
   //===============
   // Declare variables
   // The domain quantities longitude and latitude
   // and the dependent quantity altitude
   RealType longitude, latitude;
   RealType altitude;
  // Tuple to pack longitude and latitude together
   RealTupleType domain_tuple;
  // The function (domain_tuple -> altitude )
   FunctionType func_domain_alt;
   // Our Data values for the domain are represented by the Set
   Set domain_set;


  // The Data class FlatField
   FlatField vals_ff;
```

```
// The DataReference from data to display
 DataReferenceImpl data_ref;
// The 2D display, and its the maps
 DisplayImpl display;
 ScalarMap latMap, lonMap;
 ScalarMap altMap, altRGBMap;
 ColorControl colCont;
// Our color table
 float[][] myColorTable;
  // Create the quantities
  // Use RealType(String name, Unit unit, Set set);
  latitude = RealType.getRealType(sx, SI.meter, null);
  longitude = RealType.getRealType(sy, SI.meter, null);
  domain_tuple = new RealTupleType(latitude, longitude);
  altitude = RealType.getRealType("z", SI.meter, null);
  // Create a FunctionType (domain_tuple -> range_tuple )
  // Use FunctionType(MathType domain, MathType range)
  func_domain_alt = new FunctionType( domain_tuple, altitude);
  // Create the domain Set, with 12 columns and 6 rows, using an
  // LinearDSet(MathType type, double first1, double last1, int lengthX,
  //                                          double first2, double last2, int lengthY)
  // note the "inverted" first and last values of latitude
  // The actal altitudes values
  // ordered as float[ NROWS ][ NCOLS ]
   double[][] alt_samples =zz;
   int NROWS=alt_samples.length;
   int NCOLS=alt_samples[0].length;
    domain_set = new Linear2DSet(domain_tuple, x1min,x1max, NROWS,
                                                          x2min,  x2max, NCOLS);
  // Our 'flat' array
  double[][] flat_samples = new double[1][NCOLS * NROWS];
  // Fill our 'flat' array with the altitude values
  // by looping over NCOLS and NROWS
  // Note the use of an index variable, indicating the order of the samples
  int index = 0;
  for(int c = 0; c < NCOLS; c++)
    for(int r = 0; r < NROWS; r++){
            // set altitude altitude
            flat_samples[0][ index ] = alt_samples[r][c];
            // increment index
            index++;
    }

  // Create a FlatField
  // Use FlatField(FunctionType type, Set domain_set)
  vals_ff = new FlatField( func_domain_alt, domain_set);
  // ...and put the altitude values above into it
  // Note the argument false, meaning that the array won't be copied
  vals_ff.setSamples( flat_samples , false );
  // Code for slope data
    // Data in a FlatField
    FlatField slope_vals_ff ;
    // Calculate derivative of altitude with respect to longitude; assume no errors
    slope_vals_ff = (FlatField) vals_ff.derivative( longitude, Data.NO_ERRORS );

    // Get the funtionc from the FlatField for slope
    FunctionType func_domain_slope = ((FunctionType)slope_vals_ff.getType());
    // "slope" is a RealType;
```

```
    RealType slope = (RealType) func_domain_slope.getRange();
    // Create a ScalarMap to color slope's surface
    ScalarMap slopeRGBMap = new ScalarMap( slope,  Display.RGB );
  // Create Display and its maps
  display = new DisplayImplJ3D("display1");
  // Get display's graphics mode control and draw scales
  GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
  dispGMC.setScaleEnable(true);
  // Also enable Texture
  dispGMC.setTextureEnable(false);
  // Create the ScalarMaps: latitude to XAxis, longitude to YAxis and
  // altitude to ZAxis and to RGB
  // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
  latMap = new ScalarMap( latitude,    Display.XAxis );
  lonMap = new ScalarMap( longitude, Display.YAxis );
  altRGBMap = new ScalarMap( altitude,  Display.RGB );
  altMap = new ScalarMap( altitude,  Display.ZAxis );
  // Add maps to display
  display.addMap( latMap );
  display.addMap( lonMap );
  display.addMap( altMap );
  display.addMap( altRGBMap );
  display.addMap( slopeRGBMap );
  // Create a color table
  // Note: table has red, green and blue components
  // and is 8 units long, i.e float[3][8]
  myColorTable = new float[][]{{0, 1, 0, 0, 0, 1, 1, 1},  // red component
                                                {0, 0, 1, 0, 1, 0, 1, 1},  // green component
                                                        {0, 0, 0, 1, 1, 1, 0, 1}}; // blue component
  // Get the ColorControl from the altitude RGB map
  colCont = (ColorControl) altRGBMap.getControl();
  // Set the table
  colCont.setTable(myColorTable );
  // Create a data reference and set the FlatField as our data
  data_ref = new DataReferenceImpl("data_ref");
  data_ref.setData( vals_ff );
  // Data reference for slope
  DataReferenceImpl data_ref2 = new DataReferenceImpl("data_ref2");
  data_ref2.setData( slope_vals_ff );
   // Add reference to display
  ConstantMap[] constAlpha_CMap = {
                new ConstantMap( 0.50f, Display.Alpha)};
  ConstantMap[] constZ_CMap = {
                new ConstantMap( -1.0f, Display.ZAxis)};
             // Add references to display
// order MATTERS!
              display.addReference( data_ref2, constZ_CMap );
 display.addReference( data_ref, constAlpha_CMap );
  // Set maps ranges
  latMap.setRange(x1min, x1max);
  lonMap.setRange(x2min, x2max);
  //altMap.setRange(zmin, zmax);
  // Create application window and add display to window
  JFrame jframe = new JFrame(sz);
  jframe.getContentPane().add(display.getComponent());
  // Add the LabeledColorWidget to the frame
  // Set window size and make it visible
  jframe.setSize(300, 300);
```

```
    jframe.setVisible(true);
    }
   //================
   public static void plot3D_RGB(double zz[][],double x1min,double x1max,double x2min,double x2max,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
   {
   int NROWS=zz[0].length;
   int NCOLS=zz.length;
   RealType x1, x2;
   RealType z, temperature,  precipitation;
  // Two Tuples: one to pack longitude and latitude together, as the domain
  // and the other for the range (altitude, temperature)
   RealTupleType domain_tuple, range_tuple;
  // The function (domain_tuple -> range_tuple )
   FunctionType func_domain_range;
   // Our Data values for the domain are represented by the Set
   Set domain_set;
  // The Data class FlatField
   FlatField vals_ff;
  // The DataReference from data to display
   DataReferenceImpl data_ref;
  // The 2D display, and its the maps
   DisplayImpl display;
   ScalarMap latMap, lonMap;
   ScalarMap altZMap, tempRGBMap;
   ScalarMap tempZMap, altRGBMap;
   ScalarMap tempIsoMap;
    // Create the quantities
    // Use RealType(String name);
    x1 = RealType.getRealType(sx,null,null);
    x2 = RealType.getRealType(sy,null,null);
    domain_tuple = new RealTupleType(x1, x2);
    temperature = RealType.getRealType("temperature",null,null);
    z = RealType.getRealType("z",null,null);
    range_tuple = new RealTupleType( z, temperature  );
    // Create a FunctionType (domain_tuple -> range_tuple )
    // Use FunctionType(MathType domain, MathType range)
    func_domain_range = new FunctionType( domain_tuple, range_tuple);
    // Create the domain Set
    // Use LinearDSet(MathType type, double first1, double last1, int lengthX,
    //                                          double first2, double last2, int lengthY)
    domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                        x2min, x2max, NCOLS);
    // Get the Set samples to facilitate the calculations
    float[][] x_j = domain_set.getSamples( true );
    // We create another array, with the same number of elements of
    // altitude and temperature, but organized as
    // float[2][ number_of_samples ]
    float[][] zi = new float[2][NCOLS*NROWS];
    // ...and then we fill our 'flat' array with the generated values
    // by looping over NCOLS and NROWS
    double xx[]=new double[2];
    for(int c = 0; c < NCOLS; c++)
      for(int r = 0; r < NROWS; r++)
      {xx[0]=(float)(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
       xx[1]=(float)(x2min+(x2max-x2min)*r/(float)(NROWS-1));
       zi[0][ c * NROWS + r ] = (float)zz[c][r];
                zi[1][ c * NROWS + r ] = (float)xx[1];
```

```
    }
    // Create a FlatField
    // Use FlatField(FunctionType type, Set domain_set)
    vals_ff = new FlatField( func_domain_range, domain_set);
    // ...and put the values above into it
    // Note the argument false, meaning that the array won't be copied
    vals_ff.setSamples( zi , false );
    // Create Display and its maps
    // A 2D display
    display = new DisplayImplJ3D(sz);
    // Get display's graphics mode control and draw scales
    GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
    dispGMC.setScaleEnable(true);
    // Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
    // altitude to ZAxis and temperature to RGB
    // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
    latMap = new ScalarMap( x1,Display.XAxis );
    lonMap = new ScalarMap( x2,Display.YAxis );
    // Add maps to display
    display.addMap( latMap );
    display.addMap( lonMap );
     // altitude to z-axis and temperature to color
    altZMap = new ScalarMap( z,  Display.ZAxis );
    tempRGBMap = new ScalarMap( temperature,  Display.RGB );
    tempIsoMap = new ScalarMap( temperature,  Display.IsoContour );
    display.addMap( tempRGBMap );
    display.addMap( tempIsoMap );
    // Add maps to display
    display.addMap( altZMap );
    // Uncomment following lines to have different data depiction
    // temperature to z-axis and altitude to color
    //altRGBMap = new ScalarMap( zi,  Display.RGB );
    //tempZMap = new ScalarMap( temperature,  Display.ZAxis );
    //display.addMap( altRGBMap );
    //display.addMap( tempZMap );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    // Add reference to display
    display.addReference( data_ref );
    // Create application window and add display to window
    JFrame jframe = new JFrame(sz);
    jframe.getContentPane().add(display.getComponent());
    // Set window size and make it visible
    jframe.setSize(300, 300);
    jframe.setVisible(true);
    }
   public static void plot3D_RGB(double zz[],double x1min,double x1max,double x2min,double x2max,int NCOLS,int
NROWS,int NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
    {
    RealType x1, x2;
    RealType z, temperature,  precipitation;
   // Two Tuples: one to pack longitude and latitude together, as the domain
   // and the other for the range (altitude, temperature)
    RealTupleType domain_tuple, range_tuple;
   // The function (domain_tuple -> range_tuple )
    FunctionType func_domain_range;
    // Our Data values for the domain are represented by the Set
```

```
 Set domain_set;
// The Data class FlatField
 FlatField vals_ff;
// The DataReference from data to display
 DataReferenceImpl data_ref;
// The 2D display, and its the maps
 DisplayImpl display;
 ScalarMap latMap, lonMap;
 ScalarMap altZMap, tempRGBMap;
 ScalarMap tempZMap, altRGBMap;
 ScalarMap tempIsoMap;
  // Create the quantities
  // Use RealType(String name);
  x1 = RealType.getRealType(sx,null,null);
  x2 = RealType.getRealType(sy,null,null);
  domain_tuple = new RealTupleType(x1, x2);
  temperature = RealType.getRealType("temperature",null,null);
  z = RealType.getRealType("z",null,null);
  range_tuple = new RealTupleType( z, temperature  );
  // Create a FunctionType (domain_tuple -> range_tuple )
  // Use FunctionType(MathType domain, MathType range)
  func_domain_range = new FunctionType( domain_tuple, range_tuple);
  // Create the domain Set
  // Use LinearDSet(MathType type, double first1, double last1, int lengthX,
  //                                          double first2, double last2, int lengthY)
  domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                    x2min, x2max, NCOLS);
  // Get the Set samples to facilitate the calculations
  float[][] x_j = domain_set.getSamples( true );
  // We create another array, with the same number of elements of
  // altitude and temperature, but organized as
  // float[2][ number_of_samples ]
  float[][] zi = new float[2][NCOLS*NROWS];
  // ...and then we fill our 'flat' array with the generated values
  // by looping over NCOLS and NROWS
  double xx[]=new double[2];
  for(int c = 0; c < NCOLS; c++)
    for(int r = 0; r < NROWS; r++)
    {xx[0]=(float)(x1min+(x1max-x1min)*c/(float)(NCOLS-1));
     xx[1]=(float)(x2min+(x2max-x2min)*r/(float)(NROWS-1));
     zi[0][ c * NROWS + r ] = (float)zz[ c * NROWS + r ];
              zi[1][ c * NROWS + r ] = (float)xx[1];
  }
  // Create a FlatField
  // Use FlatField(FunctionType type, Set domain_set)
  vals_ff = new FlatField( func_domain_range, domain_set);
  // ...and put the values above into it
  // Note the argument false, meaning that the array won't be copied
  vals_ff.setSamples( zi , false );
  // Create Display and its maps
  // A 2D display
  display = new DisplayImplJ3D(sz);
  // Get display's graphics mode control and draw scales
  GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
  dispGMC.setScaleEnable(true);
  // Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
  // altitude to ZAxis and temperature to RGB
  // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
```

```
latMap = new ScalarMap( x1,Display.YAxis );
lonMap = new ScalarMap( x2,Display.XAxis );
// Add maps to display
display.addMap( latMap );
display.addMap( lonMap );
 // altitude to z-axis and temperature to color
altZMap = new ScalarMap( z,  Display.ZAxis );
tempRGBMap = new ScalarMap( temperature,  Display.RGB );
tempIsoMap = new ScalarMap( temperature,  Display.IsoContour );
display.addMap( tempRGBMap );
display.addMap( tempIsoMap );
// Add maps to display
display.addMap( altZMap );
// Uncomment following lines to have different data depiction
// temperature to z-axis and altitude to color
altRGBMap = new ScalarMap( z,  Display.RGB );
tempZMap = new ScalarMap( temperature,  Display.ZAxis );
display.addMap( altRGBMap );
display.addMap( tempZMap );
// Create a data reference and set the FlatField as our data
data_ref = new DataReferenceImpl("data_ref");
data_ref.setData( vals_ff );
// Add reference to display
display.addReference( data_ref );
// Create application window and add display to window
JFrame jframe = new JFrame(sz);
jframe.getContentPane().add(display.getComponent());
// Set window size and make it visible
jframe.setSize(300, 300);
jframe.setVisible(true);
}
public static void plot3D_RGB(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int
NROWS,int NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
{
RealType x1, x2;
RealType z, temperature,  precipitation;
// Two Tuples: one to pack longitude and latitude together, as the domain
// and the other for the range (altitude, temperature)
RealTupleType domain_tuple, range_tuple;
// The function (domain_tuple -> range_tuple )
FunctionType func_domain_range;
 // Our Data values for the domain are represented by the Set
Set domain_set;
// The Data class FlatField
FlatField vals_ff;
// The DataReference from data to display
DataReferenceImpl data_ref;
// The 2D display, and its the maps
DisplayImpl display;
ScalarMap latMap, lonMap;
ScalarMap altZMap, tempRGBMap;
ScalarMap tempZMap, altRGBMap;
ScalarMap tempIsoMap;
 // Create the quantities
 // Use RealType(String name);
 x1 = RealType.getRealType(sx,null,null);
 x2 = RealType.getRealType(sy,null,null);
 domain_tuple = new RealTupleType(x1, x2);
```

```
temperature = RealType.getRealType("temperature",null,null);
z = RealType.getRealType("z",null,null);
range_tuple = new RealTupleType( z, temperature  );
// Create a FunctionType (domain_tuple -> range_tuple )
// Use FunctionType(MathType domain, MathType range)
func_domain_range = new FunctionType( domain_tuple, range_tuple);
// Create the domain Set
// Use LinearDSet(MathType type, double first1, double last1, int lengthX,
//                                                double first2, double last2, int lengthY)
domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                        x2min, x2max, NCOLS);
// Get the Set samples to facilitate the calculations
float[][] x_j = domain_set.getSamples( true );
// We create another array, with the same number of elements of
// altitude and temperature, but organized as
// float[2][ number_of_samples ]
float[][] zi = new float[2][NCOLS * NROWS];
// ...and then we fill our 'flat' array with the generated values
// by looping over NCOLS and NROWS
double xx[]=new double[2];
for(int c = 0; c < NCOLS; c++)
  for(int r = 0; r < NROWS; r++)
  {xx[0]=x_j[0][ c * NROWS + r ];
   xx[1]=x_j[1][ c * NROWS + r ];
   zi[0][ c * NROWS + r ] = (float)f.func(xx);
            zi[1][ c * NROWS + r ] = x_j[1][ c * NROWS + r ];
}
// Create a FlatField
// Use FlatField(FunctionType type, Set domain_set)
vals_ff = new FlatField( func_domain_range, domain_set);
// ...and put the values above into it
// Note the argument false, meaning that the array won't be copied
vals_ff.setSamples( zi , false );
// Create Display and its maps
// A 2D display
display = new DisplayImplJ3D(sz);
// Get display's graphics mode control and draw scales
GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
dispGMC.setScaleEnable(true);
// Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
// altitude to ZAxis and temperature to RGB
// Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
latMap = new ScalarMap( x1,Display.YAxis );
lonMap = new ScalarMap( x2,Display.XAxis );
// Add maps to display
display.addMap( latMap );
display.addMap( lonMap );
 // altitude to z-axis and temperature to color
altZMap = new ScalarMap( z,  Display.ZAxis );
tempRGBMap = new ScalarMap( temperature,  Display.RGB );
//tempIsoMap = new ScalarMap( temperature,  Display.IsoContour );
display.addMap( tempRGBMap );
//display.addMap( tempIsoMap );
// Add maps to display
display.addMap( altZMap );

// Uncomment following lines to have different data depiction
// temperature to z-axis and altitude to color
```

```
    //altRGBMap = new ScalarMap( zi,  Display.RGB );
    //tempZMap = new ScalarMap( temperature,  Display.ZAxis );
    //display.addMap( altRGBMap );
    //display.addMap( tempZMap );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    // Add reference to display
    display.addReference( data_ref );
    // Create application window and add display to window
    JFrame jframe = new JFrame(sz);
    jframe.getContentPane().add(display.getComponent());
    // Set window size and make it visible
    jframe.setSize(400, 400);
    jframe.setVisible(true);
  }
public static void contour_Plot1(f_xj f, double x1min,double x1max,double x2min,double x2max,int NROWS,int
NCOLS,int NCONTOUR,String s1,String s2,String s3)
    throws RemoteException, VisADException
    {
                // Declare variables
  // The domain quantities longitude and latitude
  // and the dependent quantity temperature
  RealType longitude, latitude;
  RealType temperature;
  // Tuple to pack longitude and latitude together, as the domain
  RealTupleType domain_tuple;
  // The function (domain_tuple -> temperature )
  // Remeber, range is only "temperature"
  FunctionType func_domain_temp;
   // Our Data values for the domain are represented by the Set
  Set domain_set;
  // The Data class FlatField
  FlatField vals_ff;
  // The DataReference from data to display
  DataReferenceImpl data_ref;
  // The 2D display, and its the maps
  DisplayImpl display;
  ScalarMap latMap, lonMap;
  ScalarMap tempIsoMap, tempRGBMap;
  // These objects are for drawing isocontours
  RealType isoTemperature;
  FunctionType func_domain_isoTemp;
  FlatField iso_vals_ff;
  DataReferenceImpl iso_data_ref;
    // Create the quantities
    // Use RealType(String name);
    latitude = RealType.getRealType(s1);
    longitude = RealType.getRealType(s2);
    domain_tuple = new RealTupleType(latitude, longitude);
    temperature = RealType.getRealType("temperature", SI.kelvin, null);
    isoTemperature = RealType.getRealType("isoTemperature", SI.kelvin, null);
    // Create a FunctionType (domain_tuple -> temperature )
    // Use FunctionType(MathType domain, MathType range)
    func_domain_temp = new FunctionType( domain_tuple, temperature);
    // ... the same for isoTemperature
    func_domain_isoTemp = new FunctionType( domain_tuple, isoTemperature);
    // Create the domain Set
```

