SRM VALLIAMMAI ENGINEERING COLLEGE

(An Autonomous Institution)

S.R.M. Nagar, Kattankulathur - 603 203.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

1904306 OBJECT ORIENTED PROGRAMMING LABORATORY

Regulation 2019

II YEAR /III SEMESTER

Prepared By,

MR.G.KUMARESAN, AP (Sel.G)
MS.S.SHANTHI., AP (Sr.G)
MS.C.PABITHA., AP (Sr.G)

1904306 **OBJECT ORIENTED PROGRAMMING LABORATORY** **L T P C**

0 0 4 2

COURSE OBJECTIVES :

- To apply the concepts of classes.
- To understand and implement packages and interfaces.
- To handle I/O and exception handling.
- To understand file processing operations.
- To develop applications using event handling.

List of experiments

1. Write a java program to illustrate the concept of class and object creation.
2. Write a java program to implement constructors.
3. Write a java program to implement abstract class and abstract method.
4. Write a java program to implement Inheritance.
5. Write a java program to implement I/O, Throwing and Catching exceptions.
6. Write a java program to implement Designing Packages.
7. Write a java program to implement Interfaces in Java.
8. Write a java program to manipulate file operations.
9. Write a java program to create multithreads in Java applications.
10. Write a java program to implement Graphics classes
11. Write a java program to implement Event driven programming.

TOTAL : 60 PERIODS

COURSE OUTCOMES :

Upon completion of the course, the students will be able to

- Develop and implement Java programs for simple applications that make use of classes.
- Develop and implement packages and interfaces.
- Develop and implement Java programs with files and I/O.
- Develop and implement Java programs using exception handling and multithreading.
- Develop and implement applications using event handling.

Definitions of class, public, static, void, main, String[], System.out.println().

- **class** keyword is used to declare a class in java.
- **public** keyword is an access modifier which represents visibility. It means it is visible to all.
- **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is no need to create an object to invoke the static method.

The main method is executed by the JVM, so it doesn't require to create an object to invoke the main method. So it saves memory.

- **void** is the return type of the method. It means it doesn't return any value.
- **main** represents the starting point of the program.
- **String[] args** is used for command line argument. We will learn it later.
- **System.out.println()** is used to print statement. Here, System is a class, out is the object of PrintStream class, println() is the method of PrintStream class. We will learn about the internal working of System.out.println statement later.

Ex No 1 **Write a java program to illustrate the concept of class and object creation.**

AIM:

To write a java program to find volume of box.

ALGORITHM:

1. Declare the class box.
2. Declare height,width and depth in write data.
3. Declare another class example.
4. Create an object and access write data.
5. Print the output.

PROGRAM:

```
   class Box
{
 int width=2,height=2,depth=2,a;
 void writedata()
 {
 System.out.print("volume of box is:");
 System.out.println(width*height*depth);
 }
}
public class example
{
 public static void main(String args[])
 {
 Box mybox1=new Box();
 mybox1.writedata();
 }
 }
```

OUTPUT:

```
c:\jdk1.3\bin>java example
volume of box is 8
```

RESULT:

Thus a java program is executed for finding volume of box using objects and class.

a) Implementing Class & Objects

Aim: To write a JAVA program to implement class mechanism. – Create a class, methods and invoke them inside main method

Programs:

1.no return type and without parameter-list:

```
class A
{
int l=10,b=20;
void display()
{
System.out.println(l);
System.out.println(b);
}
}
class methoddemo
{
public static void main(String args[])
{
A a1=new A();
a1.display();
}
}
```

Output:

10
20

2.no return type and with parameter-list:

```
class A
{
void display(int l,int b)
{
System.out.println(l);
System.out.println(b);
}
}
class methoddemo
{
public static void main(String args[])
{
A a1=new A();
a1.display(10,20);
}
}
```

Output:

10
20

3. return type and without parameter-list

```
class A
```

```
{
int l=10,b=20;
        int area()
        {
                return l*b;
        }
 }
 class methoddemo
 {
         public static void main(String args[])
         {
                 A a1=new A(); int r=a1.area();
                 System.out.println("The area
                 is: "+r);
         }
 }
```

Output:

The area is:200

4.return type and with parameter-list:

```
class A
{
        int area(int l,int b)
        {
                return l*b;
        }
}
class methoddemo
{
        public static void main(String args[])
        {
                A a1=new A();
               int r=a1.area(10,20);
               System.out.println("The area is:"+r);
        }
}
```

Output:

The area is:200

RESULT: The Programs to implement objects and class in java is successfully completed.

Ex No	Write a java program to implement constructors.

Aim: To write a JAVA program to implement constructor

Programs:

(i)A constructor with no parameters:

```
class A
{
int l,b;
A()
{
l=10;
b=20;
}
int area()
{
return l*b;
}
}
class constructordemo
{
public static void main(String args[])
{
A a1=new A();
int r=a1.area();
System.out.println("The area is: "+r);
}
}
```

Output:

The area is:200

(ii)A constructor with parameters

```
class A
{
int l,b;
A(int u,int v)
{
l=u;
b=v;
}
int area()
{
return l*b;
}
}
class constructordemo
{
public static void main(String args[])
{
A a1=new A(10,20);
int r=a1.area();
```

```
System.out.println("The area is: "+r);
}
}
```

Output:

The area is:200

Result: The Java Program to implement the concept of constructor is successfully executed.

Ex No	Write a java program to implement abstract class and abstract method.

Aim: To write a java program for abstract class to find areas of different shapes

Program:

```
abstract class shape
{
abstract double area();
}
class rectangle extends shape
{
double l=12.5,b=2.5;
double area()
{
return l*b;
}
}
class triangle extends shape
{
double b=4.2,h=6.5;
double area()
{
return 0.5*b*h;
}
}
class square extends shape
{
double s=6.5;
double area()
{
return 4*s;
}
}
class shapedemo
{
public static void main(String[] args)
{
rectangle r1=new rectangle();
triangle t1=new triangle();
square s1=new square();
System.out.println("The area of rectangle is: "+r1.area());
System.out.println("The area of triangle is: "+t1.area());
System.out.println("The area of square is: "+s1.area());
}
}
```

Output:

```
The area of rectangle is: 31.25
The area of triangle is: 13.65
The area of square is: 26.0
```

Result: The program to implement abstract class in java is completed successfully.

b) Program to demonstrate abstract class and method

```
// A java program to demonstrate
// use of abstract keyword.

// abstract class
abstract class A {
	// abstract method
	// it has no body
	abstract void m1();

	// concrete methods are still
	// allowed in abstract classes
	void m2()
	{
		System.out.println("This is "
						+ "a concrete method.");
	}
}

// concrete class B
class B extends A {
	// class B must override m1() method
	// otherwise, compile-time
	// exception will be thrown
	void m1()
	{
		System.out.println("B's "
						+ "implementation of m2.");
	}
}

// Driver class
public class AbstractDemo {
	public static void main(String args[])
	{
		B b = new B();
		b.m1();
		b.m2();
	}
}
```

Output:

```
B's implementation of m2.

This is a concrete method.
```

Result: The program to implement abstract class and method in java is completed successfully.