```
// Use LinearDSet(MathType type, double first1, double last1, int lengthX,
//                                                double first2, double last2, int lengthY)
domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                        x2min, x2max, NCOLS);
// Get the Set samples to facilitate the calculations
float[][] set_samples = domain_set.getSamples( true );
// The actual temperature values are stored in this array
// float[1][ number_of_samples ]
double x[]=new double[2];
float[][] z_j = new float[1][NCOLS * NROWS];
// We fill our 'flat' array with the generated values
// by looping over NCOLS and NROWS
for(int c = 0; c < NCOLS; c++)
  {x[0]=x1min+(x1max-x1min)/(double)NCOLS*(double)c;
  for(int r = 0; r < NROWS; r++)
    {x[1]=x2min+(x2max-x2min)/(double)NROWS*(double)r;
            z_j[0][ c * NROWS + r ] = (float)f.func(x);
    }
  }
// Create the FlatFields
// Use FlatField(FunctionType type, Set domain_set)
// For the colored image
vals_ff = new FlatField( func_domain_temp, domain_set);
// ...and put the values above into it
// Note the argument false, meaning that the array won't be copied
vals_ff.setSamples( z_j , false );
  // ...and for the isocontours
iso_vals_ff = new FlatField( func_domain_isoTemp, domain_set);
// Get the values from the temperature FlatField
// create flat_isoVals array for clarity's sake
// "false" argument means "don't copy"
float[][] flat_isoVals = vals_ff.getFloats(false);
// ...and put the values above into it
// Note the argument false, meaning that the array won't be copied again
iso_vals_ff.setSamples( flat_isoVals , false );
// Create Display and its maps
// A 2D display
display = new DisplayImplJ2D(s3);
// Get display's graphics mode control and draw scales
GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
dispGMC.setScaleEnable(true);
// Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
// temperature to RGB and
// isoTemperature to IsoContour
// Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
latMap = new ScalarMap( latitude,    Display.XAxis );
lonMap = new ScalarMap( longitude, Display.YAxis );
tempIsoMap = new ScalarMap( isoTemperature,  Display.IsoContour );
tempRGBMap = new ScalarMap( temperature,  Display.RGB );
// Add maps to display
display.addMap( latMap );
display.addMap( lonMap );
display.addMap( tempIsoMap );
display.addMap( tempRGBMap );
// The ContourControl
// Note that we get the control from the IsoContour map
ContourControl isoControl = (ContourControl) tempIsoMap.getControl();
// Define some parameters for contour lines
```

```
    float interval = (float)((x1max-x1min)/NCONTOUR);  // interval between lines
    float lowValue = (float)x1min;   // lowest value
    float highValue = (float)x1max;    // highest value
    float base = (float)x1min;        //  starting at this base value
    // ...and set the lines with the method
    isoControl.setContourInterval(interval, lowValue, highValue, base);
    isoControl.enableLabels(true);
    // Create data references and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    iso_data_ref = new DataReferenceImpl("iso_data_ref");
    data_ref.setData( vals_ff );
    iso_data_ref.setData( iso_vals_ff );
    // Add reference to display
    display.addReference( iso_data_ref );
    display.addReference( data_ref );
    // Create application window and add display to window
    JFrame jframe = new JFrame(s3);
    jframe.getContentPane().add(display.getComponent());
    // Set window size and make it visible
    jframe.setSize(400, 400);
    jframe.setVisible(true);
              }
  public static void contour_Plot(f_xj f, double x1min,double x1max,double x2min,double x2max,int NROWS,int
NCOLS,int NCONTOUR,String s1,String s2,String s3)
    throws RemoteException, VisADException {
  RealType longitude, latitude;
  RealType temperature;
  // Tuple to pack longitude and latitude together, as the domain
  RealTupleType domain_tuple;
  // The function (domain_tuple -> temperature )
  // Remeber, range is only "temperature"
  FunctionType func_domain_range;
   // Our Data values for the domain are represented by the Set
  Set domain_set;
  // The Data class FlatField
  FlatField vals_ff;
  // The DataReference from data to display
  DataReferenceImpl data_ref;
  // The 2D display, and its the maps
  DisplayImpl display;
  ScalarMap latMap, lonMap;
  ScalarMap tempIsoMap, tempRGBMap;
    // Create the quantities
    // Use RealType(String name);
    latitude =  RealType.getRealType(s1);
    longitude = RealType.getRealType(s2);
    domain_tuple = new RealTupleType(latitude, longitude);
    temperature = RealType.getRealType("temperature");
    // Create a FunctionType (domain_tuple -> range_tuple )
    // Use FunctionType(MathType domain, MathType range)
    func_domain_range = new FunctionType( domain_tuple, temperature);
    // Create the domain Set
    // Use LinearDSet(MathType type, double first1, double last1, int lengthX,
    //                                                double first2, double last2, int lengthY)

    //int NCOLS = 50;
    //int NROWS = NCOLS;
    domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
```

```
                                                    x2min, x2max, NCOLS);
    // Get the Set samples to facilitate the calculations
    float[][] set_samples = domain_set.getSamples( true );
    // The actual temperature values are stored in this array
    // float[1][ number_of_samples ]
    double x[]=new double[2];
    float[][] z_j = new float[1][NCOLS * NROWS];
    // We fill our 'flat' array with the generated values
    // by looping over NCOLS and NROWS
    for(int c = 0; c < NCOLS; c++)
      {x[0]=x1min+(x1max-x1min)/(double)NCOLS*(double)c;
      for(int r = 0; r < NROWS; r++)
        {x[1]=x2min+(x2max-x2min)/(double)NROWS*(double)r;
                z_j[0][ c * NROWS + r ] = (float)f.func(x);
        }
      }
                  // Create a FlatField
    // Use FlatField(FunctionType type, Set domain_set)
    vals_ff = new FlatField( func_domain_range, domain_set);
    // ...and put the values above into it
    // Note the argument false, meaning that the array won't be copied
    vals_ff.setSamples( z_j , false );
    // Create Display and its maps
    // A 2D display
    display = new DisplayImplJ2D(s3);
    // Get display's graphics mode control and draw scales
    GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
    dispGMC.setScaleEnable(true);
    // Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
    // temperature to IsoContour and eventualy to RGB
    // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
    latMap = new ScalarMap( latitude,    Display.XAxis );
    lonMap = new ScalarMap( longitude,   Display.YAxis );
    // This is new!
    tempIsoMap = new ScalarMap( temperature,  Display.IsoContour );
    // this ScalarMap will color the isolines
    // don't foget to add it to the display
    tempRGBMap = new ScalarMap( temperature,  Display.RGB );
    // Add maps to display
    display.addMap( latMap );
    display.addMap( lonMap );
    display.addMap( tempIsoMap );
    display.addMap( tempRGBMap );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    // Add reference to display
    display.addReference( data_ref );
    // Create application window and add display to window
    JFrame jframe = new JFrame(s3);
    jframe.getContentPane().add(display.getComponent());
    jframe.setSize(300, 300);
    jframe.setVisible(true);
  }
    public static void contour_Plot(double a[][], double x1min,double x1max,double x2min,double x2max,int NROWS,int
NCOLS,int NCONTOUR,String s1,String s2,String s3)
    throws RemoteException, VisADException {
  RealType longitude, latitude;
```

```
RealType temperature;
// Tuple to pack longitude and latitude together, as the domain
RealTupleType domain_tuple;
// The function (domain_tuple -> temperature )
// Remeber, range is only "temperature"
FunctionType func_domain_range;
 // Our Data values for the domain are represented by the Set
Set domain_set;
// The Data class FlatField
FlatField vals_ff;
// The DataReference from data to display
DataReferenceImpl data_ref;
// The 2D display, and its the maps
DisplayImpl display;
ScalarMap latMap, lonMap;
ScalarMap tempIsoMap, tempRGBMap;
  // Create the quantities
  // Use RealType(String name);
  latitude =  RealType.getRealType(s1);
  longitude = RealType.getRealType(s2);
  domain_tuple = new RealTupleType(latitude, longitude);
  temperature = RealType.getRealType("temperature");
  // Create a FunctionType (domain_tuple -> range_tuple )
  // Use FunctionType(MathType domain, MathType range)
  func_domain_range = new FunctionType( domain_tuple, temperature);
  // Create the domain Set
  // Use LinearDSet(MathType type, double first1, double last1, int lengthX,
  //                                          double first2, double last2, int lengthY)
  //int NCOLS = 50;
  //int NROWS = NCOLS;
  domain_set = new Linear2DSet(domain_tuple, x1min, x1max, NROWS,
                                                    x2min, x2max, NCOLS);
  // Get the Set samples to facilitate the calculations
  float[][] set_samples = domain_set.getSamples( true );
  // The actual temperature values are stored in this array
  // float[1][ number_of_samples ]
  double x[]=new double[2];
  float[][] z_j = new float[1][NCOLS * NROWS];
  // We fill our 'flat' array with the generated values
  // by looping over NCOLS and NROWS
  for(int c = 0; c < NCOLS; c++)
    {x[0]=x1min+(x1max-x1min)/(double)NCOLS*(double)c;
    for(int r = 0; r < NROWS; r++)
      {x[1]=x2min+(x2max-x2min)/(double)NROWS*(double)r;
              z_j[0][ c * NROWS + r ] = (float)a[r][c];//(float)f.func(x);
      }
    }
               // Create a FlatField
  // Use FlatField(FunctionType type, Set domain_set)
  vals_ff = new FlatField( func_domain_range, domain_set);
  // ...and put the values above into it
  // Note the argument false, meaning that the array won't be copied
  vals_ff.setSamples( z_j , false );
  // Create Display and its maps
  // A 2D display
  display = new DisplayImplJ2D(s3);
  // Get display's graphics mode control and draw scales
  GraphicsModeControl dispGMC = (GraphicsModeControl)  display.getGraphicsModeControl();
```

```
    dispGMC.setScaleEnable(true);
    // Create the ScalarMaps: latitude to YAxis, longitude to XAxis and
    // temperature to IsoContour and eventualy to RGB
    // Use ScalarMap(ScalarType scalar, DisplayRealType display_scalar)
    latMap = new ScalarMap( latitude,    Display.XAxis );
    lonMap = new ScalarMap( longitude,   Display.YAxis );
    // This is new!
    tempIsoMap = new ScalarMap( temperature,  Display.IsoContour );
    // this ScalarMap will color the isolines
    // don't foget to add it to the display
    tempRGBMap = new ScalarMap( temperature,  Display.RGB );
    // Add maps to display
    display.addMap( latMap );
    display.addMap( lonMap );
    display.addMap( tempIsoMap );
    display.addMap( tempRGBMap );
    // Create a data reference and set the FlatField as our data
    data_ref = new DataReferenceImpl("data_ref");
    data_ref.setData( vals_ff );
    // Add reference to display
    display.addReference( data_ref );
    // Create application window and add display to window
    JFrame jframe = new JFrame(s3);
    jframe.getContentPane().add(display.getComponent());
    jframe.setSize(300, 300);
    jframe.setVisible(true);
  }

  public static void contour(f_xj f, double x1min,double x1max,double x2min,double x2max,int NROWS,int NCOLS,int
NCONTOUR,String s1,String s2,String s3) throws RemoteException, VisADException
  {contour_Plot(f,x1min,x1max,x2min,x2max,NROWS,NCOLS,NCONTOUR,s1,s2,s3);}
  public static void contour(f_xj f, double x1min,double x1max,double x2min,double x2max,String s1,String s2,String s3)
throws RemoteException, VisADException
  {contour_Plot(f,x1min,x1max,x2min,x2max,200,200,1,s1,s2,s3);}
  public static void contour(f_xj f, double x1min,double x1max,double x2min,double x2max) throws RemoteException,
VisADException
  {contour_Plot(f,x1min,x1max,x2min,x2max,200,200,1,"x","y","z");}

  public static void contour(double a[][], double x1min,double x1max,double x2min,double x2max,int NROWS,int
NCOLS,int NCONTOUR,String s1,String s2,String s3) throws RemoteException, VisADException
  {contour_Plot(a,x1min,x1max,x2min,x2max,NROWS,NCOLS,NCONTOUR,s1,s2,s3);}
  public static void contour(double a[][], double x1min,double x1max,double x2min,double x2max,String s1,String
s2,String s3) throws RemoteException, VisADException
  {contour_Plot(a,x1min,x1max,x2min,x2max,200,200,1,s1,s2,s3);}
  public static void contour(double a[][], double x1min,double x1max,double x2min,double x2max) throws
RemoteException, VisADException
  {contour_Plot(a,x1min,x1max,x2min,x2max,200,200,1,"x","y","z");}

  public static void func(f_xj f,double x1min,double x1max,double x2min,double x2max,int NCOLS,int NROWS,int
NCONTOUR,String sx,String sy,String sz) throws RemoteException, VisADException
  { plot3D_CD1(grid2D(f,x1min,x1max,x2min,x2max,NCOLS,
NROWS),x1min,x1max,x2min,x2max,NCONTOUR,sx,sy,sz);}
  public static void func(f_xj f,double x1min,double x1max,double x2min,double x2max,String sx,String sy,String sz)
throws RemoteException, VisADException
  { plot3D_CD1(grid2D(f,x1min,x1max,x2min,x2max,200, 200),x1min,x1max,x2min,x2max,1,sx,sy,sz);}
  public static void func(f_xj f,double x1min,double x1max,double x2min,double x2max) throws RemoteException,
VisADException
  { plot3D_CD1(grid2D(f,x1min,x1max,x2min,x2max,200, 200),x1min,x1max,x2min,x2max,1,"x","y","z1");}
```

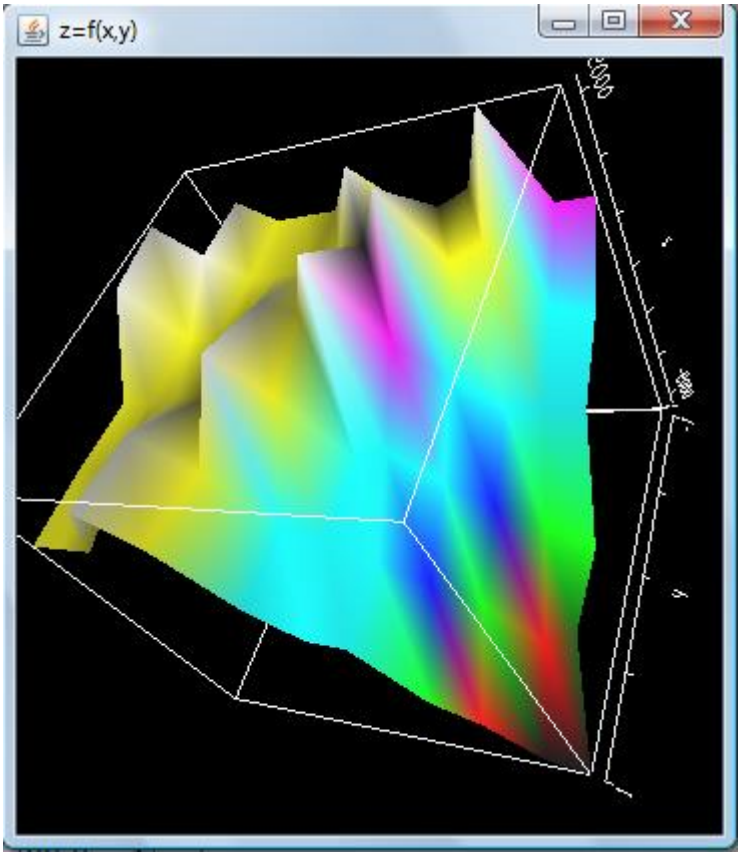
```
  public static void data(double zz[][],double x1min,double x1max,double x2min,double x2max,int NCONTOUR,String
sx,String sy,String sz) throws RemoteException, VisADException
  { plot3D_CD1(zz,x1min,x1max,x2min,x2max,NCONTOUR,sy,sx,sz);}
  public static void data(double zz[][],double x1min,double x1max,double x2min,double x2max,String sx,String sy,String
sz) throws RemoteException, VisADException
  { plot3D_CD1(zz,x1min,x1max,x2min,x2max,1,sy,sx,sz);}
  public static void data(double zz[][],double x1min,double x1max,double x2min,double x2max) throws RemoteException,
VisADException
  { plot3D_CD1(zz,x1min,x1max,x2min,x2max,1,"y","x","z");}

 public static void main(String[] args) throws RemoteException, VisADException
 { f1 f=new f1();
 double a[][]=grid2D(f,-1,1,-1,1,20,20);
 contour(f,-1.0,1.0,-1.0,1.0,100,100,100,"x","y","f(x,y)");
 }

} //end of Plot3D
```

Utilisation of the program in java environment can be shown with few examples

Program 21.2-2 Plot3DT1 3 dimensional graphic example program

```
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

 public class Plot3DT1
 {
 public static void main(String[] args) throws RemoteException, VisADException
 {
    double a[][]=new double[][]{
   {3000.0,  3000.0,  6500.0,  4000.0,  3500.0,  4000.0,  5500.0,  4000.0,  4000.0,  4000.0,  5000.0,  1000 },
   {1000.0,  1500.0,  4000.0,  3500.0,  6300.0,  4500.0,  4000.0,  3800.0,  3800.0,  3800.0,  5000.0,  6400 },
   {  0.0,     0.0,      0.0,  1500.0,  3000.0,  6500.0,  4500.0,  5000.0,  4000.0,  3800.0,  3800.0,  6200 },
   {-3000.0,  -2000.0,  -1000.0,   0.0,  1500.0,  1000.0,  4000.0,  5800.0,  4000.0,  4000.0,  3900.0,  3900 },
   {-3000.0,  -4000.0,  -2000.0,  -1000.0,  0.0,  1000.0,  1500.0,  4000.0,  5700.0,  4500.0,  4000.0,  4000.0 },
   {-6500.0,  -6000.0,  -4000.0,  -3000.0,  0.0,  100.0,  1500.0,  4500.0,  6000.0,  4000.0,  4000.0,  4000.0 }};
   Plot3D.plot3D_CD1(a,-1,2.0,-1.0,1.0,40,"x","y","z=f(x,y)");
 }
 }
```

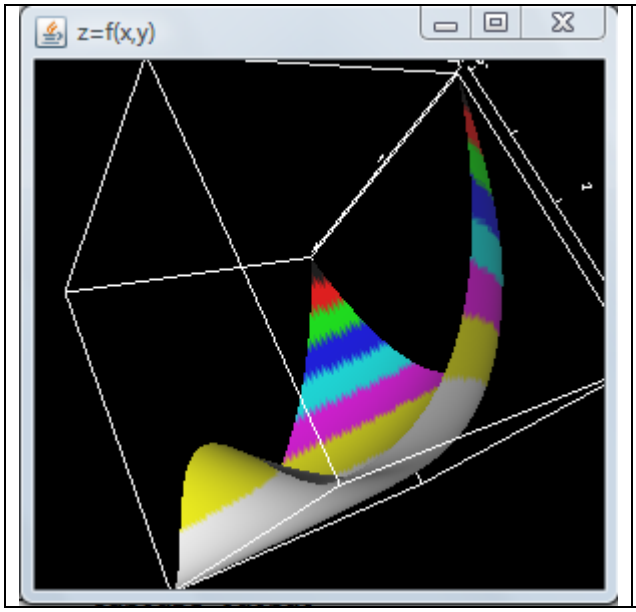
Output 21.2-1 Plot3D 3 scientific plotting example

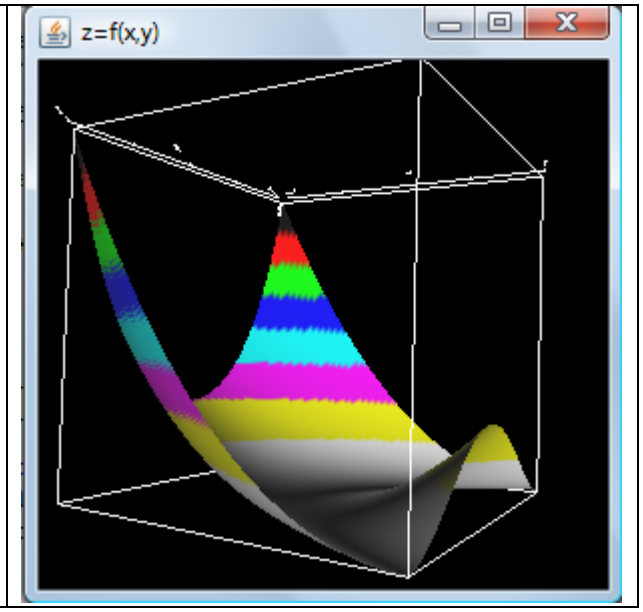


As it seen from the program, it is quite simple to plot the data. The next example is plotting a function

Program 21.2 -3 Plot3DT2 3 scientific plotting example

```
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

class f1 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=(x[1]-x[0]*x[0])*(x[1]-x[0]*x[0])+0.01*(x[0]-1.0)*(x[0]-1.0);
return -ff;
}
}

  public class Plot3DT2
  {
  public static void main(String[] args) throws RemoteException, VisADException
  { f1 f=new f1();
    Plot3D.plot3D_CD1(f,-1,1,-1,1.0,100,100,40,"x","y","z=f(x,y)");
  }
  }
```





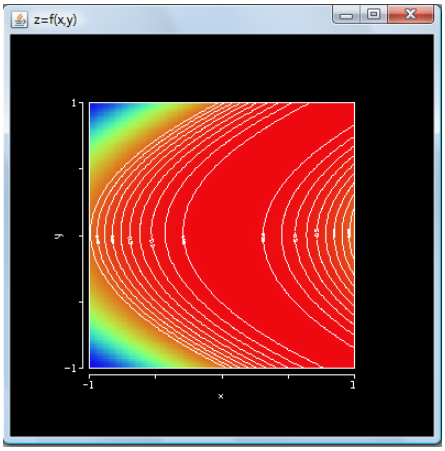
The next example draw a contour graphic

Program 21.2-4 Plot3D 3 dimensional graphic program CONTOUR graphic example

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
class f1 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=(x[1]-x[0]*x[0])*(x[1]-x[0]*x[0])+0.01*(x[0]-1.0)*(x[0]-1.0);
return -ff;
}}


 public class Plot3DMT3
 {  public static void main(String[] args)
 { f1 fa=new f1();
   Plot3D_JM pp=new Plot3D_JM(fa,-10,10,0.0,20);
   pp.plot();
 } }
```

Contour graphics are 2 dimensional mapping type of equal height graphics (isohips) of 3 dimensional shapes

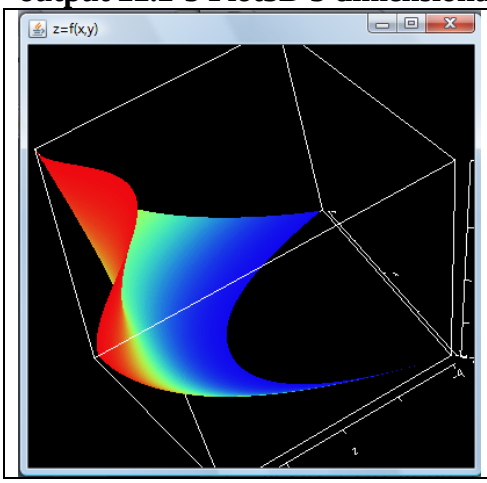
Output 21.2-4 Plot3D 3 dimensional graphic program CONTOUR graphic example

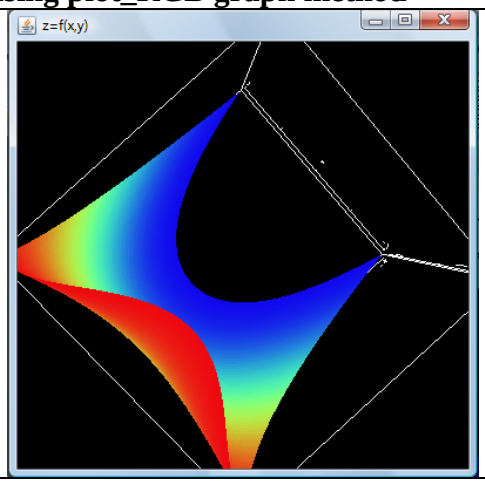


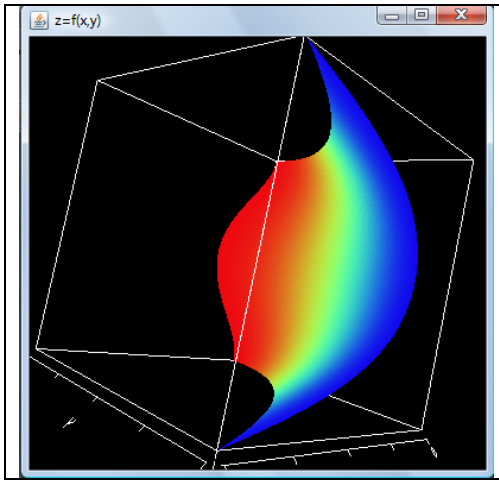
Program 22.2-5 Plot3D 3 dimensional plot by using plot_RGB graph method

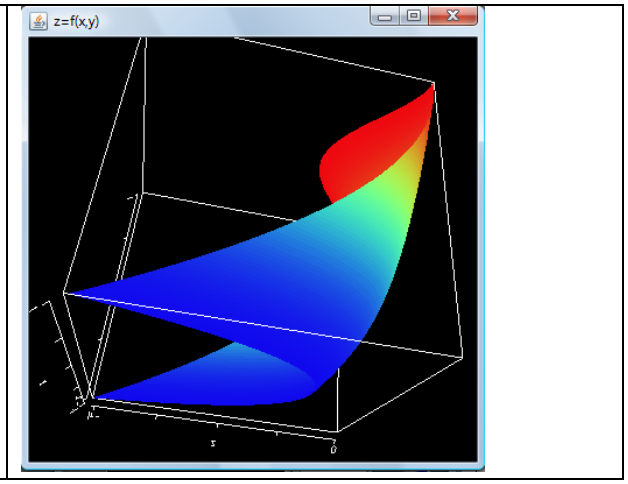
```
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

class f1 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=(x[1]-x[0]*x[0])*(x[1]-x[0]*x[0])+0.01*(x[0]-1.0)*(x[0]-1.0);
return -ff;
}
}

 public class Plot3DT5
 {
 public static void main(String[] args) throws RemoteException, VisADException
 { f1 f=new f1();
  Plot3D.plot3D_RGB(f,-1.0,1.0,-1.0,1.0,100,100,80,"x","y","z=f(x,y)");
 }
 }
```

output 22.2-5 Plot3D 3 dimensional plot by using plot_RGB graph method









JAVA 3 DIMENSIONAL GRAPHIC BY USING JMATH CLASS : PLOT3D_JM

When I search free internet graphic packages in Java, I found a packg with very simple usage : JMath. So in addition to VisAD, I also preperade an interphase to use JMath Package, as a standart graphic package in Numerical analysis applications. This package does not have the rofessional quality of VisAD, but this might be a plus for some of the users. You can obtain a copy of JMath package from http://jmathtools.sourceforge.net/doku.php. Three files you will obtain from this adres jmathio.jar, jmathplot.jar, ve jmatharray.jar should be copied into JAVAHOME/jre/lib/ext directory.

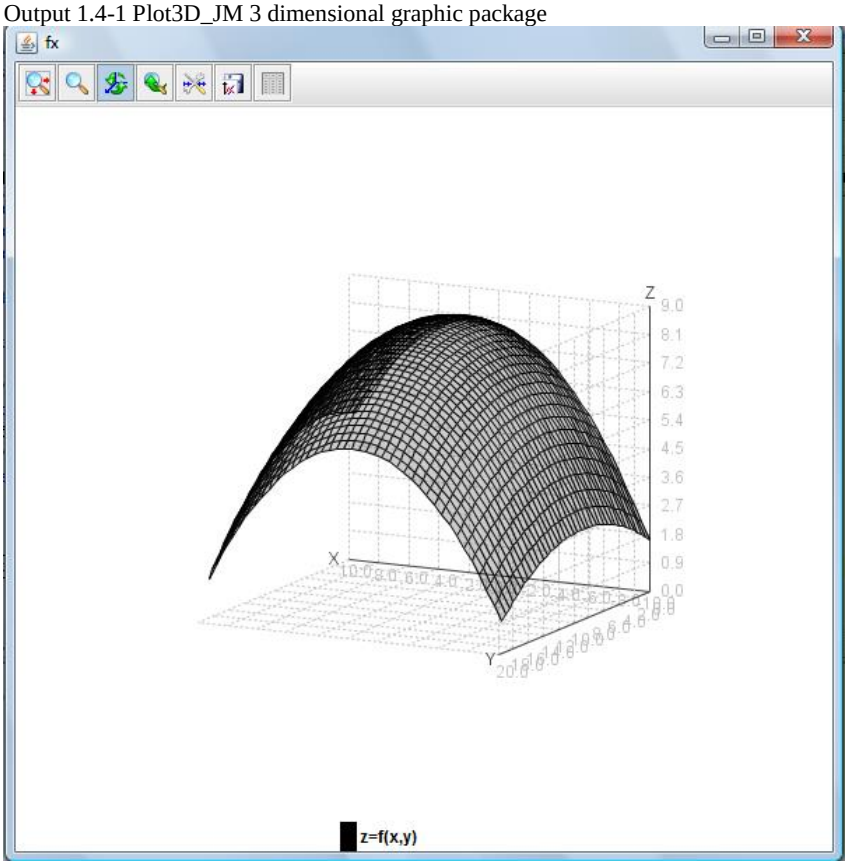
Program 22.2-6 Plot3D_JM 3 dimensional scientific graphic packge

```
//=====================================================
// Numerical Analysis package in java
// Plot3D_JM class to plot 3D data
// Dr. Turhan Coban
// EGE University, Engineering School, Mechanical Eng. Div.
// turhan.coban@ege.edu.tr
// ====================================================
// utilizes JMath  3D Plot
// http://jmathtools.sourceforge.net/doku.php
// to use this program please add to : javahome/jre/lib/ext
// jmathio.jar jmathplot.jar ve jmatharray.jar jar files

import javax.swing.*;
import org.math.plot.*;
import static java.lang.Math.*;
import static org.math.array.DoubleArray.*;
import java.awt.*;
import java.awt.event.*;

class f2 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=7.9+0.13*x[0]+0.21*x[1]-0.05*x[0]*x[0]-0.016*x[1]*x[1]-0.007*x[0]*x[1];
//ff=sin(x[0])*cos(x[1]);
return ff;
}
}

public class Plot3D_JM{

        Plot3DPanel plot;
```

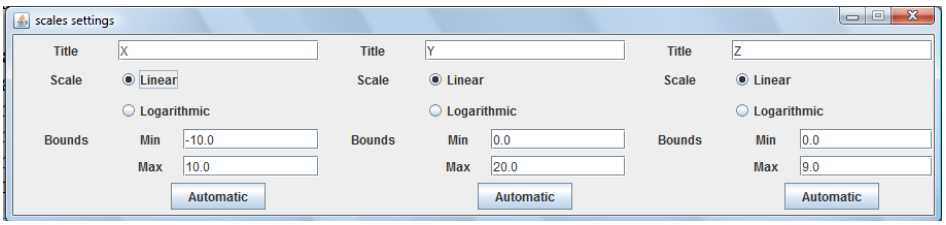
```
public Plot3D_JM(f_xj f, double x1min,double x1max,double x2min,double x2max,int NX,int NY,String sz,Color c)
{   plot = new Plot3DPanel("SOUTH");
    add(f,x1min,x1max,x2min,x2max,NX,NY,sz,c);
}
public Plot3D_JM(f_xj f, double x1min,double x1max,double x2min,double x2max,int NX,int NY)
{   plot = new Plot3DPanel("SOUTH");
    add(f,x1min,x1max,x2min,x2max,NX,NY,"z=f(x,y)",Color.blue);
}

public Plot3D_JM(f_xj f, double x1min,double x1max,double x2min,double x2max)
{   plot = new Plot3DPanel("SOUTH");
    add(f,x1min,x1max,x2min,x2max,20,20,"z=f(x,y)",Color.blue);
}

public Plot3D_JM(f_xj f, double x1min,double x1max,double x2min,double x2max,String sz)
{   plot = new Plot3DPanel("SOUTH");
    add(f,x1min,x1max,x2min,x2max,20,20,sz,Color.blue);
}

public void add(f_xj f, double x1min,double x1max,double x2min,double x2max,int NX,int NY,String sz,Color c)
{   double dx1=(x1max-x1min)/(double)NX;
double dx2=(x2max-x2min)/(double)NY; ;
double[] xx0 = increment(x1min, dx1, x1max);
double[] xx1 = increment(x2min, dx2, x2max);
double x[]=new double[2];
double[][] z = new double[NY][NX];
                    for (int i = 0; i < NX; i++)
                            for (int j = 0; j < NY; j++)
                                    {x[0]=xx0[i];x[1]=xx1[j];z[j][i] = f.func(x);}
                    // add grid plot to the PlotPanel
                    plot.addGridPlot(sz,c, xx0, xx1, z);
}

public void add(f_xj f, double x1min,double x1max,double x2min,double x2max,String sz,Color c)
{add(f,x1min,x1max,x2min,x2max,20,20,sz,c);}

public void add(f_xj f, double x1min,double x1max,double x2min,double x2max,String sz)
{add(f,x1min,x1max,x2min,x2max,20,20,sz,Color.blue);}

public void add(f_xj f, double x1min,double x1max,double x2min,double x2max)
{add(f,x1min,x1max,x2min,x2max,20,20,"z=f(x,y)",Color.blue);}
public void plot(String sz)
{JFrame frame = new JFrame(sz);
                    frame.setSize(600,600);
                    frame.setContentPane(plot);
                    frame.setVisible(true);
}
public void plot()
{plot("z=f(x,y)");}
public static void func(f_xj f, double x1min,double x1max,double x2min,double x2max,int NX,int NY,String sz)
{
double dx1=(x1max-x1min)/(double)NX;
double dx2=(x2max-x2min)/(double)NY; ;
double[] xx0 = increment(x1min, dx1, x1max);
double[] xx1 = increment(x2min, dx2, x2max);
double x[]=new double[2];
double[][] z = new double[NY][NX];
                    for (int i = 0; i < NX; i++)
                            for (int j = 0; j < NY; j++)
                                    {x[0]=xx0[i];x[1]=xx1[j];z[j][i] = f.func(x);}
    Plot3DPanel plot = new Plot3DPanel("SOUTH");
                    // add grid plot to the PlotPanel
                    plot.addGridPlot("z=f(x,y)",Color.black, xx0, xx1, z);
                    JFrame frame = new JFrame(sz);
                    frame.setSize(600,600);
                    frame.setContentPane(plot);
```

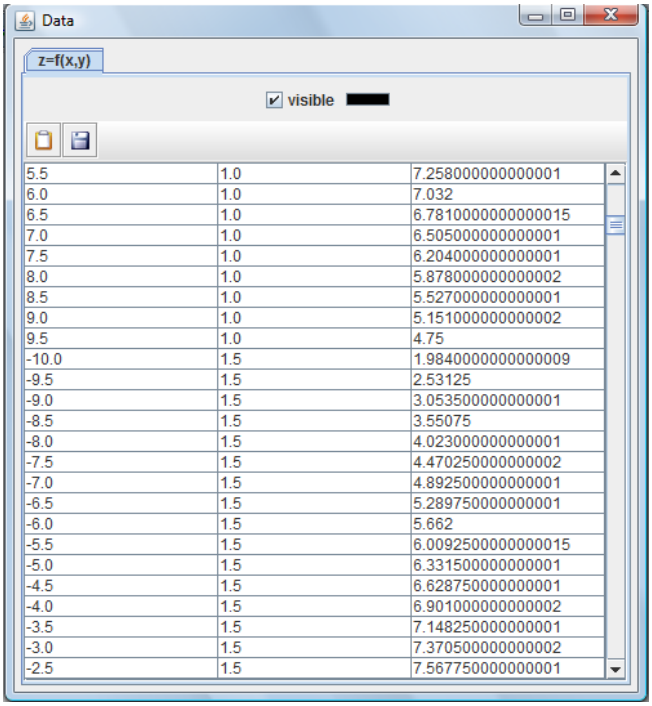
```
                        frame.setVisible(true);
  }
  public static void func(f_xj f, double x1min,double x1max,double x2min,double x2max)
  {func(f, x1min,x1max,x2min,x2max,40,40,"z=f(x,y)");}
  public static void func(f_xj f, double x1min,double x1max,double x2min,double x2max,String sz)
  {func(f, x1min,x1max,x2min,x2max,40,40,sz);}

 public static void main(String[] args)
 { f2 f=new f2();
  func(f,-10.0,10.0,0.0,20.0,"fx");
 }
} //end of Plot3D
```

Output 1.4-1 Plot3D_JM 3 dimensional graphic package



This program can be use as stand alone program, by chaging the input function or can also be called from your own program and plot te 3D output. The JMath package has some interesting facilities that you might enjoy using. For example you can change the maximum or minimum (scale of the plot) or you can plot the data in logarithmic scale.



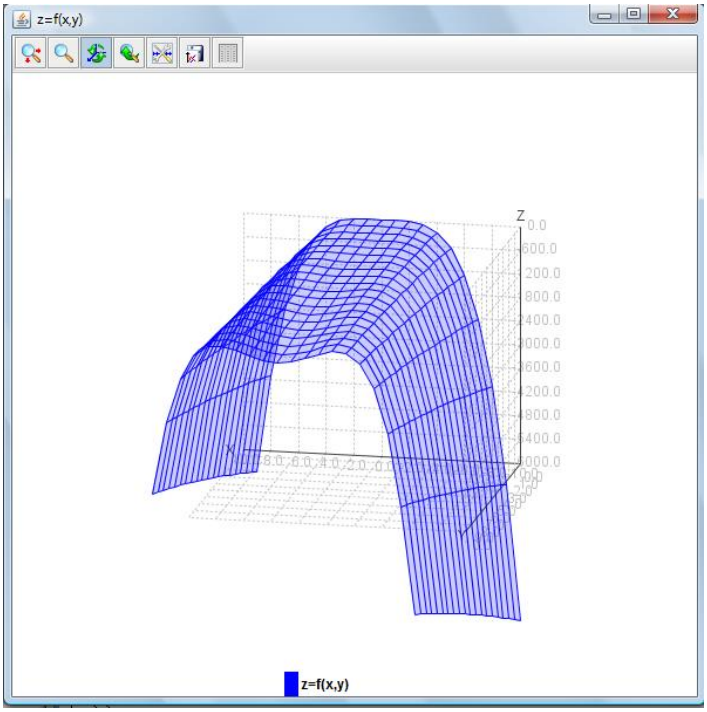


5.5	1.0	7.258000000000001
6.0	1.0	7.032
6.5	1.0	6.7810000000000015
7.0	1.0	6.505000000000001
7.5	1.0	6.204000000000001
8.0	1.0	5.878000000000002
8.5	1.0	5.527000000000001
9.0	1.0	5.151000000000002
9.5	1.0	4.75
-10.0	1.5	1.9840000000000009
-9.5	1.5	2.53125
-9.0	1.5	3.053500000000001
-8.5	1.5	3.55075
-8.0	1.5	4.023000000000001
-7.5	1.5	4.470250000000002
-7.0	1.5	4.892500000000001
-6.5	1.5	5.289750000000001
-6.0	1.5	5.662
-5.5	1.5	6.0092500000000015
-5.0	1.5	6.331500000000001
-4.5	1.5	6.628750000000001
-4.0	1.5	6.901000000000002
-3.5	1.5	7.148250000000001
-3.0	1.5	7.370500000000002
-2.5	1.5	7.567750000000001

The interphase program is very similar to VisAD interphase program to make a simple usage. An example code calling the program.