Ex No	Write a java program to implement Inheritance.

a)Implementing Single Inheritance

Aim: To write a JAVA program to implement Single Inheritance

Program:

```
class A
{
A()
{
System.out.println("Inside A's Constructor");
}
}
class B extends A
{
B()
{
System.out.println("Inside B's Constructor");
}
}
class singledemo
{
public static void main(String args[])
{
B b1=new B();
}
}
```

Output:

```
Inside A's Constructor
Inside B's Construcor
```

b) Aim: To write a JAVA program to implement multi level Inheritance

Program:

```
class A
{
A()
{
System.out.println("Inside A's Constructor");
}
}
class B extends A
{
B()
{
System.out.println("Inside B's Constructor");
}
}
class C extends B
{
C()
{
System.out.println("Inside C's Constructor");
}
}
class multidemo
{
public static void main(String args[])
{
C c1=new C();
}
}
```

Output:

```
Inside A's Constructor
Inside B's Constructor
Inside C's Constructor
```

Ex. No: 5 Implementation of Inheritance

AIM

To develop a java application with Employee class with Emp_name, Emp_id, Address, Mail_id, Mobile_no as members. Inherit the classes, Programmer, Assistant Professor, Associate Professor and Professor from employee class. Add Basic Pay (BP) as the member of all the inherited classes with 97% of BP as DA, 10 % of BP as HRA, 12% of BP as PF, 0.1% of BP for staff club fund. Generate pay slips for the employees with their gross and net salary.

PROCEDURE:

To execute a java program we require setting a class path:
1.C: \set path= C:\Program Files\Java\jdk1.6.0\bin;.;
2.C:\javac Employee.java
3.C:\java Employee

SOURCE CODE:

```
// Payslip.java

import java.util.Scanner;
class Employee
{
   String Emp_name,Mail_id,Address,Emp_id, Mobile_no;
   double BP,GP,NP,DA,HRA,PF,CF;
   Scanner get = new Scanner(System.in);
   Employee()
   {
      System.out.println("Enter Name of the Employee:");
      Emp_name = get.nextLine();
      System.out.println("Enter Address of the Employee:");
      Address = get.nextLine();
      System.out.println("Enter ID of the Employee:");
      Emp_id = get.nextLine();
      System.out.println("Enter Mobile Number:");
      Mobile_no = get.nextLine();
         System.out.println("Enter E-Mail ID of the Employee :");
         Mail_id = get.nextLine();
   }
    void display()
   {
      System.out.println("Employee Name: "+Emp_name);
      System.out.println("Employee Address: "+Address);
      System.out.println("Employee ID: "+Emp_id);
      System.out.println("Employee Mobile Number: "+Mobile_no);
System.out.println("Employee E-Mail ID"+Mail_id);
         DA=BP*0.97;
         HRA=BP*0.10;
         PF=BP*0.12;
         CF=BP*0.01;
         GP=BP+DA+HRA+PF;
         NP=GP-PF-CF;
```

```
System.out.println("Basic Pay :"+BP);
        System.out.println("Dearness Allowance : "+DA);
        System.out.println("House Rent Allowance :"+HRA);
        System.out.println("Provident Fund :"+PF);
        System.out.println("Club Fund :"+CF);

        System.out.println("Gross Pay :"+GP);
        System.out.println("Net Pay :"+NP);
    }
}
 class Programmer extends Employee
{
    Programmer()
    {
        System.out.println("Enter Basic pay of the Programmer:");
        BP = get.nextFloat();
    }
    void display()
    {
        System.out.println("============================="+"\n"+"Programmar Pay
Slip"+"\n"+"============================="+"\n");
        super.display();
    }
}
 class Assistant_Professor extends Employee
{
    Assistant_Professor()
    {
        System.out.println("Enter Basic pay of the Assistant Professor:");
        BP = get.nextFloat();
    }
    void display()
    {
        System.out.println("============================="+"\n"+"Assistant Professor
                                    Pay Slip"+"\n"+"============================="+"\n");
        super.display();
    }
}
class Associate_Professor extends Employee
{
    Associate_Professor()
    {
        System.out.println("Enter Basic pay of the Professor:");
        BP = get.nextFloat();
    }
    void display()
    {
        System.out.println("============================="+"\n"+"Associate Professor Pay
Slip"+"\n"+"============================="+"\n");
        super.display();
```

```
    }
}

class Professor extends Employee
{
    Professor()
    {
        System.out.println("Enter Basic pay of the Professor:");
        BP = get.nextFloat();
    }
    void display()
    {
        System.out.println("=============================="+"\n"+"Professor Pay
Slip"+"\n"+"=============================="+"\n");
        super.display();
    }
}

class Payslip
{
    public static void main(String args[])
    {
            String pos;
            Scanner get = new Scanner(System.in);
            System.out.println("\nEnter Employee Position :");
            pos=get.nextLine();




if(pos.equals("Programmer")||pos.equals("programmer"))
            {
              Programmer p=new Programmer();
              p.display();
            }
            if(pos.equals("AP")||pos.equals("ap"))
            {
                    Assistant_Professor AP=new Assistant_Professor();
                    AP.display();
            }
            if(pos.equals("ASSOCIATE")||pos.equals("associate"))
            {
                  Associate_Professor ASP=new Associate_Professor();
                    ASP.display();
            }
            if(pos.equals("PROFESSOR")||pos.equals("professor"))
            {
                    Professor PR=new Professor();
                    PR.display();
```

```
        }
    }
}
```

INPUT AND OUTPUT:

```
Enter Employee Position :professor
Enter Name of the Employee:raj
Enter Address of the Employee:127 RR street chennai
Enter ID of the Employee:345
Enter Mobile Number:9797342543
Enter E-Mail ID of the Employee :raj@gmail.com
Enter Basic pay of the Professor:3000
==============================
Professor Pay Slip
==============================

Employee Name: raj
Employee Address: 127 RR street chennai
Employee ID: 345
Employee Mobile Number: 9797342543
Employee E-Mail IDraj@gmail.com
Basic Pay :3000.0
Dearness Allowance : 2910.0
House Rent Allowance :300.0
Provident Fund :360.0
Club Fund :30.0
Gross Pay :6570.0
Net Pay :6180.0
```

RESULT:

Thus the Employee class was created and the Programmer, Assistant Professor, Associate Professor and Professor are inherited to the employee class and pay slip was generated.

Ex No:6A

Java Exception Handling

AIM:

To write a java program for creating simple java exception handling.

PROGRAM:

```
import java.util.Scanner;
class Division {
  public static void main(String[] args) {

  int a, b, result;

  Scanner input = new Scanner(System.in);
  System.out.println("Input two integers");

  a = input.nextInt();
  b = input.nextInt();

  result = a / b;

  System.out.println("Result = " + result);
  }
}
```

OUTPUT:

```
E:\Java>javac Division.java

E:\Java>java Division
Input two integers
4 2
Result = 2

E:\Java>java Division
Input two integers
7 0
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Division.main(Division.java:14)

E:\Java>_
```

RESULT:

Thus the java program was executed and output verified.

Ex No:6B **Java Exception Handling**

AIM: To write a java program for creating simple java exception handling.

PROGRAM:

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

  int a, b, result;

  Scanner input = new Scanner(System.in);
  System.out.println("Input two integers");

  a = input.nextInt();
  b = input.nextInt();

  // try block

  try {
    result  = a / b;
    System.out.println("Result = " + result);
  }

  // catch block

  catch (ArithmeticException e) {
    System.out.println("Exception caught: Division by zero.");
  }
  }
}
```

RESULT: Thus the java program was executed and output verified.

Ex No:6C **Java Exception Handling**

AIM: To write a java program for creating simple java exception handling.

PROGRAM:

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

  String languages[] = { "C", "C++", "Java", "Perl", "Python" };

  try {
    for (int c = 1; c <= 5; c++) {
      System.out.println(languages[c]);
    }
  }
  catch (Exception e) {
    System.out.println(e);
  }
  }
}
```

OUTPUT

```
C++
Java
Perl
Python
java.lang.ArrayIndexOutOfBoundsException: 5
```

RESULT: Thus the java program was executed and output verified.

Ex No:6D **Java Exception Handling**

AIM: To write a java program for creating simple java exception handling.

PROGRAM:

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

  try {
    long data[] = new long[1000000000];
  }
  catch (Exception e) {
    System.out.println(e);
  }
  finally {
    System.out.println("finally block will execute always.");
  }
  }
}
```

OUTPUT :

```
finally block will execute always.

Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
at Allocate.main(Allocate.java:5)
```

RESULT: Thus the java exception handling program was executed and output verified.

Ex No:6E **Java User Defined Exception Handling**

AIM: To write a java program for creating simple java user defined exception handling.

PROGRAM:

```
public class UserDefinedExceptionExample {

	public static void main(String[] args) throws Exception {

		try {

			throw new UserDefinedException();

		} catch (Exception e) {
			e.printStackTrace();
		}
	}
}

public class UserDefinedException extends Exception{

}
```

OUTPUT:

```
UserDefinedException
at UserDefinedExceptionExample.main(UserDefinedExceptionExample.java:9)
```

RESULT: Thus the java user defined exception handling program was executed and output verified.

Ex.No.7 Packages

a) Illustration of class path:

Aim: To write a JAVA program illustrate class path

```
import java.net.URL;
import java.net.URLClassLoader;
public class App
{
        public static void main(String[] args)
        {
                ClassLoader sysClassLoader =
                ClassLoader.getSystemClassLoader(); URL[] urls =
                ((URLClassLoader)sysClassLoader).getURLs();
                for(int i=0; i< urls.length; i++)
                {
                        System.out.println(urls[i].getFile());
                }
        }
}
```

OUTPUT:

E:/java%20work/

b) A case study on including in class path in os environment

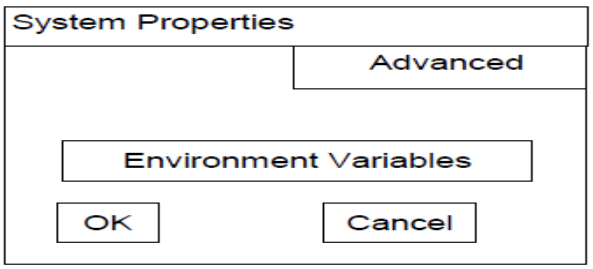
Aim: To write a case study on including in class path in your os environment of your package.

- The differences between path and classpath are given by.
  (1) The PATH is an environment variable used to locate "java" or "javac" command, to run java program and compile java source file. The CLASSPATH is an environment variable used to set
  path for java classes.
  (2) In order to set PATH in Java, we need to include **bin** directory in PATH environment while in order to set CLASSPATH we need to include all directories where we have put either our .class file
  or JAR file, which is required by our Java application.
  (3) PATH environment variable is used by operating system while CLASSPATH is used by Java
  ClassLoaders to load class files.
  (4) Path refers to the system while classpath refers to the Developing Environment.
- By default the java run time system uses the current working directory.
- Normally to execute a java program in any directory we have to set the path by as follows set path= c:\Program Files\java\jdk1.5.0_10\bin;

Setting environmental variable in windows xp:

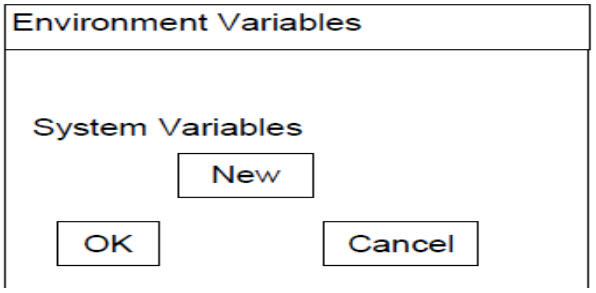
Step-1:

- Select My computer on the desktop and right click the mouse and then select properties
- It dispalys the system Properties dialogue



Step-2:

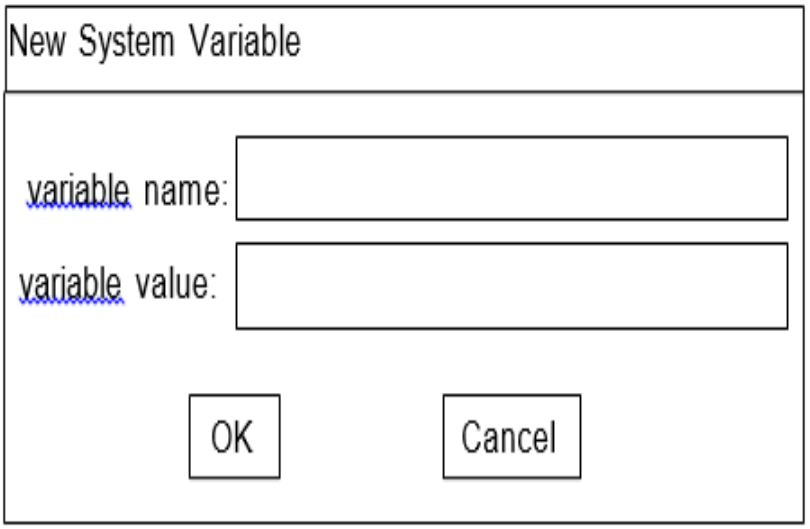
- In System Properties click Advanced and then click Environment Variables.
- It displays the following "Environment Variables" dialog.



Step -3

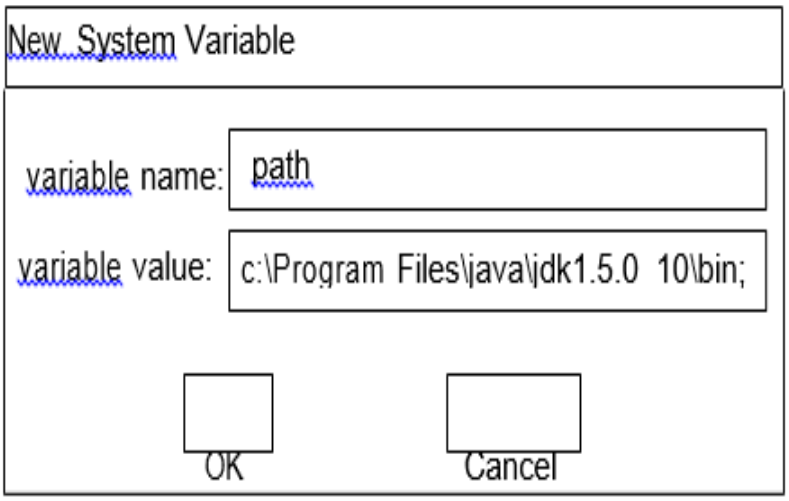
- In Environment Variables click New in System variables.

- It displays the following “New System Variable” dialog box.



Step-4:

- Now type variable name as a path and then variable value as c:\Program Files\java\jdk1.5.0_10\bin;



Step-5:

- Click OK

c) Creating and importing a package

Aim: To write a JAVA program that import and use the defined your package in the previous Problem

(i)Creating a package

Steps:

1.First declare the name of the package using te package keyword

Example: package mypack;

2. Type the following program under this package statement. In package : class ,data, methods all are **public**

```
package mypack;
public class box
{
    public int l=10,b=20;
    public void display()
    {
    System.out.println(l);
    System.out.println(b)
    }
}
```

3. Create sub directory with a name same that of package name under the current working directory by as follows. d:\>md mypack
4. Under this subdirectory store the above program with a file name “box.java”.

(ii) importing a package:

Steps:

1. packages can be accessed by using the import statement

   General form: import pack1[.pack2].(classname/*); Example:
   import java.io.*;
   Here pack1 is name of top level package and pack2 is name of sub package
2. Type the following program under the current working directory and save the program with a file name “example.java”.