Program 22.2-7 Plot3D_JM 3 dimensional plot

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
class f1 extends f_xj
{
public double func(double x[])
{
//çözümü istenen fonksiyon
double ff;
ff=(x[1]-x[0]*x[0])*(x[1]-x[0]*x[0])+0.01*(x[0]-1.0)*(x[0]-1.0);
return -ff;
}}


 public class Plot3DMT3
 {  public static void main(String[] args)
 { f1 fa=new f1();
  Plot3D_JM pp=new Plot3D_JM(fa,-10,10,0.0,20);
  pp.plot();
 } }
```



22.3 REAL TIME GRAPHIC ENVIRONMENT

The next program utilises real time programming. It will plot the function you enter to the window. In order to achive that program writes the function, compile it by using java compiler than run it using our plot program. You can enter any function you desired to it. As an example function x*x-1 be entered to the plot window. All the code required will not be listed in here but they are existed in the class file folder.

Program 22.3-1 real time plot class ce1

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class ce1
{           String s;
            public  ce1()
            {input g=new input("f2D"," ");
             s=g.vericiktisi();
             try{ g.finalize();}
             catch(Throwable e1) {System.err.println("input error.");}
            }
```

```
            public  ce1(String function)
            {           functiongenerator g=new functiongenerator("f2D",function);
               s=g.createfunction();
               try{ g.finalize();}
               catch(Throwable e1) {System.err.println("input error.");}
            }

            public static void main(String arg[])
            { //String s1=JOptionPane.showInputDialog(" Fonksiyonu giriniz : ");
                     ce1 x0=new ce1();
      double a[]=Text.readDoubleV("Enter minimum and maximum values of the function xmin xmax");
      ce3 x1=new ce3(a[0],a[1],400,x0.s);
      //ce8 x2=new ce8(a[0],a[1],x0.s);
            }
}
```

Program 22.3-2 real time plot class ce3

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import static java.lang.Math.*;

class ce3 extends JApplet
{
            public ce3(double x1,double x2,int n,String s)
            {
            Plot pp=new Plot(new f2D(),x1,x2,n);
             pp.setPlabel("f(x)="+s);
             pp.plot();
            }

            public ce3(double x1,double x2,int n)
            {Plot pp=new Plot(new f2D(),x1,x2,n);
             pp.plot();
            }

            public ce3(double x1,double x2)
            {Plot pp=new Plot(new f2D(),x1,x2,50);
             pp.plot();
            }

            public static void main(String arg[])
            {double a=Double.parseDouble(JOptionPane.showInputDialog(" Fonksiyonun alt sınırını giriniz x_minimum  :
"));
      double b=Double.parseDouble(JOptionPane.showInputDialog(" Fonksiyonun üst sınırını giriniz x_maksimum : "));
             ce3 xx=new ce3(a,b,400);}
```

```
 }
```

Program 22.3-3 real time plot class input

```
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
import java.io.*;
import java.util.*;


public class input extends Thread implements ActionListener
{
            Container c;
            String s=" ";
            String s1="";
            functiongenerator fg;
            JTextArea jt;
            JButton bb;
            boolean basildi;

            public input(String isim,String fonksiyon)
            {
            jt=new JTextArea(s,5,30);
            jt.setFont(new Font("SansSerif",Font.PLAIN,20));
            fg=new functiongenerator();
            bb=new JButton("push button after entering the function into to the window : x");
            JScrollPane jsp=new JScrollPane(jt);
   JFrame cerceve=new JFrame("function input window example: Math.sin(x)-1/(x*x-1) ");
   cerceve.addWindowListener(new BasicWindowMonitor());
   c=cerceve.getContentPane();
   c.setLayout(new BorderLayout());
   c.add(jsp,BorderLayout.NORTH);
   c.add(bb,BorderLayout.SOUTH);
   bb.addActionListener(this);
   cerceve.pack();
   cerceve.setVisible(true);
   basildi=false;
   }

   public String vericiktisi()
   {
     while(!basildi)
     {
     try {Thread.sleep(100);}
     catch(InterruptedException e)  {System.err.println(e.toString());}
     }
               return s1;
            }

   protected void finalize() throws Throwable
   {super.finalize();}

public void actionPerformed( ActionEvent e)
{
  if(e.getSource()==bb)
  {s=jt.getText();fg.addfunction(s);s1=fg.createfunction();
  basildi=true;
```

```
  try{fg.finalize();finalize();
    } catch(Throwable e1) {System.err.println("girdi hatası.");}
  }
}
}
```

Program 22.3-4 real time plot class functiongenerator

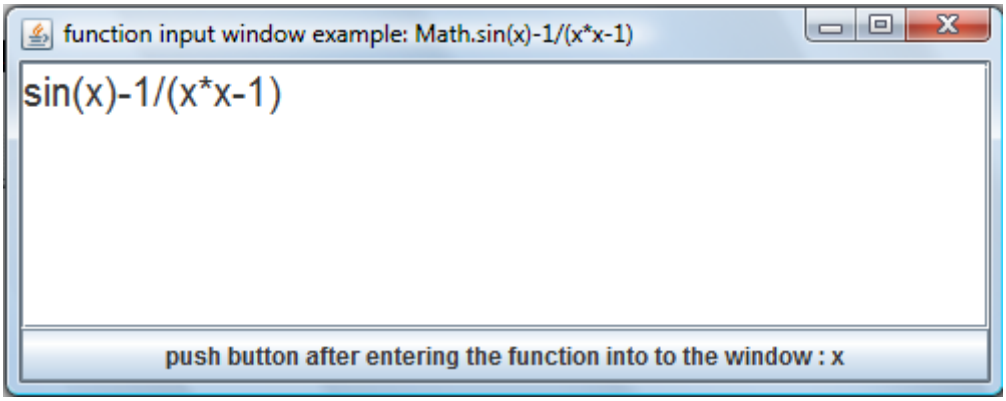
```
mport java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Math.*;
import java.lang.*;

class functiongenerator
 {
  String fonksiyon;
  String dosyaismi;
  String sinif;
  PrintStream fout;
  PrintStream ferr;

  public functiongenerator(String dosyaismi,String fonksiyon)
  { this.fonksiyon=fonksiyon;
    this.dosyaismi=dosyaismi;
    try {
    fout=new PrintStream(new FileOutputStream("print.txt"));
    ferr=new PrintStream(new FileOutputStream("error.txt"));
    }catch(FileNotFoundException e1) {System.err.println("dosya bulunamadı");}
  }

  public functiongenerator(String fonksiyon)
  {this.fonksiyon=fonksiyon;
   dosyaismi="f2D";
  }

  public void addfunction(String fonksiyon)
  {this.fonksiyon=fonksiyon;}

  public String toString()
  {return fonksiyon;}

  public functiongenerator()
  {fonksiyon="x*x-4.0";
   dosyaismi="f2D";
  }

 public  String createfunction()
 {String isim=dosyaismi+".java";
 String s="";
 try {PrintWriter ffout=new PrintWriter(new BufferedWriter(new FileWriter(isim)));
 s+="class "+dosyaismi+" extends f_x\n";
 s+=" {\n";
 s+=" public double func (double x)\n";
 s+="{ double y="+fonksiyon+";\n";
 s+=" return y;\n";
 s+="}\n";
 s+="}\n";
```

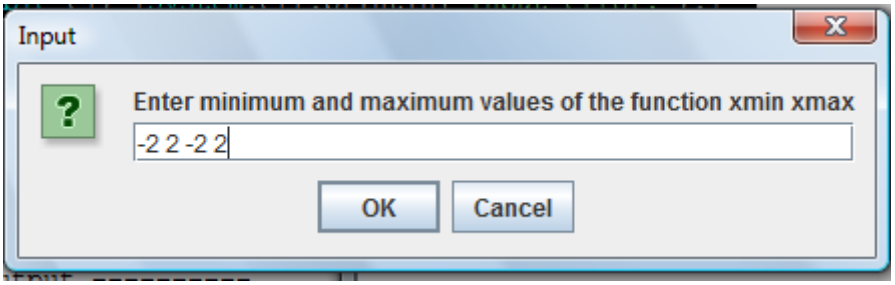
```
sinif=s;
ffout.println(s);
//System.out.println(s);
ffout.close();
}
catch(IOException e1) {System.err.println("girdi hatası.");}
catch(NumberFormatException e2){};

try{
JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
int rc = javac.run(null,null,null,isim);}
catch(NullPointerException e1) {System.err.println("NULL POINTER");}
return fonksiyon;
}

protected void finalize() throws Throwable
{super.finalize();}
}
```

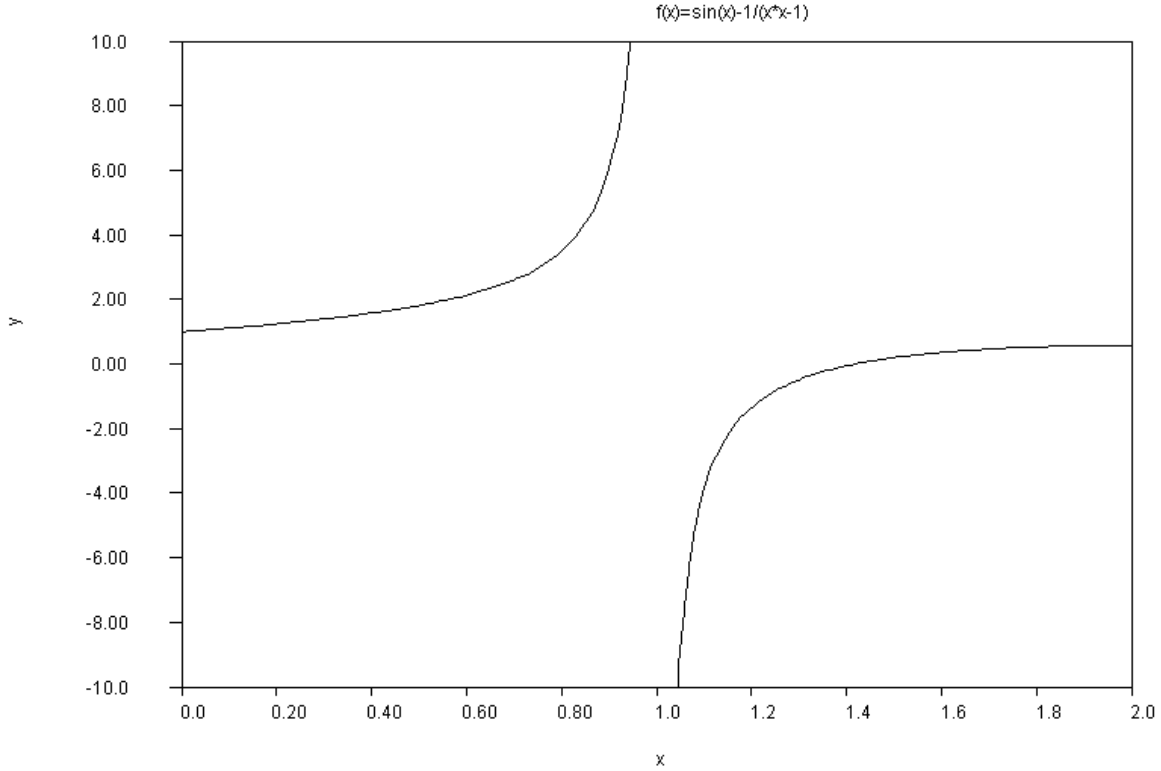
Real time program can be run directly from your screen by writing a small console program (a bat file for window dos system). That way you could have a direct print type function plot generator will be available to you.

Program 22.3-5 real time Plot class G2D.bat

```
java ce1
```







The same kind of real time program can be achived also for 3D plot environment

Program 22.3-6 real time 3D Plot class ce5

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;
import java.lang.Integer;
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.awt.image.*;
import javax.swing.*;
import java.util.Locale;
import java.text.*;
import java.util.*;
import javax.swing.table.*;
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;

class ce5 extends JApplet
{
            public  ce5()
            {
             input3 g=new input3("f3D"," ");
             String s=g.vericiktisi();
             try{g.finalize();} catch(Throwable e1)
             {System.err.
println("input hatas1[");}
            }

            public  ce5(String fonksiyon)
            {//Text.create3Dfunction(s1);
             functiongenerator3D g=new functiongenerator3D(fonksiyon);
```

```
            String s=g.createfunction();
             try{
    g.finalize();
      } catch(Throwable e1) {System.err.println("input hatas1[");}
    }

            public static void main(String arg[]) throws RemoteException,VisADException
            {
             ce5 xx=new ce5();
             double a[]=Text.readDoubleV("x_min x_max y_min y_max (enter 4 value)");
             ce6 xy=new ce6(a[0],a[1],a[2],a[3],200);
            }
 }
```

Program 22.3-6 real time 3D Plot class ce6

```
import visad.*;
import visad.java3d.DisplayImplJ3D;
import visad.java2d.DisplayImplJ2D;
import java.rmi.RemoteException;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

class ce6
{    public ce6(double xmin,double xmax,double ymin,double ymax,int n)throws RemoteException,VisADException
            {f3D f=new f3D();
     Plot3D.func(f,xmin,xmax,ymin,ymax);
     }
     public static void main(String arg[]) throws
s RemoteException,VisADException
            {
             double a[]=Text.readDoubleV("x_min x_max y_min y_max (4 değer giriniz)");
             ce6 xy=new ce6(a[0],a[1],a[2],a[3],200);
            }
}
```

Program 22.3-7 real time 3D Plot class functiongenerator3D

```
import java.io.*;
import javax.swing.*;
import javax.tools.*;
import java.util.*;

class functiongenerator3D extends Thread
 {
  String fonksiyon;
  String dosyaismi;
  String sinif;
  public functiongenerator3D(String dosyaismi,String fonksiyon)
  { this.fonksiyon=fonksiyon;
    this.dosyaismi=dosyaismi;
  }
  public functiongenerator3D(String fonksiyon)
  {this.fonksiyon=fonksiyon;
   dosyaismi="f3D";
  }
  public functiongenerator3D()
  {fonksiyon="x[0]*x[0]+x[1]*x[1]-4.0";
   dosyaismi="f3D";
  }
  public void addfunction(String fonksiyon)
  {this.fonksiyon=fonksiyon;
```

```
 }

 public String toString()
 {return fonksiyon;}

public  String createfunction()
{String isim=dosyaismi+".java";
String s="";
try {PrintWriter fout=new PrintWriter(new BufferedWriter(new FileWriter(isim)));
s+="class "+dosyaismi+" extends f_xj\n";
s+="{\n";
s+="public double func(double w[])\n";
s+="{//çözümü istenen fonksiyon\n";
s+="double x=w[0];\n";
s+="double y=w[1];\n";
s+="double ff;\n";
s+="ff="+fonksiyon+";\n";
s+="return ff;\n";
;
s+="}\n";
s+="}\n";
sinif=s;
fout.println(s);
//System.out.println(s);
fout.close();
}
catch(IOException e1) {System.err.println("girdi hatası.");}
catch(NumberFormatException e2){};
JavaCompiler javac = ToolProvider.getSystemJavaCompiler();
int rc = javac.run(null, null, null,isim);
return fonksiyon;
}

protected void finalize() throws Throwable
{super.finalize();}

}
```

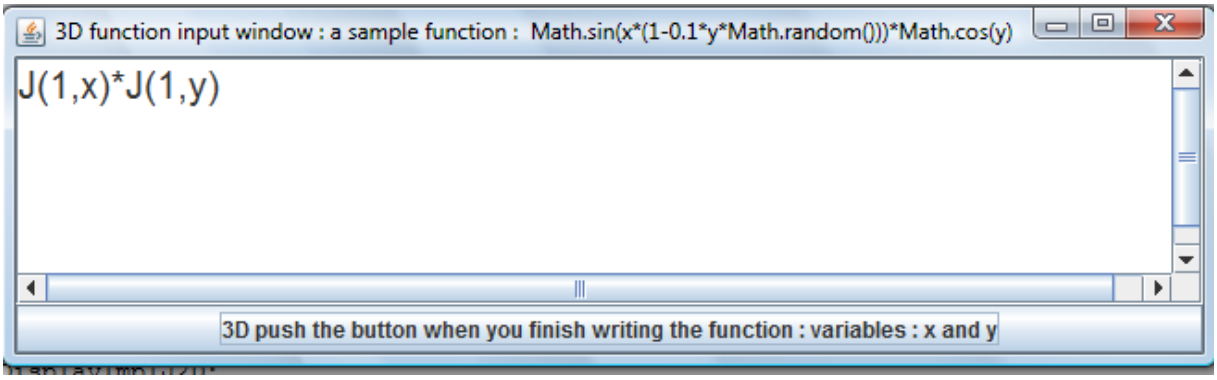
Program 22.3-8 real time 3D Plot class functiongenerator3D

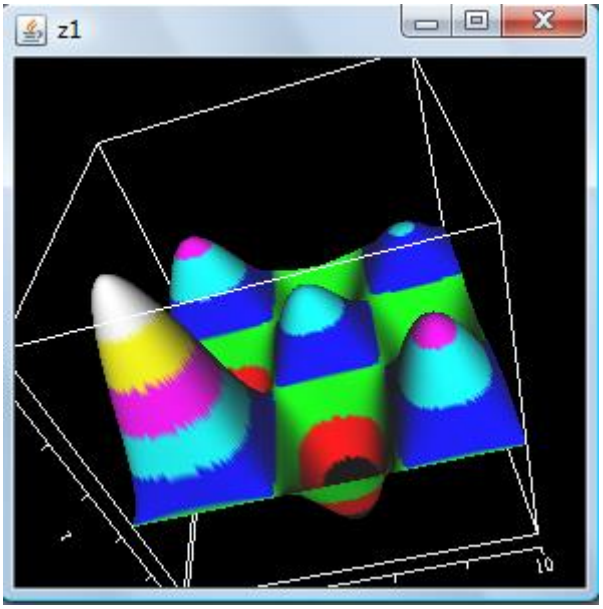
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;
import javax.swing.event.*;
import java.io.*;
import java.util.*;

public class input3 extends Thread implements ActionListener
{
        Container c;
        String s=" ";
        String s1="";
        functiongenerator3D fg;
        JTextArea jt;
        JButton bb;
        boolean basildi;

        public input3(String isim,String fonksiyon)
        {
        jt=new JTextArea(s,5,40);
        jt.setFont(new Font("SansSerif",Font.PLAIN,20));
        fg=new functiongenerator3D();
        bb=new JButton("3D push the button when you finish writing the function : variables : x and y");
        JScrollPane jsp=new JScrollPane(jt);
    JFrame cerceve=new JFrame("3D function input window : a sample function :  Math.sin(x*(1-
```

```
0.1*y*Math.random()))*Math.cos(y) ");
    cerceve.addWindowListener(new BasicWindowMonitor());
    c=cerceve.getContentPane();
    c.setLayout(new BorderLayout());
    c.add(jsp,BorderLayout.NORTH);
    c.add(bb,BorderLayout.SOUTH);
    bb.addActionListener(this);
    cerceve.pack();
    cerceve.setVisible(true);
    basildi=false;
    }

    public String vericiktisi()
    {
      while(!basildi)
      {
      try {Thread.sleep(100);}
      catch(InterruptedException e)  {System.err.println(e.toString());}
      }
              return s1;
            }

    protected void finalize() throws Throwable
    {super.finalize();}

public void actionPerformed( ActionEvent e)
{
 if(e.getSource()==bb)
 {s=jt.getText();fg.addfunction(s);s1=fg.createfunction();
 basildi=true;
 try{fg.finalize();finalize();
   } catch(Throwable e1) {System.err.println("input error");}
 }
}
}
```

Program 22.3-9 real time 3D Plot bat file G3D.bat

```
javaw ce5
```





22.4 READING DATA FROM THE GRAPHIC ENVIRONMENT

In numerical analysis one of the problem we face is the obtain data from some plot or graphic. In order to that scanning and digitizing of the data is needed. A simple digitizing program is given here.

Program 22.4-1 scanP.java