```
Import mypack.box;
Class packagedemo
{
    public static void main(String args[])
    {
        box b1=new box();
        b1.display();
    }
}
```

3. Now compile the above program in the current working directory d:\

   javac packagedemo.java
4. Execute the above program in current working

   directory java packagedemo

Output:
10
20

Ex No:9 **Java Program using Files**

AIM:
To write a Java program that reads a file name from the user, displays information about whether the file exists, whether the file is readable, or writable, the type of file and the length of the file in bytes.

PROGRAM:

```
import java.io.*;
public class FileInfo
{
    public static void main(String[] args) throws IOException
   {
       BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      System.out.print("\nEnter A File Name:");
      String fName = br.readLine();
       File f = new File(fName);
      String result = f.exists() ? "exists." : "does not exist.";
      System.out.println("\nThe given file " + result);
      if(f.exists())
      {
         result = f.canRead() ? "readable." : "not readable.";
         System.out.println("The given file is " + result);
          result = f.canWrite() ? "writable." : "not writable.";
         System.out.println("The given file is " + result);

         if (fName.endsWith(".jpg") || fName.endsWith(".gif") || fName.endsWith(".png"))
         {
            System.out.println("The given file is an image file.");
         }
         else if (fName.endsWith(".exe"))
         {
            System.out.println("The given file is an executable file.");
         }
         else if (fName.endsWith(".txt"))
         {
            System.out.println("The given file is a text file.");
         }
         else
         {
            System.out.println("The file type is unknown.");
         }
      }
   }
}
```

OUTPUT:

```
F:\PU>java FileInfo

Enter A File Name:sample.txt

The given file exists.
The given file is readable.
The given file is writable.
The given file length is 0 in bytes.
The given file is a text file.
```

RESULT:
Thus the java program to implement files was executed and output verified.

Ex No: 10 **Multithreaded Application**

AIM:

To write a java program that implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

PROGRAM:

```
import java.util.*;
class even implements Runnable {
public int x;
public even(int x) {
this.x = x;
}
public void run() {
System.out.println("Thread Name:Even Thread and " + x + "is even Number and Square of " + x
+ " is: " + x * x);
}
}
class odd implements Runnable {
public int x;
public odd(int x) {
this.x = x;
}
public void run() {
System.out.println("Thread Name:ODD Thread and " + x + " is odd number and Cube of " + x +
" is: " + x * x * x);
}
}
class A extends Thread {
public String tname;
public Random r;
public Thread t1, t2;
public A(String s) {
tname = s;
}
public void run() { int num = 0;
r = new Random();
try {
for (int i = 0; i < 50; i++) {
num = r.nextInt(100);
System.out.println("Main Thread and Generated Number is " + num);
if (num % 2 == 0) {
t1 = new Thread(new even(num));
t1.start();
} else {
t2 = new Thread(new odd(num));
t2.start();
}
Thread.sleep(1000);
System.out.println("--------------------------------------");
```

```
}
}
catch (Exception ex)
{
System.out.println(ex.getMessage());
}
}
}
public class Mthread
{
public static void main(String[] args)
{
A a = new A("One");
a.start();
}
}
```

OUTPUT

```
run:
Main Thread and  Generated Number is 95
Thread Name:ODD Thread and 95 is odd number and Cube of 95 is: 857375
---------------------------------------
Main Thread and  Generated Number is 90
Thread Name:Even Thread and 90is even Number and Square of 90 is: 8100
---------------------------------------
Main Thread and  Generated Number is 35
Thread Name:ODD Thread and 35 is odd number and Cube of 35 is: 42875
---------------------------------------
Main Thread and  Generated Number is 41
Thread Name:ODD Thread and 41 is odd number and Cube of 41 is: 68921
---------------------------------------
Main Thread and  Generated Number is 75
Thread Name:ODD Thread and 75 is odd number and Cube of 75 is: 421875
---------------------------------------
Main Thread and  Generated Number is 29
Thread Name:ODD Thread and 29 is odd number and Cube of 29 is: 24389
---------------------------------------
Main Thread and  Generated Number is 55
Thread Name:ODD Thread and 55 is odd number and Cube of 55 is: 166375
---------------------------------------
Main Thread and  Generated Number is 84
Thread Name:Even Thread and 84is even Number and Square of 84 is: 7056
---------------------------------------
Main Thread and  Generated Number is 36
Thread Name:Even Thread and 36is even Number and Square of 36 is: 1296
---------------------------------------
Main Thread and  Generated Number is 51
Thread Name:ODD Thread and 51 is odd number and Cube of 51 is: 132651
---------------------------------------
```

RESULT:

Thus the java program to implement multithreaded application was executed and output verified.

a)Extending Thread class

Aim: To write a JAVA program that creates threads by extending Thread class First thread display “Good Morning “every 1 sec, the second thread displays “Hello “every 2 seconds and the third display “Welcome” every 3 seconds ,(Repeat the same by implementing Runnable)

Programs:

(i)Creating multiple threads using Thread class

```
class A extends Thread
{
        public void run()
        {
            try
            {
                for(int i=1;i<=10;i++)
                {
                    sleep(1000);
                    System.out.println("good morning");
                }
                }
                catch(Exception e)
                {
                    System.out.println(e);
                }
                }
            }

              class B extends Thread
              {
                    public void run()
                    {
                    try
                    {
                        for(int j=1;j<=10;j++)
                        sleep(2000);
                        System.out.println("hello");
                                }
                      }
                      catch(Exception e)
                      {
                            System.out.println(e);
                      }
                }
              }
              class C extends Thread
              {
                    public void run()
                    {
                        try
                        {
                            for(int k=1;k<=10;k++)
                             {
```

```
                    s                       System.out.println("welcome");
                    l                   }
                    e               }
                    e               catch(Exception e)
                    p               {
                    (                   System.out.println(e);
                    3               }
                    0   }
                }   0
                class threaddemo
                {   )
                    ;   public static void main(String args[])
                    S   {
        A a1=new A(); B
        b1=new B(); C
        c1=new C(); a1.start();
        b1.start(); c1.start();

    }
}
```

Output:

```
good morning hello
good morning good morning
welcome hello
good morning good morning
hello
good morning welcome
good morning
hello
good morning
good morning welcome
hello
good morning hello
welcome
hello welcome hello hello welcome hello welcome welcome welcome welcome
```

(ii) Creating multiple threads using Runnable interface

```
class A implements Runnable
{
    public void run()
    {
        try
        {

        for(int i=1;i<=10;i++)
        {
                Thread.sleep(1000);
                System.out.println("good morning");
            }
          }
          catch(Exception e)
          {
              System.out.println(e);
           }
    }
}
class B implements Runnable
{
    public void run()
```

```
        {
                try
                {
                for(int j=1;j<=10;j++)
                {
                Thread.sleep(2000); System.out.println("hello");
                }
                }
                catch(Exception e)
                {
                System.out.println(e);
                }
                }
                }
                class C implements Runnable
                {
                public void run()
                {
                try
                {
                for(int k=1;k<=10;k++)
                {
                                Thread.sleep(3000);
                                System.out.println("welcome");
                        }
                }
                catch(Exception e)
                {
                        System.out.println(e);
                }

        }
}
class runnabledemo
{
        public static void main(String args[])
        {
                A a1=new A(); B
                b1=new  B();  C
                c1=new C();
                Thread    t1=new    Thread(a1);
                Thread    t2=new    Thread(b1);
                Thread    t3=new    Thread(c1);
                t1.start();
                 t2.start();
                 t3.start();
       }
}
```

Output:

```
good    morning
good    morning
hello
good morning
welcome
good morning
hello
good morning
good morning
welcome
hello
good morning
good morning
```

hello
good morning
welcome
good morning
hello
welcome
hello
hello
welcome
hello
welcome
hello
hello
welcome
welcome
welcome
welcome

(b) Implementing isAlive() and join()

Aim: To write a program illustrating isAlive() and join()

Program:

```
class A extends Thread
{
        public void run()
        {
                try
                {
                        for(int i=1;i<=10;i++)
                        {
                                sleep(1000); System.out.println("good
                                morning");
                         }
                  }
                  catch(Exception e)
                  {
                        System.out.println(e);
                  }
        }
}
class B extends Thread
{
        public void run()
        {
                try
                {
                        for(int j=1;j<=10;j++)
                        {
```

```
                                sleep(2000);
                                System.out.println("hello");
                        }
                }
                catch(Exception e)
                {
                        System.out.println(e);
                }
        }
}
class C extends Thread
{
        public void run()
        {


                try
                {
                        for (int k=1;k<=10;k++)
                          {
                                sleep(3000);
                                System.out.println("welcome");
                           }
                 }
                 catch(Exception e){
                        System.out.println(e);
                 }

        }
}
class isalivedemo
{
        public static void main(String args[])
        {
                A a1=new A(); B
                b1=new B(); C
                c1=new C();
                a1.start();
                b1.start();
                c1.start();
                System.out.println(a1.isAlive());
                System.out.println(b1.isAlive());
                System.out.println(c1.isAlive()); try
                {
                        a1.join();
                        b1.join();
                        c1.join();
                }
                catch(InterruptedException e)
                {
                        System.out.println(e);
                }
                System.out.println(a1.isAlive());
                System.out.println(b1.isAlive());
                System.out.println(c1.isAlive());
        }
}
```

Output:

```
true
true true
good morning good morning
hello
```

good morning welcome
good morning
hello
good morning good morning
welcome
hello
good morning good morning
hello
good morning welcome
good morning hello
welcome
hello hello welcome hello welcome hello hello welcome welcome welcome welcome false false false

c) Implementation of Daemon Threads

Aim: To write a Program illustrating Daemon Threads

Program:

```
class A extends Thread
{
        public void run()
        {
                if(Thread.currentThread().isDaemon())
                System.out.println("daemon thread work"); else
                System.out.println("user thread work");
        }
}
class daemondemo
{
        public static void main(String[] args)
        {
                A a1=new A(); A
                a2=new   A();   A
                a3=new A();
                a1.setDaemon(true);
                a1.start();
                a2.start();
                a3.start();
         }
}
```

Output:

daemon thread work user
thread work user thread
work

d)Producer-Consumer problem

Aim: Write a JAVA program Producer Consumer Problem

Program:

```
class A
{
        int n;
        boolean b=false;
        synchronized int get()
        {
                if(!b)
                try
                {
```

```
                }
                wait();
                catch(Exception e)
                {
                        System.out.println(e);
                }
                System.out.println("Got:"+n);
                b=false;
                notify();
                return n;
        }
        synchronized void put(int n)
        {
                if(b)
                try
                {}
                wait();
                catch(Exception e)
                {
                        System.out.println(e);
                }
                this.n=n; b=true;
                System.out.println("Put:"+n);
                notify();
        }
}
class producer implements Runnable
{
        A a1; Thread t1;
        producer(A a1)
        {
                this.a1=a1;
                t1=new Thread(this);
                t1.start();
        }

        public void run()
{
for(int i=1;i<=10;i++)
                {
                        a1.put(i);
                }
        }
}

class consumer implements Runnable
{
        A a1; Thread t1;
        consumer(A a1)
        {
                this.a1=a1;
                t1=new Thread(this);
                t1.start();
        }
        public void run()
        {
                for(int j=1;j<=10;j++)
                {
                        a1.get();
                }
        }
}
class interdemo
```

```
{
        public static void main(String args[])
        {
                A a1=new A();
                producer p1=new producer(a1);
                consumer c1=new consumer(a1);
        }
}
```

Output:

```
Put:1
Got:1
Put:2
Got:2
Put:3
Got:3
Put:4
Got:4
Put:5
Got:5
Put:6
Got:6
Put:7
Got:7
Put:8
Got:8
Put:9
Got:9
Put:10
Got:10
```

Ex No: 11 **Graphics Program**

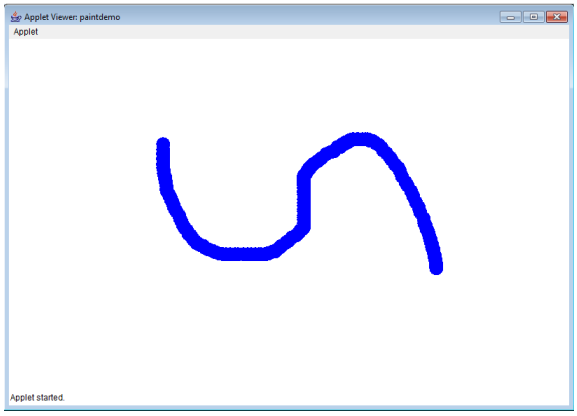
11a) Paint like Paint Brush in Applet

Aim: To write a JAVA program to paint like paint brush in applet. **Program:**

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
 //<applet code="paintdemo"  width="800" height="500"></applet>
public class paintdemo extends Applet implements MouseMotionListener
{
          int w, h; Image i;
          Graphics g1; public
          void init()
          {
                    w = getSize().width;  h = getSize().height;
                    i = createImage( w, h );
                    g1 = i.getGraphics();
                    g1.setColor( Color.white ); g1.fillRect( 0, 0, w, h ); g1.setColor( Color.red );
                    i = createImage( w, h );
                    g1 = i.getGraphics();
                    g1.setColor( Color.white ); g1.fillRect( 0, 0, w, h ); g1.setColor( Color.blue );
                    addMouseMotionListener( this );
           }
           public void mouseMoved( MouseEvent e ) { }
           public void mouseDragged( MouseEvent me )
           {
                    int x = me.getX(); int y = me.getY();
                    g1.fillOval(x-10,y-10,20,20); repaint();
                    me.consume();
           }
```

```
        public void update( Graphics g )
        {
                g.drawImage( i, 0, 0, this );
        }
        public void paint( Graphics g )
        {
                update(g);
        }
}
```

Output:



11 b) Display Analog Clock using Applet

Aim: To write a JAVA program to display analog clock using Applet.

Program:

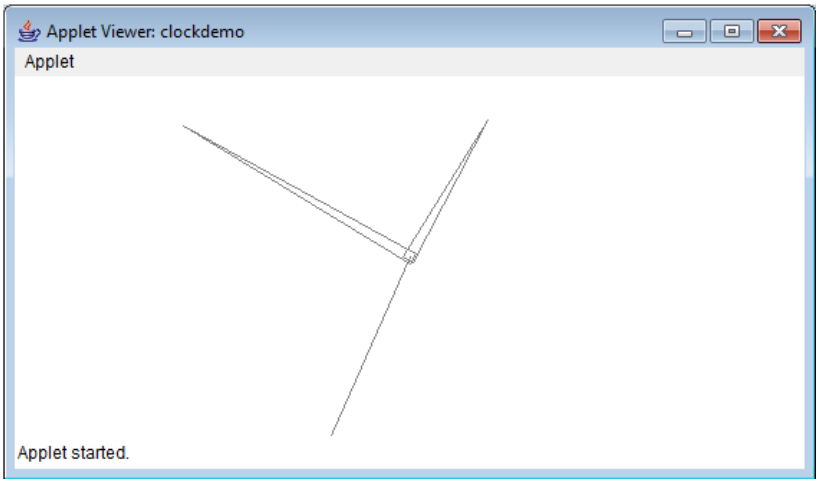
```
import java.util.*;
import java.text.*;
import java.applet.*;
import java.awt.*;
//<applet code="clockdemo" width="550" height="250"></applet
public class clockdemo extends Applet implements Runnable
{
        int h=0, m=0, s=0;
        String str=""; int wt, ht;  Thread thr=null;  boolean b;
        public void init()
        {
                wt=getSize().width; ht=getSize().height;
        }
        public void start()
        {
                if (thr==null)
                {
                        thr=new Thread(this);
                        b=false;
                        thr.start();

                }
                 else
                {
                        if(b)
                        {
                                b=false;
                                synchronized(this)
                                {
                                        notify();
                                }
                        }
                 }
         }
         public void stop()
         {
                b=true;
         }
         public void run()
         {
                 try
                 {

                        while(true)
                        {
                                Calendar        clndr=Calendar.getInstance();
                                h=clndr.get(Calendar.HOUR_OF_DAY);
                                if(h>12)h-=12;
                                m=clndr.get(Calendar.MINUTE); s=clndr.get(Calendar.SECOND);
                                SimpleDateFormat frmatter=new SimpleDateFormat("hh:mm:ss",
                                                                Locale.getDefault());
                                Date d=clndr.getTime();  str=frmatter.format(d);
                                if(b)
                                {
                                        synchronized (this)
                                        {
```

```
                                        while(b)
                                        {
                                                wait();
                                        }
                                }
                        }
                        repaint();
                        thr.sleep(1000);
                }
        }
        catch(Exception e)
        {
                System.out.println(e);
        }
    }
    void drawHand(double angle, int radius, Graphics grp)
    {
        angle-=0.5*Math.PI;
        int a=(int)(radius*Math.cos(angle)); int b=(int)(radius*Math.sin(angle));
        grp.drawLine(wt/2,ht/2,wt/2+a,ht/2+b);
    }
    void drawWedge(double angle,int radius, Graphics grp)
    {
        angle-=0.5*Math.PI;
        int a=(int)(radius*Math.cos(angle)); int b=(int)(radius*Math.sin(angle));
        angle+=2*Math.PI/3;
        int a2=(int)(5*Math.cos(angle)); int b2=(int)(5*Math.sin(angle));
        angle+=2*Math.PI/3;
        int a3=(int)(5*Math.cos(angle)); int b3=(int)(5*Math.sin(angle));
        grp.drawLine(wt/2+a2, ht/2+b2,wt/2+a,ht/2+b);
        grp.drawLine(wt/2+a3, ht/2+b3,wt/2+a,ht/2+b);
        grp.drawLine(wt/2+a2, ht/2+b2,wt/2+a3,ht/2+b3);
     }
     public void paint(Graphics grp)
     {
        grp.setColor(Color.gray);
        drawWedge(2*Math.PI*h/12,wt/5,grp);  drawWedge(2*Math.PI*m/60,wt/3,grp);
        drawHand(2*Math.PI*s/60,wt/2,grp);
     }
}
```

Output:



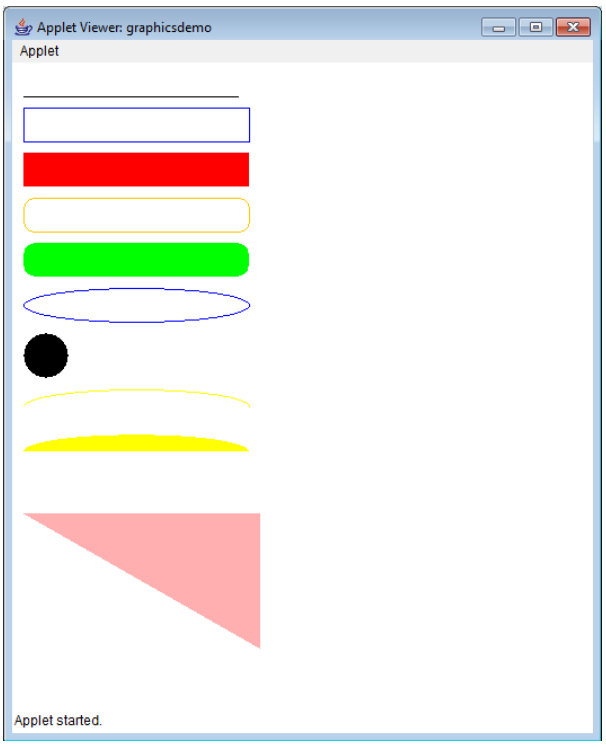
11c) Display Shapes and Fill Colors

Aim: To write a JAVA program to create different shapes and fill colors using Applet

Program:

```
import java.awt.*;
import java.applet.*;
//<applet code="graphicsdemo" width="400" height="400"></applet>
public class graphicsdemo extends Applet
{
        public void paint(Graphics g)
        {
                int x[]={10,220,220}; int
                y[]={400,400,520}; int
                n=3;
                g.drawLine(10,30,200,30);
                g.setColor(Color.blue);          g.drawRect(10,40,200,30);
                g.setColor(Color.red);           g.fillRect(10,80,200,30);
                g.setColor(Color.orange);        g.drawRoundRect(10,120,200,30,20,20);
                g.setColor(Color.green);         g.fillRoundRect(10,160,200,30,20,20);
                g.setColor(Color.blue);          g.drawOval(10,200,200,30);
                g.setColor(Color.black);         g.fillOval(10,240,40,40);
                g.setColor(Color.yellow);        g.drawArc(10,290,200,30,0,180);
                g.setColor(Color.yellow);        g.fillArc(10,330,200,30,0,180);
                g.setColor(Color.pink);          g.fillPolygon(x,y,n);
        }
}
```

Output:



Ex No:12 **Event driven Calculator**

AIM:
To Design a calculator using event-driven programming paradigm of Java with the following options.