```
import java.io.*;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import java.awt.geom.*;
import java.net.URL;
import java.awt.image.*;
import java.util.*;

public class  scanP extends JPanel implements MouseListener,MouseMotionListener,MouseWheelListener
{
  int    x1[],y1[];
  public JLabel jp;
  public double a[][];
  int n;
  boolean mousemoved;
  Rectangle2D l;
  Ellipse2D l1,l2;
  Color c;
  Image img;
  public PrintWriter fout;
  String dosya;
  //BufferedWriter fout;
  double xmin,xmax,ymin,ymax;
  int    xcmin,xcmax,ycmin,ycmax;

 public scanP(String p1, String idosya,double ixmin,double ixmax,double iymin,double iymax)
```

```
{super();
    dosya=idosya;
    try {fout=new PrintWriter(new BufferedWriter(new FileWriter(dosya))); }
    catch(IOException e1) {System.err.println("input error.");}
    catch(NumberFormatException e2){}
 URL url = getClass().getResource(p1);
 img = getToolkit().getImage(url);
 xmin=ixmin;
 xmax=ixmax;
 ymin=iymin;
 ymax=iymax;
 jp=new JLabel();
 xcmin=(int)xmin;
 ycmin=(int)ymin;
 xcmax=(int)xmax;
 ycmax=(int)ymax;
 x1=new int[1000];
 y1=new int[1000];
 a=new double[2][1000];
 n=-2;
 addMouseListener(this);
 addMouseMotionListener(this);
 addMouseWheelListener(this);
 mousemoved=false;
}
public void setMinMax(double ixmin,double ixmax,double iymin,double iymax)
{  xmin=ixmin;
 xmax=ixmax;
 ymin=iymin;
 ymax=iymax;
}
public JLabel getLabel()
{  return jp;
}

public void setDosyaismi(String Di)
{  dosya=Di;
   try {fout=new PrintWriter(new BufferedWriter(new FileWriter(dosya))); }
    catch(IOException e1) {System.err.println("input error.");}
    catch(NumberFormatException e2){}}

public void setResimismi(String Ci)
{  URL url = getClass().getResource(Ci);
   img = getToolkit().getImage(url);}

public void setCizgi(int x1i,int y1i)
{if(n==-2) {xcmin=x1i;ycmin=y1i;}
else if(n==-1) {xcmax=x1i;ycmax=y1i;}
else
{x1[n]=x1i;y1[n]=y1i;
 a[0][n]=xmin+(xmax-xmin)/(xcmax-xcmin)*(x1i-xcmin);
 a[1][n]=ymin+(ymax-ymin)/(ycmax-ycmin)*(y1i-ycmin);
 System.out.println(a[0][n]+" "+a[1][n]);
 fout.println(a[0][n]+" "+a[1][n]);
}}

public void paintComponent(Graphics g)
{
//super.paintComponent(g);
Graphics2D g2=(Graphics2D)g;
g2.setFont(new Font("Serif",Font.BOLD,24));
g2.setColor(Color.blue);
g2.setStroke(new BasicStroke(2.0f));
Dimension d=getSize();
g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,RenderingHints.VALUE_ANTIALIAS_ON);
int dx = d.width;
int dy = d.height;
```

```
 g2.drawImage(img, 0, 0,dx,dy, this);
 if(n>=-1)
 {
 l1=new Ellipse2D.Double(xcmin-2,ycmin-2,5,5);
 l2=new Ellipse2D.Double(xcmax-2,ycmax-2,5,5);
 g2.draw(l1);
 g2.draw(l2);
 }
 for(int i=0;i<n;i++)
 {l=new Rectangle2D.Double(x1[i]-1,y1[i]-1,3,3);
 g2.draw(l);
 }
}

//MouseListener (fare dinleyicisi)
public void mouseClicked(MouseEvent e)
 {}

public void mouseclose()
{if(!mousemoved)
    {fout.close();
     Text.print(dosya,"digitiser output data : "+dosya+" file ");
     mousemoved=true;
    }
}

 public void mousePressed(MouseEvent e)
 {  if(e.isMetaDown()) {mouseclose();} //when right button is pushed
   else if(e.isAltDown())            //when middle button is pushed
   {}
   else { setCizgi(e.getX(),e.getY());n++;repaint();}  //when left button is pushed
 }

  public void mouseReleased(MouseEvent e)
 { }

 public void mouseEntered(MouseEvent e)
 { }

 public void mouseExited(MouseEvent e)
 { }
//MouseMotionListener methods

 public void mouseDragged(MouseEvent e)
 {   }

 public void mouseMoved(MouseEvent e)
 { double x1,y1;
  x1=xmin+(xmax-xmin)/(xcmax-xcmin)*(e.getX()-xcmin);
  y1=ymin+(ymax-ymin)/(ycmax-ycmin)*(e.getY()-ycmin);
  jp.setText("x = "+x1+"y = "+y1);
 }

 public void mouseWheelMoved(MouseWheelEvent e)
 { }

}
```

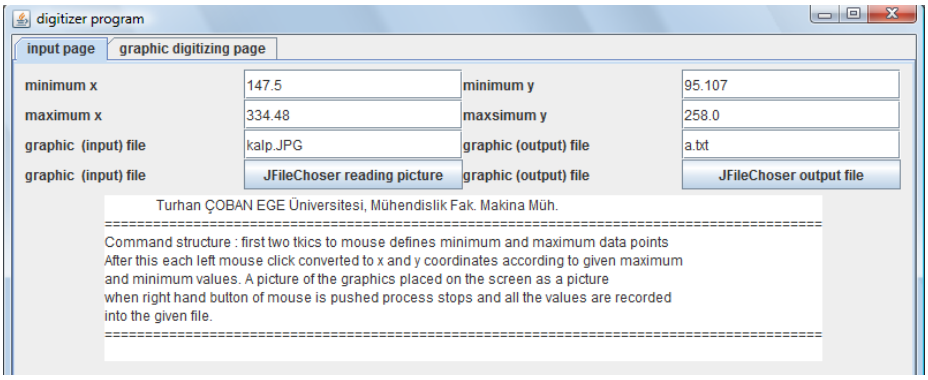
Program 22.4-2 digitizer.java

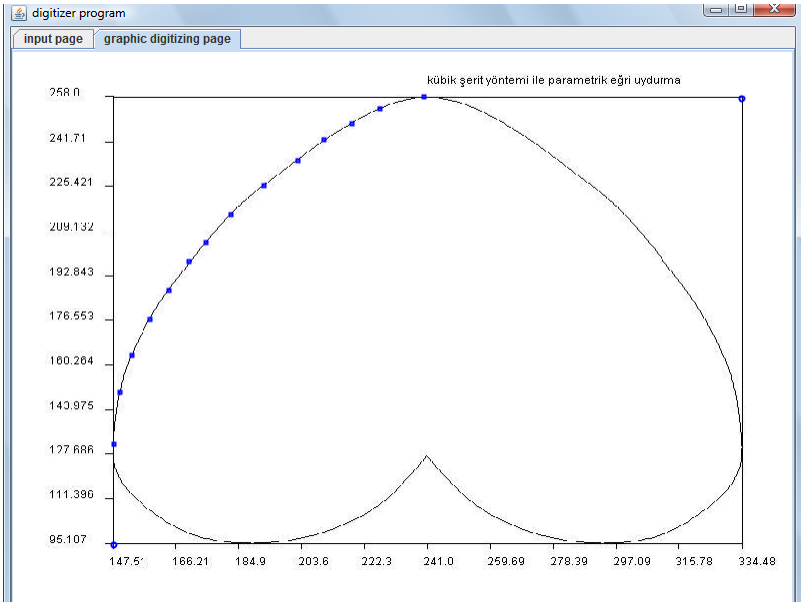
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
```

```
import javax.swing.filechooser.*;

 public class digitizer extends JApplet implements ActionListener
 {
 private JButton B[];
 private JLabel l[];
 JLabel altbar;
 private JTextField t[];
 private JTextArea ta;
 private String S[]={"input page","graphic digitizing page"};
 private Container c;
 private JPanel J[];
 scanP fp;
 double xmin,xmax,ymin,ymax;
 String resim,dosya;
 JTabbedPane jtp;

 public void init()
 {
            xmin=0.0;
            xmax=10.0;
            ymin=0.0;
            ymax=10.0;
            resim="kalp.JPG";
            dosya="a.txt";
 c=getContentPane();
 c.setLayout(new BorderLayout(5,5));
 B=new JButton[2];
 l=new JLabel[8];
 t=new JTextField[6];
 altbar=new JLabel();
 ta=new JTextArea(toString());
 jtp=new JTabbedPane();
 J=new JPanel[3];
 J[0]=new JPanel();
 J[0].setLayout(new GridLayout(4,4));
 J[1]=new JPanel();
 J[2]=new JPanel();
 B[0]=new JButton("JFileChoser reading picture");
 B[1]=new JButton("JFileChoser output file");
 l[0]=new JLabel("minimum x");
 l[1]=new JLabel("minimum y");
 l[2]=new JLabel("maximum x");
 l[3]=new JLabel("maxsimum y");
 l[4]=new JLabel("graphic  (input) file");
 l[5]=new JLabel("graphic (output) file");
 l[6]=new JLabel("graphic  (input) file");
 l[7]=new JLabel("graphic (output) file");
 t[0]=new JTextField(""+xmin);
 t[1]=new JTextField(""+ymin);
 t[2]=new JTextField(""+xmax);
 t[3]=new JTextField(""+ymax);
 t[4]=new JTextField(resim);
 t[5]=new JTextField(dosya);
 for(int i=0;i<5;i++) t[i].addActionListener(this);
 for(int i=0;i<2;i++) B[i].addActionListener(this);
 J[0].add(l[0]);
 J[0].add(t[0]);
 J[0].add(l[1]);
 J[0].add(t[1]);
 J[0].add(l[2]);
 J[0].add(t[2]);
 J[0].add(l[3]);
 J[0].add(t[3]);
 J[0].add(l[4]);
 J[0].add(t[4]);
 J[0].add(l[5]);
```

```
J[0].add(t[5]);
J[0].add(l[6]);
J[0].add(B[0]);
J[0].add(l[7]);
J[0].add(B[1]);
J[1].add(J[0],BorderLayout.NORTH);
J[1].add(ta,BorderLayout.CENTER);
jtp.addTab(S[0],J[1]);
fp=new scanP(resim,dosya,xmin,xmax,ymin,ymax);
//fp.setSize(1000,600);
//J[2].setLayout(new FlowLayout());
J[2].add(fp,BorderLayout.CENTER);
//J[2].add(fp.jp,BorderLayout.SOUTH);
jtp.addTab(S[1],fp);
c.add(jtp);
}

public String toString()
{String s;
s="              Turhan ÇOBAN EGE Üniversitesi, Mühendislik Fak. Makina Müh.\n";

s+="==========================================================================================\n";
s+="Command structure : first two tkics to mouse defines minimum and maximum data points\n";
s+="After this each left mouse click converted to x and y coordinates according to given maximum \n";
s+="and minimum values. A picture of the graphics placed on the screen as a picture\n";
s+="when right hand button of mouse is pushed process stops and all the values are recorded \n";
s+="into the given file.\n";

s+="==========================================================================================\n";
return s;
}

public void actionPerformed(ActionEvent e)
{
if(e.getSource()==t[0])
{           Double V0=new Double(t[0].getText());
   xmin=V0.doubleValue();
}
else if(e.getSource()==t[1])
{           Double V0=new Double(t[1].getText());
   ymin=V0.doubleValue();
}
else if(e.getSource()==t[2])
{           Double V0=new Double(t[2].getText());
   xmax=V0.doubleValue();
}
else if(e.getSource()==t[3])
{           Double V0=new Double(t[3].getText());
   ymax=V0.doubleValue();
}
else if(e.getSource()==t[4])
{           resim=t[4].getText();
  fp.setResimismi(resim);}
else if(e.getSource()==t[5])
{           dosya=t[4].getText();
   fp.setDosyaismi(dosya);
}
else if(e.getSource()==B[0])
{           File file=new File("kalp.jpg");
            JFileChooser fc=new JFileChooser();
   if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {file = fc.getSelectedFile();resim=file.getName();}
   t[4].setText(resim);
   fp.setResimismi(resim);
}
else if(e.getSource()==B[1])
{           File file=new File("a.txt");
            JFileChooser fc=new JFileChooser();
```

```
    if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {file = fc.getSelectedFile();dosya=file.getName();}
    t[5].setText(dosya);
    fp.setDosyaismi(dosya);
}
System.out.println("xmin = "+xmin+"xmax="+xmax+"ymin="+ymin+"ymax="+ymax);
fp.setMinMax(xmin,xmax,ymin,ymax);
repaint();
}


   public static void main(String s[])
   {
      JFrame f = new JFrame("digitizer program");
      JApplet applet = new digitizer();
      applet.init();
      f.add(applet);
      f.pack();
      f.setSize(800,1000);
      f.setVisible(true);
   }
}
```





digitiser output data : a.txt file

0	1
147.20367670364502	57.013392857142854
150.16690966719494	93.29464285714285
156.68602218700477	123.81696428571428
165.57572107765452	150.88392857142856
174.1690966719493	173.34374999999997
183.65144215530904	194.07589285714283
195.8006973058637	213.65624999999997
207.94995245641837	231.50892857142856
222.173470681458	248.20982142857142
240.54551505546752	259.15178571428567

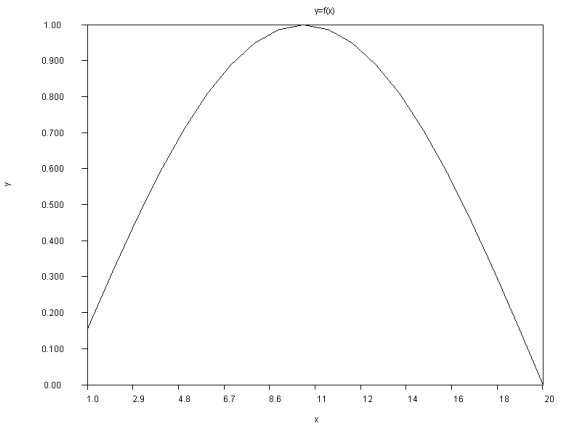
When the program starts, a JPG Picture of the required graphic input is selected .The first act is to sign minimum and maximum point in the picture. Number equivalents of this points should also be given to the program through the input windows. After this each click of the left Mouse button will be register a new point in cartesian coordinate system. Points will be recorded in the defined text output file. When the right button is clicked program will stop and brings out a list of all clicked points as file content and table output.

22.5 READING DATA FROM SPREADSHEET AND PLOTTING

In previous chapters, reading data from a spreadsheet is introduced. The data is then easily plotted by using Plot class. An interphase program plot_from_excel is created. This class uses class excel to read the data and plots it by using class Plot.

```
import javax.tools.*;
import java.util.*;
import java.lang.Math.*;
import java.lang.*;
import java.io.*;
import javax.swing.*;
import jxl.*;
import jxl.write.*;
import jxl.format.*;
import java.io.File;
import java.util.Hashtable;
import java.util.Enumeration;
import javax.swing.filechooser.*;
import java.util.*;
import java.awt.*;
import java.applet.Applet;
import java.awt.event.*;
import java.text.*;
import java.util.Locale;

public class plot_from_excel
{
public static void plot_from_excel(String filename, String pagetitle,int x0,int y0,int nx,int ny)
{
double a[][]=excel.read_double_from_excel(filename,pagetitle,x0,y0,nx,ny);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot_from_excel(String filename, String pagetitle,String xy0,String xyn)
{
double a[][]=excel.read_double_from_excel(filename,pagetitle,xy0,xyn);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot_from_excel(String filename, String pagetitle,String xy0,int nx,int ny)
```

```
{
double a[][]=excel.read_double_from_excel( filename, pagetitle,xy0,nx,ny);
Plot pp=new Plot(a);
pp.plot();
}

public static plot_from_excel(String filename, int pageno,int x0,int y0,int nx,int ny)
{
double a[][]=excel.read_double_from_excel(filename,pageno,x0,y0,nx,ny);
Plot pp=new Plot(a);
pp.plot();}

public static double[][] plot_from_excel(String filename, int pageno,int x0,int y0,int nx)
{
double a[][]=excel.read_double_from_excel(filename,pageno,x0,y0,nx);
Plot pp=new Plot(a);
pp.plot();
}
public static double[][] plot_from_excel(String filename, String pagetitle,int x0,int y0,int nx)
{
double a[][]=excel.read_double_from_excel(filename,pagetitle,x0,y0,nx);
Plot pp=new Plot(a);
pp.plot();
}

public static void plot_from_excel(String filename, int pageno,String xy0,String xyn)
{
double a[][]=excel.read_double_from_excel(filename,pageno,xy0,xyn);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot_from_excel(String filename, int pageno,String xy0,int nx,int ny)
{
double a[][]=excel.read_double_from_excel(filename,pageno,xy0,nx,ny);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot_from_excel(String filename, int pageno,String xy0,int nx)
{
double a[][]=excel.read_double_from_excel(filename,pageno,xy0,nx);
Plot pp=new Plot(a);
pp.plot();
}
public static void plot_from_excel(String filename, String pagetitle,String xy0,int nx)
{
double a[][]=excel.read_double_from_excel(filename,pagetitle,xy0,nx);
Plot pp=new Plot(a);
pp.plot();
}
public static void main(String args[]) throws IOException
{
plot_from_excel("data1.xls",0,"B2","C21");
}
}
```



EGE ÜNİVERSİTESİ, MÜHENDİSLİK FAKULTESİ, MAKİNA MÜHENDİSLİĞİ BÖLÜMÜ
BİLGİSAYAR PROGRAMLAMA II 2009-2010 Bahar dönemi VİZE İMTİHANI
COMPUTER PROGRAMMING II 2009-2010 Spring semester MIDTERM EXAM

Dr. M. Turhan ÇOBAN
Dr. Mehmet SARIKANAT

Öğrenci ismi	
Öğrenci numarası	
1	
2	
3	
4	
5	
6	
7	
8	
9	

PROBLEM 1 Explain the program and write the output

```
import java.sql.*;
import java.io.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class createMTPr1{
public static void main(String args[]) throws
IOException
{
try {
String
url="jdbc:mysql://localhost/temp?user=root";
Connection conn
=DriverManager.getConnection(url);
Statement stmt = null;
ResultSet rs1 = null;
stmt = conn.createStatement();
String s1= "create table MTpr1 (x float, y float,
y2 float );";
if (stmt.execute(s1)) {rs1 = stmt.getResultSet();}
} catch (SQLException ex) {
// handle any errors
System.out.println("SQLException: " +
ex.getMessage());
System.out.println("SQLState: " +
ex.getSQLState());
```

```
import java.io.*;
import java.sql.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class insertMTPr1
{
public static void main(String args[]) throws
IOException
{
  try {
  String
url="jdbc:mysql://localhost/temp?user=root";
  Connection conn
=DriverManager.getConnection(url);
  Statement stmt = null;
  ResultSet rs = null;
  stmt = conn.createStatement();
  String s2="insert into MTPr1 values(";
  String s3="";
  for(int i=0;i<6;i++)
  {
  s3=s2+i+","+i+", 0.0);";
  stmt.executeUpdate(s3);
  }
```

```
System.out.println("VendorError: " +
ex.getErrorCode());
 }
}
}
```

```
  System.out.println("data inserted into  database
MTPr1");
  System.exit(0);
  }
   catch (SQLException ex) {
      System.out.println("SQLException: " +
ex.getMessage());
      System.out.println("SQLState: " +
ex.getSQLState());
      System.out.println("VendorError: " +
ex.getErrorCode());
      }
}
}
```

```
import java.sql.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.io.*;

public class updateMTPr1{
public static void main(String args[]) throws IOException
{
try {
String url="jdbc:mysql://localhost/temp?user=root";
Connection conn =DriverManager.getConnection(url);
Statement stmt = null;
ResultSet rs1 = null;
stmt = conn.createStatement();
String query2="select * from MTPr1";
String query1="UPDATE MTPr1 SET y2=x*x+y*y";
if (stmt.execute(query1)) {rs1 = stmt.getResultSet();}
if (stmt.execute(query2)) {rs1 = stmt.getResultSet();}
while (rs1.next()) {
            float x = rs1.getFloat("x");
            float y = rs1.getFloat("y");
            float y2 = rs1.getFloat("y2");
            System.out.println("x= "+x+" y  = "+y+" y2 = "+y2);
        }
} catch (SQLException ex) {
// handle any errors
System.out.println("SQLException: " + ex.getMessage());
System.out.println("SQLState: " + ex.getSQLState());
System.out.println("VendorError: " + ex.getErrorCode());
 }}}
```