PROGRAM:

```
import javax.swing.*;
import javax.swing.event.*;
import java.awt.*;
import java.awt.event.*;
class Calculator
{
   JFrame frame = new JFrame();
   // Creating the Menu Bar
   JMenuBar menubar = new JMenuBar();
   //---> Creating the "Calculator-->Exit" Menu
   JMenu firstmenu = new JMenu("Calculator");
   JMenuItem exitmenu = new JMenuItem("Exit");
   // Creating The TextArea that gets the value
   JTextField editor = new JTextField();
   JRadioButton degree = new JRadioButton("Degree");
   JRadioButton radians = new JRadioButton("Radians");

   String[] buttons = {"BKSP","CLR","sin","cos","tan","7","8","9","/","+/-
","4","5","6","X","x^2","1","2","3","-","1/x","0",".","=","+","sqrt"};

    JButton[] jbuttons = new JButton[26];
    double buf=0,result;
    boolean opclicked=false,firsttime=true;
    String last_op;
    public Calculator()
    {
      frame.setSize(372,270);
      frame.setTitle("Calculator - Edited By M.Baran Mahamood.");
      frame.setLayout(new BorderLayout());
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      frame.setResizable(false);

      ButtonHandler bhandler = new ButtonHandler();

      menubar.add(firstmenu);
      firstmenu.add(exitmenu);
      exitmenu.setActionCommand("mExit");
      exitmenu.addActionListener(bhandler);

      editor.setPreferredSize(new Dimension(20,50));

      Container buttoncontainer = new Container();

      buttoncontainer.setLayout(new GridLayout(5,5));
      for(int i=0;i<buttons.length;i++)
```

```
        {
         jbuttons[i] = new JButton(buttons[i]);
         jbuttons[i].setActionCommand(buttons[i]);
         jbuttons[i].addActionListener(bhandler);
         buttoncontainer.add(jbuttons[i]);
        }
        JPanel degrad = new JPanel();
        degrad.setLayout(new FlowLayout());
        ButtonGroup bg1 = new ButtonGroup();
        bg1.add(degree);
        bg1.add(radians);
        degrad.add(degree);
        radians.setSelected(true);
        degrad.add(radians);

        frame.setJMenuBar(menubar);
        frame.add(editor,BorderLayout.NORTH);
        frame.add(degrad,BorderLayout.CENTER);
        frame.add(buttoncontainer,BorderLayout.SOUTH);
        frame.setVisible(true);
    }
    public static void main(String args[])
    {
     Calculator a = new Calculator();
    }
    public class ButtonHandler implements ActionListener
    {
     public void actionPerformed(ActionEvent e)
     {
      String action = e.getActionCommand();
      if(action == "0" || action=="1" || action=="2" || action=="3" || action=="4" || action=="5" ||
action=="6" || action=="7" || action=="8" || action=="9" || action==".")
      {
       if(opclicked == false)
        editor.setText(editor.getText() + action);
       else
       {
        editor.setText(action);
        opclicked = false;
       }
      }
      if(action == "CLR")
      {
       editor.setText("");
       buf=0;
       result=0;
       opclicked=false;
       firsttime=true;
       last_op=null;
      }
      //Addition
      if(action=="+")
```

```
{
   firsttime = false;
   if(last_op!="=" && last_op!="sqrt" && last_op!="1/x" && last_op!="x^2" && last_op!="+/-")
   {
    buf = buf + Double.parseDouble(editor.getText());
    editor.setText(Double.toString(buf));
    last_op = "+";
    opclicked=true;
   }
   else
   {
    opclicked=true;
    last_op = "+";
   }
}
// Subtraction
if(action=="-")
{
  if(firsttime==true)
  {
  buf = Double.parseDouble(editor.getText());
  firsttime = false;
  opclicked=true;
  last_op = "-";
  }
  else
  {
   if(last_op!="=" && last_op!="sqrt" && last_op!="1/x" && last_op!="x^2" && last_op!="+/-")
   {
    buf = buf - Double.parseDouble(editor.getText());
    editor.setText(Double.toString(buf));
    last_op = "-";
    opclicked=true;
   }
   else
   {
    opclicked=true;
    last_op = "-";
   }
  }
}
//Multiplication
if(action=="X")
{
  if(firsttime==true)
  {
  buf = Double.parseDouble(editor.getText());
  firsttime = false;
  opclicked = true;
  last_op = "X";
  }
  else
```

```
    {
     if(last_op!="=" && last_op!="sqrt" && last_op!="1/x" && last_op!="x^2" && last_op!="+/-")
     {
      buf = buf * Double.parseDouble(editor.getText());
      editor.setText(Double.toString(buf));
      last_op = "X";
      opclicked=true;
     }
     else
     {
      opclicked=true;
      last_op = "X";
     }
    }
}
//Division
if(action=="/")
{
   if(firsttime==true)
   {
   buf = Double.parseDouble(editor.getText());
   firsttime = false;
   opclicked=true;
   last_op = "/";
   }
   else
   {
    if(last_op!="=" && last_op!="sqrt" && last_op!="1/x" && last_op!="x^2" && last_op!="+/-")
    {
     buf = buf / Double.parseDouble(editor.getText());
     editor.setText(Double.toString(buf));
     last_op = "/";
     opclicked=true;
    }
    else
    {
     opclicked=true;
     last_op = "/";
    }
   }
}
// Equal to
if(action=="=")
{
  result = buf;
  if(last_op=="+")
  {
    result = buf + Double.parseDouble(editor.getText());
    buf = result;
  }
      if(last_op=="-")
  {
```

```
          result = buf - Double.parseDouble(editor.getText());
          buf = result;
        }
            if(last_op=="X")
        {
          result = buf * Double.parseDouble(editor.getText());
          buf = result;
        }
            if(last_op=="/")
        {
          try
          {
          result = buf / Double.parseDouble(editor.getText());
          }
          catch(Exception ex)
          {
            editor.setText("Math Error " + ex.toString());

          }
          buf = result;
        }

        editor.setText(Double.toString(result));
        last_op = "=";
      }
      // Sqrt
      if(action=="sqrt")
      {

        if(firsttime==false)
        {
          buf = Math.sqrt(buf);
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "sqrt";
        }
        else
        {
          if(editor.getText()=="")
           JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
          else
          {
          buf = Double.parseDouble(editor.getText());
          buf = Math.sqrt(buf);
          editor.setText(Double.toString(buf));
          firsttime = false;
          opclicked=true;
          last_op = "sqrt";
          }
        }
```

```
    }
    // Reciprocal

    if(action=="1/x")
    {

      if(firsttime==false)
      {
        buf = 1/ buf;
        editor.setText(Double.toString(buf));
        opclicked=true;
        last_op = "1/x";
      }
      else
      {
       if(editor.getText()==null)
        JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
       else
       {
        buf = Double.parseDouble(editor.getText());
        buf = 1 / buf;
        editor.setText(Double.toString(buf));
        firsttime = false;
        opclicked=true;
        last_op = "1/x";
       }
      }

    }

    // Square

    if(action=="x^2")
    {

      if(firsttime==false)
      {
        buf =  buf * buf;
        editor.setText(Double.toString(buf));
        opclicked=true;
        last_op = "x^2";
      }
      else
      {
       if(editor.getText()==null)
        JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
       else
       {
        buf = Double.parseDouble(editor.getText());
        buf =  buf * buf;
```

```
            editor.setText(Double.toString(buf));
            firsttime = false;
            opclicked=true;
            last_op = "x^2";
           }
          }

        }

       // Negation +/-
        if(action=="+/-")
        {

          if(firsttime==false)
          {
            buf =  -(buf);
            editor.setText(Double.toString(buf));
            opclicked=true;
            last_op = "+/-";
          }
          else
          {
           if(editor.getText()==null)
            JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
           else
           {
            buf = Double.parseDouble(editor.getText());
            buf =  -(buf);
            editor.setText(Double.toString(buf));
            firsttime = false;
            opclicked=true;
            last_op = "+/-";
           }
          }

        }
        // Exit
        if(action=="mExit")
        {
                frame.dispose();
                System.exit(0);
        }
        if(action=="mCut")
          editor.cut();
        if(action=="mCopy")
          editor.copy();
        if(action=="mPaste")
          editor.paste();
        if(action=="sin")
        {
          if(radians.isSelected())
```

```
        {
         if(firsttime==false)
        {
          buf =  Math.sin(Double.parseDouble(editor.getText()));
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "sin";
        }
        else
        {
         if(editor.getText()=="")
          JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
         else
         {
          buf = Double.parseDouble(editor.getText());
          buf =  Math.sin(Double.parseDouble(editor.getText()));
          editor.setText(Double.toString(buf));
          firsttime = false;
          opclicked=true;
          last_op = "sin";
         }
        }
        }
        else
        {
         if(firsttime==false)
        {
          double rad=Math.toRadians(Double.parseDouble(editor.getText()));
          buf =  Math.sin(rad);
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "sin";
        }
        else
        {
         if(editor.getText()=="")
          JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
         else
         {
          double rad=Math.toRadians(Double.parseDouble(editor.getText()));
          buf =  Math.sin(rad);
          editor.setText(Double.toString(buf));
          firsttime = false;
          opclicked=true;
          last_op = "sin";
         }
        }
        }
      }// end of sin
```

```
      if(action=="cos")
      {
       if(radians.isSelected())
       {
        if(firsttime==false)
       {
          buf =  Math.cos(Double.parseDouble(editor.getText()));
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "cos";
       }
       else
       {
        if(editor.getText()=="")
         JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
        else
        {
         buf = Double.parseDouble(editor.getText());
         buf =  Math.sin(Double.parseDouble(editor.getText()));
         editor.setText(Double.toString(buf));
         firsttime = false;
         opclicked=true;
         last_op = "cos";
        }
       }
       }
       else
       {
        if(firsttime==false)
       {
          double rad=Math.toRadians(Double.parseDouble(editor.getText()));
          buf =  Math.cos(rad);
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "cos";
       }
       else
       {
        if(editor.getText()=="")
         JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
        else
        {
         double rad=Math.toRadians(Double.parseDouble(editor.getText()));
         buf =  Math.cos(rad);
         editor.setText(Double.toString(buf));
         firsttime = false;
         opclicked=true;
         last_op = "cos";
        }
       }
```

```
        }
       }// end of cos
       if(action=="tan")
       {
        if(radians.isSelected())
        {
         if(firsttime==false)
        {
          buf =  Math.tan(Double.parseDouble(editor.getText()));
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "tan";
        }
        else
        {
         if(editor.getText()=="")
          JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
         else
         {
          buf = Double.parseDouble(editor.getText());
          buf =  Math.tan(Double.parseDouble(editor.getText()));
          editor.setText(Double.toString(buf));
          firsttime = false;
          opclicked=true;
          last_op = "tan";
         }
        }
        }
        else
        {
         if(firsttime==false)
        {
          double rad=Math.toRadians(Double.parseDouble(editor.getText()));
          buf =  Math.tan(rad);
          editor.setText(Double.toString(buf));
          opclicked=true;
          last_op = "tan";
        }
        else
        {
         if(editor.getText()=="")
          JOptionPane.showMessageDialog(frame,"Enter input pls...","Input
Missing",JOptionPane.ERROR_MESSAGE);
         else
         {
          double rad=Math.toRadians(Double.parseDouble(editor.getText()));
          buf =  Math.tan(rad);
          editor.setText(Double.toString(buf));
          firsttime = false;
          opclicked=true;
          last_op = "tan";
```

```
            }
          }
          }
        }// end of tan
      }
    }
  }
```