Program Explanation:

Program **createMTPr1**creates database table MTPr1with 3 data columns x,y and y2Program insertMTPr1inserts data 1 to 5 into x column and 1 to 5 into y column and 0 into y2 column

Program updateMTPr1replace y2 column values with x*x+y*y values

Program output :

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" updateMTPr1
x= 0.0 y  = 0.0 y2 = 0.0
x= 1.0 y  = 1.0 y2 = 2.0
x= 2.0 y  = 2.0 y2 = 8.0
x= 3.0 y  = 3.0 y2 = 18.0
x= 4.0 y  = 4.0 y2 = 32.0
x= 5.0 y  = 5.0 y2 = 50.0

> Terminated with exit code 0.
```

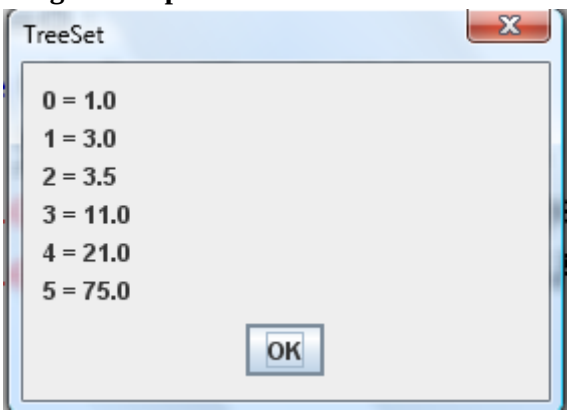
PROBLEM 2 Explain the program and give the output

```
import javax.swing.*;
import java.util.*;

class MTPr2 {

        public static void main(String args[])
        {
        TreeSet<Double> a=new TreeSet<Double>();
        Double s1=new Double(1.0);
        Double s2=new Double(3.0);
        Double s3=new Double(21.0);
        Double s4=new Double(3.5);
        Double s5=new Double(75.0);
        Double s6=new Double(11.0);
        a.add(s1);
        a.add(s2);
        a.add(s3);
        a.add(s4);
        a.add(s5);
        a.add(s6);
        Iterator i=a.iterator();
        String s="";
        int j=0;
        while(i.hasNext())
        {s+=(j++)+" = "+i.next()+"\n";}
        JOptionPane.showMessageDialog(null,s,"TreeSet",
        JOptionPane.PLAIN_MESSAGE);
        }
      }
```

Program explanation:

Program inputs data into TreeSet class and gives sorted output due to nature of TreeSet class (it is directly sorteed when data is enetered into the class structure)

Program output :



PROBLEM 3 Explain the program and give the output

```
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;

public class MTPr3
{
public static double[] readdata(String filename) throws IOException {
    BufferedReader fin=new BufferedReader(new FileReader(filename));
    String s1=fin.readLine();
    StringTokenizer token=new StringTokenizer(s1);
    int n=token.countTokens()-1;
    int m=n+1;
    double a[]=new double[m];
    int j=0;
    while(token.hasMoreTokens())
    {
    Double ax=new Double(token.nextToken());
    a[j++]=ax.doubleValue();
    }
    return a;
    }
```

```
public static void main(String args[]) throws IOException
{
double number[]=readdata("b.txt");
String s="";
for(int i=0;i<number.length;i++)
{s+="number["+i+"] = "+number[i]+"\n";}
JOptionPane.showMessageDialog(null,s,"Reading the data",JOptionPane.PLAIN_MESSAGE);
System.exit(0);
}
}
```

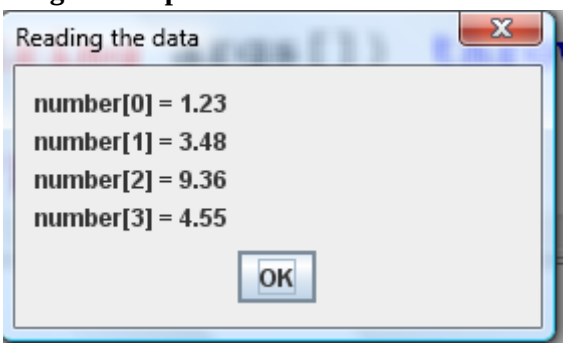
Data file b.txt

```
1.23 3.48 9.36 4.55
5.33 6.66 7.77
```

Program explanation:

Program reads the first line of data from the datafile b.txt and by using String Tokenizer class in readData method seperates it to tokens and convert it into double array output will be double array of first line as shown in the output below

Program output :



PROBLEM 4 write the program output

```
class box
{
private double length;
private double width;
private double height;

box(double l,double w, double h) throws zero_or_negative_sizeException
{ if(l<=0 || w<=0 || h<=0) throw new zero_or_negative_sizeException();
  else
  {
  length=l;
  width=w;
  height=h;
```

```
  }
}

box(box bi) throws zero_or_negative_sizeException
{if(bi.length<=0 || bi.width<=0 || bi.height<=0) throw new zero_or_negative_sizeException();
  else
  {
  length=bi.length;
  width=bi.width;
  height=bi.height;
  }
}
// This methods are written to access private length,
// width and height variables
public double read_length() {return length;}
public double read_width() {return width;}
public double read_height() {return height;}
//This methods are written to change private length,
// width and height variables
public void write_length(double l) throws zero_or_negative_sizeException
{if(l<=0 ) throw new zero_or_negative_sizeException();
length=l;
}
public void write_width(double w) throws zero_or_negative_sizeException
{if(w<=0 ) throw new zero_or_negative_sizeException();
width=w;
}
public void write_height(double h) throws zero_or_negative_sizeException
{if(h<=0 ) throw new zero_or_negative_sizeException();
height=h;
}
double volume()
{return length*width*height;}

public String toString()
{ String s="";
  s+="length = "+length+"\n";
  s+="width = "+width+"\n";
  s+="height = "+height+"\n";
  s+="Volume = "+volume()+"\n";
  return s;
}
}
```

```
public class zero_or_negative_sizeException extends ArithmeticException
{     public zero_or_negative_sizeException()  { super("the given dimension is zero or negative "); } }
```

```
import javax.swing.JOptionPane;

class MTPr4
{
  public static void main(String args[])
  {
  double volume;
  try{
    box firstbox= new box(1.0,1.2,0.2);
    String s="First box : \n"+firstbox.toString()+"\n";
    firstbox.write_length(3.0);
    firstbox.write_width(2.0);
    firstbox.write_height(0.0);
    s+="value changed first box : \n"+firstbox.toString()+"\n";
    JOptionPane.showMessageDialog(null,s,
    "class example:boxtest4",JOptionPane.PLAIN_MESSAGE);
    }  catch(zero_or_negative_sizeException e1)
  {  System.out.println(e1.toString());}
    System.exit(0);
  }
}
```

Program output :

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" MTPr4
zero_or_negative_sizeException: the given dimension is zero or negative

> Terminated with exit code 0.
```

PROBLEM 5 write the program output

```
import javax.swing.JOptionPane;
class MTPr5
{   public static void main(String args[])
   {
   double volume;
   String s="";
   try{
    box firstbox= new box(1.0,2.0,3.0);
    firstbox.write_length(4.0);
    firstbox.write_width(5.0);
    firstbox.write_height(6.0);
    s+="first box : \n"+firstbox.toString()+"\n";
    System.out.println(s);
    }  catch(zero_or_negative_sizeException e1)
  {  System.out.println(e1.toString());}
    System.exit(0);}}
```

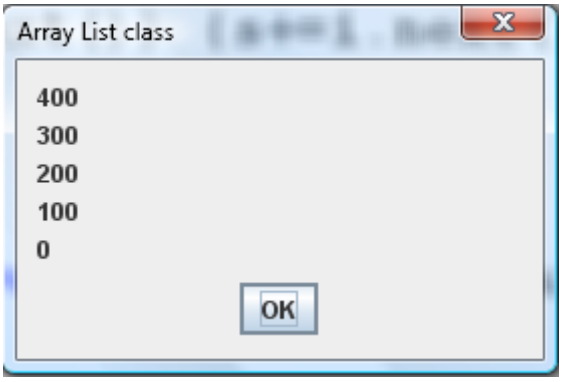
Program output :

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" MTPr5
first box :
length = 4.0
width = 5.0
height = 6.0
Volume = 120.0



> Terminated with exit code 0.
```

PROBLEM 6 Write the program output

```
import java.util.*;
import javax.swing.*;
public class MTPr6
{
  public static String print(Collection c)
  {
    String s="";
    Iterator i=c.iterator();
    while(i.hasNext()) {s+=i.next()+"\n";}
    return s;
  }
  public static void main(String args[])
  {
  ArrayList<Integer> a=new ArrayList<Integer>();
  for(int i=0;i<5;i++)
  {int sayi=i*100;
   a.add(new Integer(sayi));
  }
  String s1="Array List class";
  Collections.reverse(a);
  s+=print(a);
  JOptionPane.showMessageDialog(null,s,s1,
  JOptionPane.PLAIN_MESSAGE);
  }}
```

Program çıktısı:



PROBLEM 7 Write the program output

a.txt file

```
1 2 3 4
5 6
```

```
import java.util.Scanner; // program uses class Scanner
import java.util.Locale;  // international format library Locale
import javax.swing.*;     //Java swing graphic library
import java.io.*;         //Java input library for File class

public class MTPr7
{
    // main method begins execution of Java application
    public static void main( String args[] ) throws FileNotFoundException
    {
      Scanner input = new Scanner(new File("a.txt"));
      Locale us=new Locale("US");
      input.useLocale(us);
      double number1; // first number to add
      double number2; // second number to add
      double sum; // sum of number1 and number2
      number1 = input.nextDouble(); // read first number from user
      number2 = input.nextDouble(); // read second number from user
      sum = number1 + number2; // add numbers
      String s=String.format(us,"Sum is %10.5f\n", sum );
      System.out.println(s); // display sum
    } // end method main
} // end class MTPr7
```

Program output

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" MTPr7
Sum is    3.00000
> Terminated with exit code 0.
```

PROBLEM 8 explain what program is doing and give results

```
import java.util.*;
import java.util.concurrent.*;
public class MTPr8{
```

```
  public static void main(String[] args) {
    System.out.println("Navigable set\n");
    NavigableSet <Integer>nSet = new ConcurrentSkipListSet<Integer>();
    nSet.add(10);
    nSet.add(20);
    nSet.add(50);
    nSet.add(30);
    nSet.add(100);
    nSet.add(80);
    System.out.println( nSet.headSet(40));
  }}
```

Program explanation

Program ads data into NavigableSet cass and output values smaller than 40

Program output

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" MTPr8
Navigable set
[10, 20, 30]
> Terminated with exit code 0.
```

PROBLEM 9 MTPr9.java program is given belove as uncompleted. Enter a long String variable with words seperated by empty space such as : String s=”Once upon a time in a country far far away”; Then separete the String by using **public static String[] enterStringarray(String s)** method. Printout the seperated String array.

```
import java.io.*;
import java.util.*;
import javax.swing.*;

public class MTPr9
{
    public static String[] enterStringarray(String s)
    {
    StringTokenizer token=new StringTokenizer(s);
    int n=token.countTokens()-1;
    int m=n+1;
    String a[]=new String[m];
    int j=0;
    while(token.hasMoreTokens())
    {a[j++]=token.nextToken();}
    return a;
    }

  public static void main(String arg[])
  {
```

```
String s="Once upon a time in a country far far away";
String s1[]=enterStringarray(s);
String s2="";
for(int i=0;i<s1.length;i++)
{
s2+="string["+i+"]  "+s1[i]+"\n";
}
System.out.println(s2);
}
}
```

```
---------- Capture Output ----------
> "D:\java\bin\javaw.exe" MTPr9
string[0]  Once
string[1]  upon
string[2]  a
string[3]  time
string[4]  in
string[5]  a
string[6]  country
string[7]  far
string[8]  far
string[9]  away

> Terminated with exit code 0.
```

EGE ÜNİVERSİTESİ, MÜHENDİSLİK FAKULTESİ, MAKİNA MÜHENDİSLİĞİ BÖLÜMÜ
BİLGİSAYAR PROGRAMLAMA II 2010-2011 Güz dönemi VİZE İMTİHANI
COMPUTER PROGRAMMING II 2010-2011 Fall semester MIDTERM EXAM

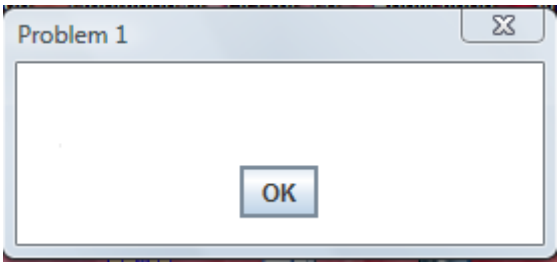
Dr. M. Turhan ÇOBAN
Dr. Mehmet SARIKANAT

Öğrenci ismi (Student name)	
Öğrenci numarası (student id number)	
1	
2	
3	
4	
5	
6	

7	
8	

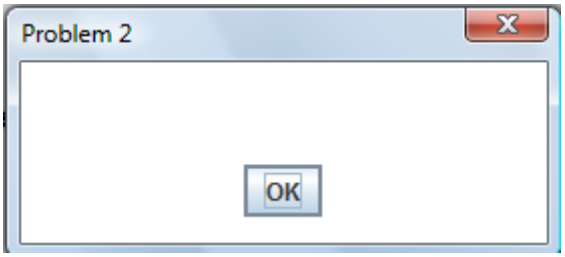
Problem 1 Give the output of the program

```
import javax.swing.*;
public class problem1
{
public static void main(String arg[])
{
double a=2.0;
double y;
if(a<0) {y=2*a;}
else if(a==0) {y=a*a;}
else {y=a*a-3*a;}
String s="y = "+y+" a= "+a;
JOptionPane.showMessageDialog(null,s,"Problem 1",JOptionPane.PLAIN_MESSAGE);
}
}
```



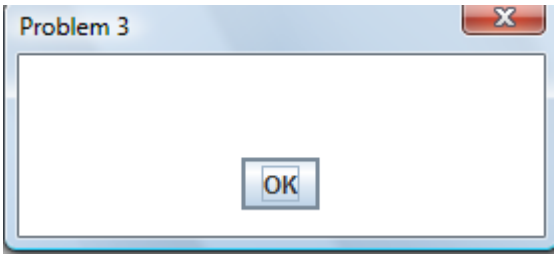
Problem 2 Give the output of the program

```
import javax.swing.*;
public class problem2
{
public static void main(String arg[])
{
double a=1.0;
double y=1;
while(a<4)
{
y+=a;
a++;
}
String s="y = "+y+" a= "+a;
JOptionPane.showMessageDialog(null,s,"Problem 2",JOptionPane.PLAIN_MESSAGE);
}
}
```



Problem 3 Give the output of the program

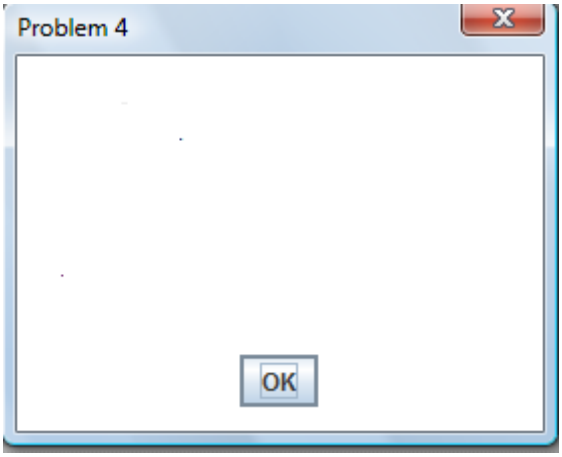
```
import javax.swing.*;
public class problem3
{

public static String wheather(int T)
{
String s="";
if(T<10) {s="today is chilly";}
else if(T<15) {s="today is cold";}
else if(T>=15 && T<=25) {s="today is nice";}
else if(T>=25 && T<=30) {s="today is warm";}
else if(T>=30 && T<=40) {s="today is hot";}
else  {s="it is like a hell today";}
return s;
}

public static void main(String arg[])
{
int T=27;
String s="T = "+T+" wheather  = "+wheather(T);
JOptionPane.showMessageDialog(null,s,"Problem 3",JOptionPane.PLAIN_MESSAGE);
}
}
```



Problem 4 Give the output of the program

```
import javax.swing.*;
public class problem4
{

public static String result(int number1,int number2)
{
```

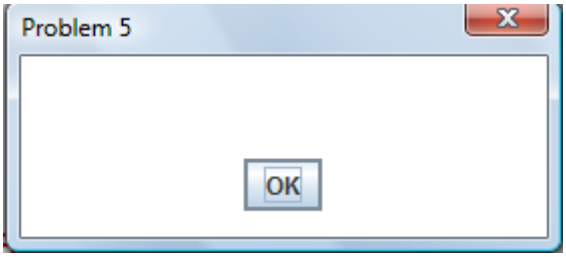
```
        int sum,substract,multiply,divide,remain;
        sum = number1+number2;
        substract = number1-number2;
        multiply = number1*number2;
        divide=number1/number2;
        remain=number1%number2;
        String s=number1+" +  "+number2+" = "+sum+"\n";
        s+=number1+" -  "+number2+" = "+substract+"\n";
        s+=number1+" x  "+number2+" = "+multiply+"\n";
        s+=number1+" /  "+number2+" = "+divide+"\n";
        s+=number1+" %  "+number2+" = "+remain+"\n";
        return s;
}


public static void main(String arg[])
{
int number1=2;
int number2=1;
String s="number1 = "+number1+"\nnumber2  = "+number2+"\n"+result(number1,number2);
JOptionPane.showMessageDialog(null,s,"Problem 4",JOptionPane.PLAIN_MESSAGE);
}
}
```



Problem 5 Give the output of the program

```
import javax.swing.*;
public class problem5
{
public static double f(double x)
{
        double y=0;
        double pow=1.0;
        for(int n=1;n<=4;n++)
        {pow*=x;
```

```
        y+=pow;
        }
        return y;
}

public static void main(String arg[])
{
double x=2.0;
String s="x = "+x+" y  = "+f(x);
JOptionPane.showMessageDialog(null,s,"Problem 5",JOptionPane.PLAIN_MESSAGE);
}
}
```

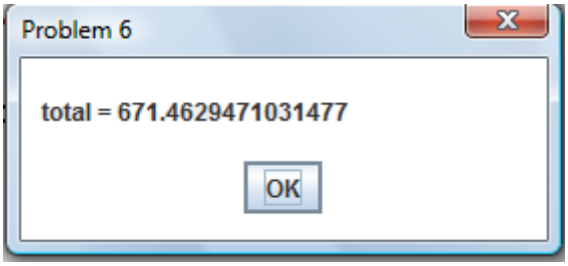


Problem 6 complete the program to calculate total of root squares of the numbers from 1 to 100

$$total=\sum_{n=1}^{100}\sqrt{n}$$

```
import javax.swing.*;
public class problem6
{

public static double total()
{
        double total=0;
```

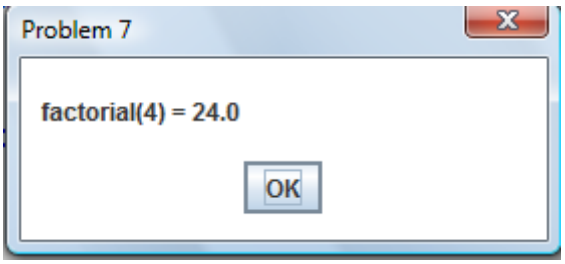
```
return total;
}


public static void main(String arg[])
{
String s="total = "+total();
JOptionPane.showMessageDialog(null,s,"Problem 6",JOptionPane.PLAIN_MESSAGE);
}
}
```



Problem 7 Write a program to calculate factorial of a given number