RESULT:

Thus the event driven java program to implement calculator was executed and output verified.

VIVA QUESTIONS

Viva Questions for Java Lab

What is Java?
Java is an object oriented programming language

Is Java invented for Internet?
No, it was developed for programming towards tiny devices

What is byte code?
It is machine understandable code and targeted for JVM (Java Virtual Machine). It is class file.

What is JVM?
Java Virtual Machine which accepts java byte code and produces result

What are java buzzwords?
Java buzzwords explain the important features of java. They are Simple,
Secured, Portable, architecture neutral, high performance, dynamic, robust, interpreted etc.

Is byte code is similar to .obj file in C?
Yes, both are machine understandable codes
No, .obj file directly understood by machine, byte code requires JVM

What is new feature in control statements comparing with C/C++?
Labeled break and labeled continue are new

What are new features in basic features comparing with C/C++?
Data types: All data types on all machines have fixed size;
Constants: final is used to declare constant
Boolean Type: boolean is new data type in Java which can store true/false
There are no structures/unions/pointers

Is String data type?
No, Strings are classes in Java (Arrays are classes)

What are length and length( ) in Java?
Both gives number of char/elements, length is variable defined in Array class, length( ) is method defined in String class

Object-Orientation Section

What is class?
Class is blue print for objects; Class is collection of objects; Class gives the general structure for objects.

What is object?
Object is an instance of class. Object is real world entity which has state, identity and behavior.

What is encapsulation?
Encapsulation is packing of data with their methods

What is abstraction?
Abstraction is the process of identification of essential features of objects

What is data hiding?
Implementation details of methods are hidden from the user

What is hierarchy?
Ordering of classes (or inheritance)

What is inheritance?
Extending an existing class is called inheritance. Extracting the features of super class

Why inheritance is important?
Reusability is achieved through inheritance

Which types of inheritances are available in Java?
Simple, Hierarchical and Multilevel

Can we achieve multiple inheritances in Java?
With the help of interfaces we can achieve

What is interface?
Interface is collection of final variables and abstract methods
(We need not give final and abstract keywords and By default they are public methods)

What is the difference between class and interface?
Class contains implemented methods, but interface contains non-implemented methods

What is polymorphism?
Multiple (poly) forms (morphs)
Same instance can respond different manners is called polymorphism

Can we achieve run time polymorphism in Java?
Yes, also called as dynamic polymorphism. Possible by dynamic method dispatch

What is dynamic method dispatch?
When you assign object of sub class for the super class instance, similar methods of super class are hidden

What is overloading?
Same method name can be used with different type and number of arguments (in same class)

What is overriding?
Same method name, similar arguments and same number of arguments can be
defined in super class and sub class. Sub class methods override the super class methods.

What is the difference between overloading and overriding?
Overloading related to same class and overriding related sub-super class
Compile time polymorphism achieved through overloading, run time polymorphism achieved through overriding

Keywords section

What is keyword?
Java reserved word which should is not used as variable/class-name (e.g.: break, for, if, while etc)

What is final keyword?
Used before variables for declaring constants
Used before methods for preventing from overriding
Used before class-name for preventing from inheritance

What is static keyword?
Used before variable for defining shared variables
Used before method for defining class-level methods, these methods
can be called without creating objects (e.g.: parseInt method of Integer class)

What is abstract keyword?
Used for creating abstract class, class which doesn't have any instance

What is this keyword?
To call current class variables and current class methods we can use this key word

What is this( )?
Used for calling another constructor of current class

What is super keyword?
To call super class variables and super class methods we can use super key word

What is super( )?
Used for calling of super class constructor and it should be first executable statement in sub class constructor

Packages and Exception handling section:

What is package?
Package is collection of classes and interfaces

How to define package?
By using package keyword before the definition of class

What is CLASSPATH?
It is an environment variable which is used for defining the location of class files

What is jar?
Jar stands for Java archive files, compressed set of class files and can be used in CLASSPATH
What is the meaning of import java.awt.*;?
Import all the classes and interfaces in the java.awt.package. This doesn't imports other packages
defined in java.awt. package.

Is it necessary to give import java.awt.event.*;, when already import java.awt.* given?
Yes, import java.awt.* doesn't imports event package defined in java.awt. package.

What is exception?
Abnormal termination of program is called exception

What is exception handler?
A method which controls abnormal termination of program

What are the keywords used for exception handling?
try, catch, throw, throws and finally

What is the difference between throw and throws keywords?
throw keyword is used for invoking an exception (For raising)
throws keyword lists exception names which can be ignored from the method execution

What is the difference between final and finally?
final is used for defining constants
finally is used for executing code after completion of exception handler

What is catch( ) block?
Exception handler that controls abnormal termination of program

Can we have multiple catch( ) blocks?
Yes, but first sub class exceptions should be handled

What happen is we not handle sub class exception first?
Generates compile time error saying that unreachable code is defined in the program

Multithreading Section

What is thread?
Thread is a part of process, also called as light weight process

What is multithreading?
Simultaneous execution of threads is called multithreading

What is a process?
Program under execution is called process

How to create thread?
By creating instance of Thread class or implementing Runnable interface

Can you name any methods of Thread class?
currentThread( ), setName( ), getName( ), setPriority( ), getPriority( ), join( ), isAlive( )

What is join( ) method of Thread?
Combines two or more threads running process and wait till their execution is completed

What are the states in Thread Life Cycle?
Ready, running, block, wait, dead

Applets, AWT Section

What is an applet?
Applet is a java program which runs via java-enabled browser

Is graphics possible only in applets?
No, stand alone program frames can also display graphics

What is the relation between java and Internet?
With the help of java applets, we can write programming for Internet

Which package is required to write GUI (Graphical User Interface) programs?
Java.awt

Which package is required to write Applets?
Java.applet

What is an event?
A kind of action

What is event handling?
A procedure which gives functionality for the events

How to implement event handling?
By using interfaces like ActionListener, MouseListener

What is the method available in ActionListener Interface?
public void action performed(AcitonEvent e)

How to pass parameters to Applets?
By using PARAM tag.