```
import javax.swing.*;
public class problem7
{
public static double factorial(int n)
{
        double fact=1;
```

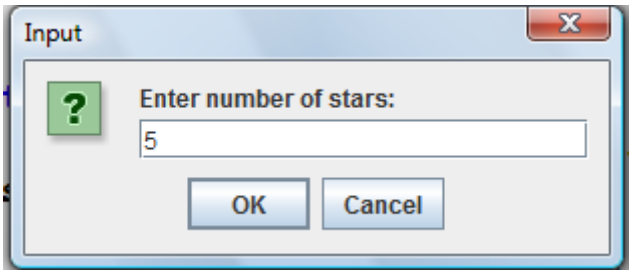
```
return fact;
}

public static void main(String arg[])
{
int n=4;
String s="factorial("+n+") = "+factorial(n);
JOptionPane.showMessageDialog(null,s,"Problem 7",JOptionPane.PLAIN_MESSAGE);
}
}
```



Problem 8 : Give the result of the following problem

```
import javax.swing.*;

public class problem8
{
    public static void main( String args[] )
    {
      int number1;
      number1=Integer.parseInt(JOptionPane.showInputDialog("Enter number of stars: ")); // read
number
      String star="*";
```

```
        String space="   ";
        String sum=star;
        String s="";
        for(int i=0;i<number1;i++)
        {  sum = space+sum;
          s+=sum+"\n";
        }
        JOptionPane.showMessageDialog(null,s,"star map",JOptionPane.PLAIN_MESSAGE);
    }
}
```

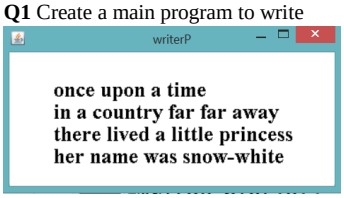


COMPUTER PROGRAMMING I final 2013 Ege University, School of Engineering, Dept. of Mechanical eng.

name	
Student id #	

Q1	
Q2	
Q3	
Q4	
Q5	
Q6	
Q7	
Q8	

Q1 Create a main program to write



By using writeP and FrameGraphic classes

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
public class writeP extends JPanel
{ //grafik penceresine yazı yazar
 String s[];
 public writeP(String si[])
 {super();
 s=new String[si.length];
 for(int i=0;i<si.length;i++){s[i]=si[i];}
 }
 public void paintComponent(Graphics g)
   {  super.paintComponent(g);
              Graphics2D g2 = (Graphics2D)g;
     g2.setFont(new Font("Serif",Font.BOLD,24));
```

```
        g2.setColor(Color.black);
        for(int i=0;i<s.length;i++)
                    g2.drawString(s[i],50,50+25*i);
    }
}
```

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
public class FrameGraphic extends JFrame
{
  JPanel d;
   public FrameGraphic(String a,JPanel di)
   {    super(a);
        d=di;
        add(d);
   }
   public static void plot(String a,JPanel di)
   {FrameGraphic f = new FrameGraphic(a,di);
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(1200,800);
        f.setVisible(true);
   }}
```

```
public class Q1
{
public static void main(String arg[])
{
String s[]={"once upon a time","in a country far far away","there lived a little princess","her name was snow-white"};
writeP pp=new writeP(s);
FrameGraphic.plot("Q1",pp);
}
}
```

Q2 write program output

```
import javax.swing.*;
class prime
{     public static int[] prime(int n1,int n2)
      {    // prime number = asal sayı
                    int i,j;
           int pp[]=new int[n2-n1];
           int nn=0;
           for(i=n1;i<=n2;i++)
           {for(j=2;j<i && (i%j!=0);j++) {}
           if(i==j) {pp[nn]=i;nn++;}
           }
           int p1[]=new int[nn];
           for(i=0;i<nn;i++)
           {p1[i]=pp[i];}
           return p1;
      }
      public static String toString(int x[])
      {String s="";
       for(int i=0;i<x.length;i++)
           {s+=" "+x[i];}
       return s;
      }

      public static void main(String args[])
      {
           int x[]=prime(2,15);
           System.out.println(toString(x));
      }
}
```

```
---------- Capture Output ----------
> "c:\java\bin\javaw.exe" prime
 2 3 5 7 11 13

> Terminated with exit code 0.
```

Q3 write program output

The serial solution of square root function

$$\sqrt{a} = e^{0.5*\ln(a)} = \sum_{k=1}^{\infty} \frac{[0.5\ln(a)]^k}{k!}$$

The following program is given, unfortunately method code of sqrt is missing write the code

```
import javax.swing.*;

public class Q3
{     public static double log(double x)
      {
       if(x<0) {System.out.println("number is less than 0");return 0;}
       double y=(x-1.0)/(x+1.0);
       double power=2.0*y;
       double z=y*y;
       int n=0;
       double ln=0;
       while(n<=300)
       {
       ln+=1./(2.0*n+1.0)*power;
       power*=z;
       n++;
       }
       return ln;
      }
      public static double exp(double x)
      { double exponent=1;
       double factorial=1;
       int number=1;
       double power=1;
       while(number<=300)
       {
       factorial*=number;
       power*=x;
       exponent+=power/factorial;
       number++;
       }
      return exponent;
      }
  public static double sqrt(double x)
      {
  return exp(0.5*log(x));
    }

     public static void main( String args[] )
     {
       String sa=JOptionPane.showInputDialog("x=");
       double x = Double.parseDouble(sa);
       String s="sqrt("+x+") = "+sqrt(x)+" Math library value = "+Math.sqrt(x);
       JOptionPane.showMessageDialog(null,s);
     } // end method main
} // end class
```

Q4 write program output

```
import javax.swing.*;
public class Q4
```

```
{
public static int countcharacters(char a,String s)
{
int n=s.length();
int number=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) number++;}
return number;
}

public static String[] word(char a,String s)
{
int nn=countcharacters(a,s);
int n=s.length();
String s1[]=new String[nn+1];
String s2="";
int j=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) {s1[j]=s2;s2="";j++;}
 else              {s2+=s.charAt(i);}
}
s1[nn]=s2;
return s1;
}

public static String[] word(String s)
{
char a=' ';
return word(a,s);
}

public static String toString(String s[])
{int n=s.length;
 String ss="";
 for(int i=0;i<n;i++)
 {ss+="i = "+i+" "+s[i]+"\n";}
  return ss;
}

public static void main(String arg[])
{
String s="Java is such an easy language";
String s1[]=word(s);
System.out.println(toString(s1));
}}
```

---------- Capture Output ----------
> "c:\java\bin\javaw.exe" Q4
i = 0 Java
i = 1 is
i = 2 such
i = 3 an
i = 4 easy
i = 5 language

> Terminated with exit code 0.

Q5 write program output

```
public class Q5
{
public static double[] toDouble(String s)
{String s1[]=Q4.word(s);
 int n=s1.length;
```

```
 double d[]=new double[n];
 for(int i=0;i<n;i++)
 {d[i]=Double.parseDouble(s1[i]);
 }
 return d;
}
}
```

```
public class f
{// General polynomial function
 // y=a[0]+a[1]*x+a[2]*x^2+...
 public double a[];
 public f(double ia[])
 {int n=ia.length;
  a=new double[n];
  for(int i=0;i<n;i++) {a[i]=ia[i];}
 }
 public double func(double x)
 {
  int n=a.length;
  double ff=a[n-1];
  for(int i=n-2;i>=0;i--)
  {ff=ff*x+a[i];}
  return ff;
 }
}
```

```
import javax.swing.*;
public class Q5A
{
public static void main(String arg[])
{
String s="1 1";
double a[]=Q5.toDouble(s);
f z=new f(a);
double x[]=new double[201];
double y[]=new double[201];
for(int i=0;i<5;i++)
{x[i]=i;y[i]=z.func(x[i]);
 System.out.println("i="+i+"x="+x[i]+"y="+y[i]);
}}}
```

---------- Capture Output ----------
> "c:\java\bin\javaw.exe" Q5A
i=0x=0.0y=1.0
i=1x=1.0y=2.0
i=2x=2.0y=3.0
i=3x=3.0y=4.0
i=4x=4.0y=5.0

> Terminated with exit code 0.

Q6 write a sample test program to add complex numbers z1=2+i; and z2=1+2i; by using class complex

```
public class complex
{ public double real;
  public double imaginary;
  public complex(double g,double s)
  {real=g;
   imaginary=s;
  }
  public complex(complex s1)
  {real=s1.real;
   imaginary=s1.imaginary;
  }
```

```
  public void add(complex s1)
  {real=real+s1.real;
   imaginary=imaginary+s1.imaginary;
  }
  public  static complex add(complex s1,complex s2)
  {complex z3=new complex((s1.real+s2.real),(s1.imaginary+s2.imaginary));
   return z3;
  }
  public void multiply(complex s1)
  {real=real*s1.real-imaginary*s1.imaginary;
   imaginary=real*s1.imaginary+imaginary*s1.real;
  }

  public  static complex4 multiply(complex4 s1,complex4 s2)
  { double r=s1.real*s2.real-s1.imaginary*s2.imaginary;
double imag=s1.real*s2.imaginary+s1.imaginary*s2.real;
complex4   s3=new complex4(r,imag);
return s3;
  }

  public String toString()
  { String s="";
    if(imaginary>0) {s+=real+" + i*"+imaginary+" \n";}
    else if(imaginary<0) {s+=real+" - i*"+(-imaginary)+" \n";}
    else {s+=real+" \n";}
    return s;
  }
}
```

```
import javax.swing.*;
public class Q6
{
public static void main(String si[])
{
complex r1=new complex(2.0,1.0);
complex r2=new complex(1.0,2.0);
r2.add(r1);
String s="r2="+r2;
JOptionPane.showMessageDialog(null,s,
     "complex numbers",
     JOptionPane.ERROR_MESSAGE);
}
}
```

Q7 write a program to calculate average of the numbers

```
 import javax.swing.*;
public class Q7
{
public static double average(int n)
{
double sum=0;
for(int i=1;i<=n;i++)
{sum+=i;}
sum/=n;
return sum;
}
public static void main(String arg[])
{
int n=Integer.parseInt(JOptionPane.showInputDialog("n="));
double avg=average(n);
JOptionPane.showMessageDialog(null,"average="+avg);
}
}
```

Q8 give program output

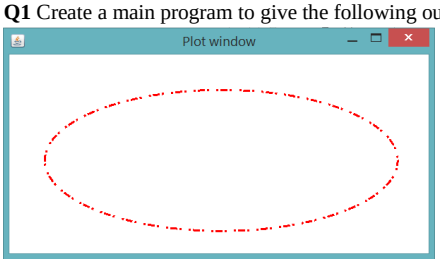
```
import javax.swing.*;
public class Q8
{
public static int countcharacters(char a,String s)
{
int n=s.length();
int number=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) number++;}
return number;
}

public static void main(String arg[])
{
String s="once upon a time in a country far far away";
int nn=countcharacters('a',s);
System.out.println(nn);
}
```

COMPUTER PROGRAMMING I make-up exam 2013 Ege University, School of Engineering, Dept. of Mechanical eng.

name	
Student id #	

Q1	
Q2	
Q3	
Q4	
Q5	
Q6	
Q7	
Q8	

Q1 Create a main program to give the following output



By using ellipseP1 and FrameGraphic classes

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class ellipseP1 extends JPanel
{
    public void paintComponent(Graphics g) {
            super.paintComponent(g);
    Graphics2D g2=(Graphics2D)g;
    g2.setFont(new Font("Serif",Font.BOLD,24));
    g2.setColor(Color.RED);
    this.setBackground(new Color(255,255,255));
    float dash3[] = {10.0f,3.0f,3.0f};
    BasicStroke d3 = new BasicStroke(3.0f,BasicStroke.CAP_BUTT,
                                     BasicStroke.JOIN_MITER,
                                     3.0f, dash3, 2.0f);
    g2.setStroke(d3);
    Ellipse2D x=new Ellipse2D.Double(50,50,500,200);
    g2.draw(x);
   }
}
```

```
import javax.swing.*;
import java.awt.Graphics;
import java.awt.*;
import java.awt.event.*;
public class FrameGraphic extends JFrame
{
  JPanel d;
   public FrameGraphic(String a,JPanel di)
   {    super(a);
        d=di;
        add(d);
   }
   public static void plot(String a,JPanel di)
   {FrameGraphic f = new FrameGraphic(a,di);
        f.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        f.setSize(1200,800);
        f.setVisible(true);
   }}
```

```
public class Q1
{
public static void main(String arg[])
{




}
}
```

Q2 write program output

```
 import javax.swing.*;
class even
{     public static int[] even(int n1,int n2)
      {
                        int i;
           int pp[]=new int[n2-n1];
           int nn=0;
           for(i=n1;i<=n2;i++)
           {  if(i%2==0) {pp[nn]=i;nn++;}}
           int p1[]=new int[nn];
           for(i=0;i<nn;i++)
           {p1[i]=pp[i];}
           return p1;
      }
      public static String toString(int x[])
      {String s="";
       for(int i=0;i<x.length;i++)
           {s+=" "+x[i];}
       return s;
      }

      public static void main(String args[])
      {
           int x[]=even(2,10);
           System.out.println(toString(x));
      }
}
```

Q3 A series Formula for ln(x) is given by

$$\ln(x) = 2y * \sum_{n=0}^{\infty} \frac{y^{2n}}{(2n+1)} = 2y * (1 + \frac{y^2}{3} + \frac{y^4}{5} + \frac{y^6}{7} + ...)$$

$y = \frac{x-1}{x+1}$ write program to calculate series by using a for loop.. Note that equation can be further simplify if a second conversion z=y*y is made

$$\ln(x) = 2y * \sum_{n=0}^{\infty} \frac{z^{n}}{(2n+1)} = 2y * (1 + \frac{z}{3} + \frac{z^2}{5} + \frac{z^3}{7} + ...)$$

The following program is given, unfortunately method code of sqrt is missing write the code

```
import javax.swing.*;

public class Q3
{     public static double ln(double x)
      {









        }
        return ln;
      }

     public static void main( String args[] )
     {
        String sa=JOptionPane.showInputDialog("x=");
        double x = Double.parseDouble(sa);
        String s="ln("+x+") = "+ln(x)+" Math library value = "+Math.log(x);
        JOptionPane.showMessageDialog(null,s);
     } // end method main
} // end class
```

Q4 write program output

```
 import javax.swing.*;
public class Q4A
{
public static int count_Turkish_vovels(String s)
{ char c[]={'a','e','i','ı','o','ö','u','ü'};
int m=c.length;
int n=s.length();
int number=0;
for(int i=0;i<n;i++)
{ for(int j=0;j<m;j++) {if(s.charAt(i)==c[j]) number++;}
}
return number;
}

public static void main(String arg[])
{
String s="ali topu at";
System.out.println("number of vovels = "+count_Turkish_vovels(s));
}}
```

Q5 write program output

```
 import javax.swing.JOptionPane;

// class rational numbers

public class rational {
    public int   numerator;
    public int   denominator;
public rational(int nnumerator,int ndenominator)
{numerator=nnumerator;
denominator=ndenominator;
}
public rational(rational n)
{numerator=n.numerator;
denominator=n.denominator;
}
public int leastcommondivisor()
{
// (least common divisor=en büyük ortak bölen)
int n=numerator;
int m=denominator;
if(n==0)  return m;
if(m==0)  return n;
while(m != n)
{  if(n>m) n=n-m;
   else    m=m-n;
}
return n;
}

public void simplify()
{
int isaret=1;
if(numerator<0)
{
 isaret=-isaret;
 numerator=-numerator;
}
if(denominator<0)
{
 isaret=-isaret;
 denominator=-denominator;
}
if(denominator==0)
   { JOptionPane.showMessageDialog(null,toString(),
     "dividing to zero error",
     JOptionPane.ERROR_MESSAGE);System.exit(0);
   }
int ebob=leastcommondivisor();
ebob=Math.abs(ebob);
numerator=isaret*numerator/ebob;
denominator=denominator/ebob;
}

public double toDouble()
{
//real equivalent of rational number
return ((double)numerator/(double)denominator);
}
```

```
        public static rational toRational(double d)
        { rational b=new rational((int)d*10000,10000);
          b.simplify();
          return b;
        }

        public void add(rational  sag)
        {
        numerator = numerator*sag.denominator + sag.numerator*denominator;
        denominator = denominator*sag.denominator;
        simplify();
        }

        public static rational  add(rational  r1,rational r2)
        {
        int n = r1.numerator*r2.denominator + r2.numerator*r1.denominator;
        int d = r1.denominator*r2.denominator;
        rational r=new rational(n,d);
        r.simplify();
        return r;
        }

        public void substract(rational  sag)
        {
        numerator = numerator*sag.denominator - sag.numerator*denominator;
        denominator = denominator*sag.denominator;
        simplify();
        }

        public void multiply(rational sag )
        {
        numerator = numerator*sag.numerator;
        denominator = denominator*sag.denominator;
        simplify();
        }

        public void divide(rational sag )
        {
        numerator = numerator*sag.denominator;
        denominator = denominator*sag.numerator;
        simplify();
        }

        public String toString()
        {
        String b="";
        if(Math.abs(denominator)!=1)
          {b=b+"( "+numerator+" / "+denominator+" )";}
        else
          {b=b+numerator+" ";}
        return b;
        }
};
```

```
import javax.swing.JOptionPane;

public class ratioanaltest1
{
public static void main(String si[])
{
rational r1=new rational(1,2);
rational r2=new rational(r1);
rational r3=new rational(0,1);
r3.add(r1);
```

```
r3.add(r2);
String s="r1="+r1+"r2="+r2+"r3="+r3;
JOptionPane.showMessageDialog(null,s,
    "rational numbers",
    JOptionPane.ERROR_MESSAGE);
}
```

Q6 write a method to calculate factorial

```
 import javax.swing.*;
public class Q6A
{
public static double factorial(int n)
{ double f=1;



  return f;
}

public static void main(String arg[])
{
System.out.println("factorial(4) = "+factorial(4));
}}
```

Q7 give program output

```
import javax.swing.*;
public class Q7A
{
public static boolean isGreekCharacter(char a)
{
String s="";
char x1='\u03B1';
char x2='\u03C9';
boolean b=false;
for(char x=x1;x<=x2;x++)
{if(a==x) b=true;break;}
return b;
}

public static void main(String arg[])
{
char c ='\u03B1';
System.out.println(isGreekCharacter(c));
}

}
```

Q8 give program output

```
public class f
{// General polynomial function
 // y=a[0]+a[1]*x+a[2]*x^2+...
 public double a[];
 public f(double ia[])
 {int n=ia.length;
  a=new double[n];
  for(int i=0;i<n;i++) {a[i]=ia[i];}
 }
 public double func(double x)
 {
  int n=a.length;
  double ff=a[n-1];
  for(int i=n-2;i>=0;i--)
  {ff=ff*x+a[i];}
  return ff;
 }
}
```

```
 import javax.swing.*;
public class Q8A
{
public static double[] toDouble(String s)
{String s1[]=word(s);
 int n=s1.length;
 double d[]=new double[n];
 for(int i=0;i<n;i++)
 {d[i]=Double.parseDouble(s1[i]);
 }
 return d;
}
public static int countcharacters(char a,String s)
{
int n=s.length();
int number=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) number++;}
return number;
}
public static String[] word(char a,String s)
{
int nn=countcharacters(a,s);
int n=s.length();
String s1[]=new String[nn+1];
String s2="";
int j=0;
for(int i=0;i<n;i++)
{if(s.charAt(i)==a) {s1[j]=s2;s2="";j++;}
 else            {s2+=s.charAt(i);}
}
s1[nn]=s2;
return s1;
}

public static String[] word(String s)
{
char a=' ';
return word(a,s);
}
```

```
public static String toString(String s[])
{int n=s.length;
 String ss="";
 for(int i=0;i<n;i++)
 {ss+="i = "+i+" "+s[i]+"\n";}
  return ss;
}

public static void main(String arg[])
{
String s="2 1";
double a[]=toDouble(s);
f z=new f(a);
double x[]=new double[201];
double y[]=new double[201];
for(int i=0;i<=5;i++)
{x[i]=i;y[i]=z.func(x[i]);
System.out.println("x["+i+"]="+x[i]+"y["+i+"]="+y[i]);
}}}
```

APPENDIX I JAVA LIBRARY PACKAGES

(http://download.oracle.com/javase/7/docs/api/index.html**)**

Package	Description
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.
java.awt.event	Provides interfaces and classes for dealing with different types of events fired by AWT components.
java.awt.font	Provides classes and interface relating to fonts.
java.awt.geom	Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.

java.awt.im	Provides classes and interfaces for the input method framework.
java.awt.im.spi	Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
java.awt.image	Provides classes for creating and modifying images.
java.awt.image.renderable	Provides classes and interfaces for producing rendering-independent images.
java.awt.print	Provides classes and interfaces for a general printing API.
java.beans	Contains classes related to developing *beans* -- components based on the JavaBeans™ architecture.
java.beans.beancontext	Provides classes and interfaces relating to bean context.
java.io	Provides for system input and output through data streams, serialization and the file system.
java.lang	Provides classes that are fundamental to the design of the Java programming language.
java.lang.annotation	Provides library support for the Java programming language annotation facility.
java.lang.instrument	Provides services that allow Java programming language agents to instrument programs running on the JVM.
java.lang.invoke	The java.lang.invoke package contains dynamic language support provided directly by the Java core class libraries and virtual machine.
java.lang.management	Provides the management interfaces for monitoring and management of the Java virtual machine and other components in the Java runtime.
java.lang.ref	Provides reference-object classes, which support a limited degree of interaction with the garbage collector.
java.lang.reflect	Provides classes and interfaces for obtaining reflective information about classes and objects.
java.math	Provides classes for performing arbitrary-precision integer arithmetic (BigInteger) and arbitrary-precision decimal arithmetic (BigDecimal).
java.net	Provides the classes for implementing networking applications.

java.nio	Defines buffers, which are containers for data, and provides an overview of the other NIO packages.
java.nio.channels	Defines channels, which represent connections to entities that are capable of performing I/O operations, such as files and sockets; defines selectors, for multiplexed, non-blocking I/O operations.
java.nio.channels.spi	Service-provider classes for the java.nio.channels package.
java.nio.charset	Defines charsets, decoders, and encoders, for translating between bytes and Unicode characters.
java.nio.charset.spi	Service-provider classes for the java.nio.charset package.
java.nio.file	Defines interfaces and classes for the Java virtual machine to access files, file attributes, and file systems.
java.nio.file.attribute	Interfaces and classes providing access to file and file system attributes.
java.nio.file.spi	Service-provider classes for the java.nio.file package.
java.rmi	Provides the RMI package.
java.rmi.activation	Provides support for RMI Object Activation.
java.rmi.dgc	Provides classes and interface for RMI distributed garbage-collection (DGC).
java.rmi.registry	Provides a class and two interfaces for the RMI registry.
java.rmi.server	Provides classes and interfaces for supporting the server side of RMI.
java.security	Provides the classes and interfaces for the security framework.
java.security.acl	The classes and interfaces in this package have been superseded by classes in the java.security package.
java.security.cert	Provides classes and interfaces for parsing and managing certificates, certificate revocation lists (CRLs), and certification paths.
java.security.interfaces	Provides interfaces for generating RSA (Rivest, Shamir and Adleman AsymmetricCipher algorithm) keys as defined in the RSA Laboratory Technical Note PKCS#1, and DSA (Digital Signature Algorithm) keys as defined in NIST's

	FIPS-186.
java.security.spec	Provides classes and interfaces for key specifications and algorithm parameter specifications.
java.sql	Provides the API for accessing and processing data stored in a data source (usually a relational database) using the Java™ programming language.
java.text	Provides classes and interfaces for handling text, dates, numbers, and messages in a manner independent of natural languages.
java.text.spi	Service provider classes for the classes in the java.text package.
java.util	Contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array).
java.util.concurrent	Utility classes commonly useful in concurrent programming.
java.util.concurrent.atomic	A small toolkit of classes that support lock-free thread-safe programming on single variables.
java.util.concurrent.locks	Interfaces and classes providing a framework for locking and waiting for conditions that is distinct from built-in synchronization and monitors.
java.util.jar	Provides classes for reading and writing the JAR (Java ARchive) file format, which is based on the standard ZIP file format with an optional manifest file.
java.util.logging	Provides the classes and interfaces of the Java™ 2 platform's core logging facilities.
java.util.prefs	This package allows applications to store and retrieve user and system preference and configuration data.
java.util.regex	Classes for matching character sequences against patterns specified by regular expressions.
java.util.spi	Service provider classes for the classes in the java.util package.
java.util.zip	Provides classes for reading and writing the standard ZIP and GZIP file formats.
javax.accessibility	Defines a contract between user-interface components and an assistive technology that

	provides access to those components.
javax.activation	
javax.activity	Contains Activity service related exceptions thrown by the ORB machinery during unmarshalling.
javax.annotation	
javax.annotation.processing	Facilities for declaring annotation processors and for allowing annotation processors to communicate with an annotation processing tool environment.
javax.crypto	Provides the classes and interfaces for cryptographic operations.
javax.crypto.interfaces	Provides interfaces for Diffie-Hellman keys as defined in RSA Laboratories' PKCS #3.
javax.crypto.spec	Provides classes and interfaces for key specifications and algorithm parameter specifications.
javax.imageio	The main package of the Java Image I/O API.
javax.imageio.event	A package of the Java Image I/O API dealing with synchronous notification of events during the reading and writing of images.
javax.imageio.metadata	A package of the Java Image I/O API dealing with reading and writing metadata.
javax.imageio.plugins.bmp	Package containing the public classes used by the built-in BMP plug-in.
javax.imageio.plugins.jpeg	Classes supporting the built-in JPEG plug-in.
javax.imageio.spi	A package of the Java Image I/O API containing the plug-in interfaces for readers, writers, transcoders, and streams, and a runtime registry.
javax.imageio.stream	A package of the Java Image I/O API dealing with low-level I/O from files and streams.
javax.jws	
javax.jws.soap	
javax.lang.model	Classes and hierarchies of packages used to model the Java programming language.
javax.lang.model.element	Interfaces used to model elements of the Java programming language.

javax.lang.model.type	Interfaces used to model Java programming language types.
javax.lang.model.util	Utilities to assist in the processing of program elements and types.
javax.management	Provides the core classes for the Java Management Extensions.
javax.management.loading	Provides the classes which implement advanced dynamic loading.
javax.management.modelmbean	Provides the definition of the ModelMBean classes.
javax.management.monitor	Provides the definition of the monitor classes.
javax.management.openmbean	Provides the open data types and Open MBean descriptor classes.
javax.management.relation	Provides the definition of the Relation Service.
javax.management.remote	Interfaces for remote access to JMX MBean servers.
javax.management.remote.rmi	The RMI connector is a connector for the JMX Remote API that uses RMI to transmit client requests to a remote MBean server.
javax.management.timer	Provides the definition of the Timer MBean.
javax.naming	Provides the classes and interfaces for accessing naming services.
javax.naming.directory	Extends the javax.naming package to provide functionality for accessing directory services.
javax.naming.event	Provides support for event notification when accessing naming and directory services.
javax.naming.ldap	Provides support for LDAPv3 extended operations and controls.
javax.naming.spi	
javax.net	Provides classes for networking applications.
javax.net.ssl	Provides classes for the secure socket package.
javax.print	Provides the principal classes and interfaces for the Java™ Print Service API.
javax.print.attribute	Provides classes and interfaces that describe the types of Java™ Print Service attributes and how they can be collected into attribute sets.

javax.print.attribute.standard	Package javax.print.attribute.standard contains classes for specific printing attributes.
javax.print.event	Package javax.print.event contains event classes and listener interfaces.
javax.rmi	Contains user APIs for RMI-IIOP.
javax.rmi.CORBA	Contains portability APIs for RMI-IIOP.
javax.rmi.ssl	Provides implementations of RMIClientSocketFactory and RMIServerSocketFactory over the Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocols.
javax.script	The scripting API consists of interfaces and classes that define Java TM Scripting Engines and provides a framework for their use in Java applications.
javax.security.auth	This package provides a framework for authentication and authorization.
javax.security.auth.callback	This package provides the classes necessary for services to interact with applications in order to retrieve information (authentication data including usernames or passwords, for example) or to display information (error and warning messages, for example).
javax.security.auth.kerberos	This package contains utility classes related to the Kerberos network authentication protocol.
javax.security.auth.login	This package provides a pluggable authentication framework.
javax.security.auth.spi	This package provides the interface to be used for implementing pluggable authentication modules.
javax.security.auth.x500	This package contains the classes that should be used to store X500 Principal and X500 Private Crendentials in a *Subject*.
javax.security.cert	Provides classes for public key certificates.
javax.security.sasl	Contains class and interfaces for supporting SASL.
javax.sound.midi	Provides interfaces and classes for I/O, sequencing, and synthesis of MIDI (Musical Instrument Digital Interface) data.
javax.sound.midi.spi	Supplies interfaces for service providers to implement when offering new MIDI devices, MIDI file readers and writers, or sound bank

	readers.
javax.sound.sampled	Provides interfaces and classes for capture, processing, and playback of sampled audio data.
javax.sound.sampled.spi	Supplies abstract classes for service providers to subclass when offering new audio devices, sound file readers and writers, or audio format converters.
javax.sql	Provides the API for server side data source access and processing from the Java™ programming language.
javax.sql.rowset	Standard interfaces and base classes for JDBC RowSet implementations.
javax.sql.rowset.serial	Provides utility classes to allow serializable mappings between SQL types and data types in the Java programming language.
javax.sql.rowset.spi	The standard classes and interfaces that a third party vendor has to use in its implementation of a synchronization provider.
javax.swing	Provides a set of "lightweight" (all-Java language) components that, to the maximum degree possible, work the same on all platforms.
javax.swing.border	Provides classes and interface for drawing specialized borders around a Swing component.
javax.swing.colorchooser	Contains classes and interfaces used by the JColorChooser component.
javax.swing.event	Provides for events fired by Swing components.
javax.swing.filechooser	Contains classes and interfaces used by the JFileChooser component.
javax.swing.plaf	Provides one interface and many abstract classes that Swing uses to provide its pluggable look-and-feel capabilities.
javax.swing.plaf.basic	Provides user interface objects built according to the Basic look and feel.
javax.swing.plaf.metal	Provides user interface objects built according to the Java look and feel (once codenamed *Metal*), which is the default look and feel.
javax.swing.plaf.multi	Provides user interface objects that combine two or more look and feels.
javax.swing.plaf.nimbus	Provides user interface objects built according to the cross-platform Nimbus look and feel.

javax.swing.plaf.synth	Synth is a skinnable look and feel in which all painting is delegated.
javax.swing.table	Provides classes and interfaces for dealing with javax.swing.JTable.
javax.swing.text	Provides classes and interfaces that deal with editable and noneditable text components.
javax.swing.text.html	Provides the class HTMLEditorKit and supporting classes for creating HTML text editors.
javax.swing.text.html.parser	Provides the default HTML parser, along with support classes.
javax.swing.text.rtf	Provides a class (RTFEditorKit) for creating Rich-Text-Format text editors.
javax.swing.tree	Provides classes and interfaces for dealing with javax.swing.JTree.
javax.swing.undo	Allows developers to provide support for undo/redo in applications such as text editors.
javax.tools	Provides interfaces for tools which can be invoked from a program, for example, compilers.
javax.transaction	Contains three exceptions thrown by the ORB machinery during unmarshalling.
javax.transaction.xa	Provides the API that defines the contract between the transaction manager and the resource manager, which allows the transaction manager to enlist and delist resource objects (supplied by the resource manager driver) in JTA transactions.
javax.xml	
javax.xml.bind	Provides a runtime binding framework for client applications including unmarshalling, marshalling, and validation capabilities.
javax.xml.bind.annotation	Defines annotations for customizing Java program elements to XML Schema mapping.
javax.xml.bind.annotation.adapters	XmlAdapter and its spec-defined sub-classes to allow arbitrary Java classes to be used with JAXB.
javax.xml.bind.attachment	This package is implemented by a MIME-based package processor that enables the interpretation and creation of optimized binary data within an MIME-based package format.
javax.xml.bind.helpers	**JAXB Provider Use Only:** Provides partial default implementations for some of the javax.xml.bind interfaces.
javax.xml.bind.util	Useful client utility classes.

javax.xml.crypto	Common classes for XML cryptography.
javax.xml.crypto.dom	DOM-specific classes for the javax.xml.crypto package.
javax.xml.crypto.dsig	Classes for generating and validating XML digital signatures.
javax.xml.crypto.dsig.dom	DOM-specific classes for the javax.xml.crypto.dsig package.
javax.xml.crypto.dsig.keyinfo	Classes for parsing and processing KeyInfo elements and structures.
javax.xml.crypto.dsig.spec	Parameter classes for XML digital signatures.
javax.xml.datatype	XML/Java Type Mappings.
javax.xml.namespace	XML Namespace processing.
javax.xml.parsers	Provides classes allowing the processing of XML documents.
javax.xml.soap	Provides the API for creating and building SOAP messages.
javax.xml.stream	
javax.xml.stream.events	
javax.xml.stream.util	
javax.xml.transform	This package defines the generic APIs for processing transformation instructions, and performing a transformation from source to result.
javax.xml.transform.dom	This package implements DOM-specific transformation APIs.
javax.xml.transform.sax	This package implements SAX2-specific transformation APIs.
javax.xml.transform.stax	Provides for StAX-specific transformation APIs.
javax.xml.transform.stream	This package implements stream- and URI-specific transformation APIs.
javax.xml.validation	This package provides an API for validation of XML documents.
javax.xml.ws	This package contains the core JAX-WS APIs.
javax.xml.ws.handler	This package defines APIs for message handlers.

javax.xml.ws.handler.soap	This package defines APIs for SOAP message handlers.
javax.xml.ws.http	This package defines APIs specific to the HTTP binding.
javax.xml.ws.soap	This package defines APIs specific to the SOAP binding.
javax.xml.ws.spi	This package defines SPIs for JAX-WS.
javax.xml.ws.spi.http	Provides HTTP SPI that is used for portable deployment of JAX-WS web services in containers(for e.g.
javax.xml.ws.wsaddressing	This package defines APIs related to WS-Addressing.
javax.xml.xpath	This package provides an *object-model neutral* API for the evaluation of XPath expressions and access to the evaluation environment.
org.ietf.jgss	This package presents a framework that allows application developers to make use of security services like authentication, data integrity and data confidentiality from a variety of underlying security mechanisms like Kerberos, using a unified API.
org.omg.CORBA	Provides the mapping of the OMG CORBA APIs to the Java™ programming language, including the class ORB, which is implemented so that a programmer can use it as a fully-functional Object Request Broker (ORB).
org.omg.CORBA_2_3	The CORBA_2_3 package defines additions to existing CORBA interfaces in the Java[tm] Standard Edition 6. These changes occurred in recent revisions to the CORBA API defined by the OMG. The new methods were added to interfaces derived from the corresponding interfaces in the CORBA package. This provides backward compatibility and avoids breaking the JCK tests.
org.omg.CORBA_2_3.portable	Provides methods for the input and output of value types, and contains other updates to the org/omg/CORBA/portable package.
org.omg.CORBA.DynAnyPackage	Provides the exceptions used with the DynAny interface (InvalidValue, Invalid, InvalidSeq, and TypeMismatch).
org.omg.CORBA.ORBPackage	Provides the exception InvalidName, which is thrown by the method ORB.resolve_initial_references and the exception

	InconsistentTypeCode, which is thrown by the Dynamic Any creation methods in the ORB class.
org.omg.CORBA.portable	Provides a portability layer, that is, a set of ORB APIs that makes it possible for code generated by one vendor to run on another vendor's ORB.
org.omg.CORBA.TypeCodePackage	Provides the user-defined exceptions BadKind and Bounds, which are thrown by methods in in the class TypeCode.
org.omg.CosNaming	Provides a naming service for Java IDL.
org.omg.CosNaming.NamingContextExtPackage	This package contains the following classes, which are used in org.omg.CosNaming.NamingContextExt:
org.omg.CosNaming.NamingContextPackage	This package contains Exception classes for the org.omg.CosNaming package.
org.omg.Dynamic	This package contains the Dynamic module specified in the OMG Portable Interceptor specification, http://cgi.omg.org/cgi-bin/doc?ptc/2000-08-06, section 21.9.
org.omg.DynamicAny	Provides classes and interfaces that enable traversal of the data value associated with an any at runtime, and extraction of the primitive constituents of the data value.
org.omg.DynamicAny.DynAnyFactoryPackage	This package contains classes and exceptions from the DynAnyFactory interface of the DynamicAny module specified in the OMG *The Common Object Request Broker: Architecture and Specification*, http://cgi.omg.org/cgi-bin/doc?formal/99-10-07, section 9.2.2.
org.omg.DynamicAny.DynAnyPackage	This package contains classes and exceptions from the DynAny interface of the DynamicAny module specified in the OMG *The Common Object Request Broker: Architecture and Specification*, http://cgi.omg.org/cgi-bin/doc?formal/99-10-07, section 9.2.
org.omg.IOP	This package contains the IOP module specified in the OMG document *The Common Object Request Broker: Architecture and Specification*, http://cgi.omg.org/cgi-bin/doc?formal/99-10-07, section 13.6.
org.omg.IOP.CodecFactoryPackage	This package contains the exceptions specified in the IOP::CodeFactory interface (as part of the Portable Interceptors spec).

org.omg.IOP.CodecPackage	This package is generated from the IOP::Codec IDL interface definition.
org.omg.Messaging	This package contains the Messaging module specified in the OMG CORBA Messaging specification, http://cgi.omg.org/cgi-bin/doc?formal/99-10-07.
org.omg.PortableInterceptor	Provides a mechanism to register ORB hooks through which ORB services can intercept the normal flow of execution of the ORB.
org.omg.PortableInterceptor.ORBInitInfoPackage	This package contains the exceptions and typedefs from the ORBInitInfo local interface of the PortableInterceptor module specified in the OMG Portable Interceptor specification, http://cgi.omg.org/cgi-bin/doc?ptc/2000-08-06, section 21.7.2.
org.omg.PortableServer	Provides classes and interfaces for making the server side of your applications portable across multivendor ORBs.
org.omg.PortableServer.CurrentPackage	Provides method implementations with access to the identity of the object on which the method was invoked.
org.omg.PortableServer.POAManagerPackage	Encapsulates the processing state of the POAs it is associated with.
org.omg.PortableServer.POAPackage	Allows programmers to construct object implementations that are portable between different ORB products.
org.omg.PortableServer.portable	Provides classes and interfaces for making the server side of your applications portable across multivendor ORBs.
org.omg.PortableServer.ServantLocatorPackage	Provides classes and interfaces for locating the servant.
org.omg.SendingContext	Provides support for the marshalling of value types.
org.omg.stub.java.rmi	Contains RMI-IIOP Stubs for the Remote types that occur in the java.rmi package.
org.w3c.dom	Provides the interfaces for the Document Object Model (DOM) which is a component API of the Java API for XML Processing.
org.w3c.dom.bootstrap	
org.w3c.dom.events	
org.w3c.dom.ls	

org.xml.sax	This package provides the core SAX APIs.
org.xml.sax.ext	This package contains interfaces to SAX2 facilities that conformant SAX drivers won't necessarily support.
org.xml.sax.helpers	This package contains "helper" classes, including support for bootstrapping SAX-based applications.