PROGRAM:

```
import java.io.*;
import java.util.*;
class Student //display() setName() setAge() setMarks()-overloaded
calculateTotal()
{
   String name;
   int age,m1,m2,m3,flag;
   int[] marks;
   static Scanner sc = new Scanner(System.in);
   Student()
   {
      name = "unknown";
      age = 23;
      m1=m2=m3=0;
      flag = 0;
      marks = new int[5];
   }
   Student(String name,int age)
   {
      this.name = name;
      this.age = age;
      m1=m2=m3=0;
      flag = 0;
      marks = new int[5];
   }
   public void display()
   {
      System.out.println('\n' +"Name: "+ name + '\n' + "Age: " + age + '\n' +
"Total: " +  calculateTotal()+'\n');
   }
   public void setName()
   {
      System.out.println("Enter the name: ");
   name = sc.next();
   }
```

```
    public void setAge()
    {
       System.out.println("Enter the age: ");
       age = sc.nextInt();
    }
    public void setMarks(int a,int b,int c)
    {
       flag = 1;
       m1 = a;
       m2 = b;
       m3 = c;
    }
    public void setMarks(int arr[])
    {
       int i=0;
       flag = 2;
       for(int a: arr)
       {
       marks[i]=a;
       i++;
       }
    }
    public int calculateTotal()
    {
       int total=0;
       if(flag==1)
       {
       total = m1 + m2 + m3;
       }
       else
       {
       for(int a:marks)
       total += a;
       }
       return total;
    }
}
```

```
public class ClassAndObject {
   public static void main(String[] args) {
      System.out.println("\n-- CLASSES AND OBJECTS --\n");
      Student h = new Student();
      h.setName();
      h.setAge();
      h.setMarks(88,86,87);
      h.display();
      Student g = new Student("devan",22);
      g.setMarks(new int[]{50,50,50,50,50});
      g.display();
   }
}
```

OUTPUT:

```
-- CLASSES AND OBJECTS --

Enter the name:
maha
Enter the age:
21

Name: maha
Age: 21
Total: 261

Name: devan
Age: 22
Total: 250
```

RESULT:

PROGRAM:

```
import java.io.*;
import java.util.*;

class Teacher
{
	private int id;
	private String name;
	private float sal;
	Scanner in = new Scanner(System.in);

	Teacher(int id,String name)
	{
		this.id = id;
		this.name = name;
	}

	Teacher(int id,String name,float sal)
	{
		this.id = id;
		this.name = name;
		this.sal = sal;
	}

	public int getId(){
		return id;
	}

	public String getName(){
		return name;
	}

	public float getSal(){
		return sal;
	}
```

```
	public int getNoOfBookCanTake()
	{
		return 3;
	}

}

interface courses
{
	public String[] getCourses();
}

interface placement
{
	public String[] getAttendedCompanies();
}

class MCAstudent extends Teacher implements courses,placement
{
	int marks;

	MCAstudent(int id,String name,int marks)
	{
		super(id,name);
		this.marks = marks;
	}

	void setMarks(int marks)
	{
		this.marks = marks;
	}

	int getMarks()
	{
		return marks;
	}
```

```
	public String[] getCourses()
	{
		String[] courses={"OPERATING SYSTEM","C PROGRAMMING"};
		return courses;
	}

	public String[] getAttendedCompanies()
	{
		String[] atndComp={"TCS","ZOHO"};
		return atndComp;
	}

	public int getNoOfBookCanTake()
	{
		return 2;
	}
}


public class Inheritpoly {

	public static void main(String[] args)throws IOException {
		  System.out.println("\n-- Inheritance and Interface --\n");
		int id,marks;
		String name;
		// float sal;
		Scanner sc = new Scanner(System.in);

		System.out.println("Enter the id: ");
		id = sc.nextInt();

		sc.nextLine();

		System.out.println("Enter the name: ");
		name = sc.nextLine();

		System.out.println("Enter the marks: ");
```

```
                marks = sc.nextInt();

                MCAstudent t1 = new MCAstudent(id,name,marks);

                System.out.printf("\nID : %d\nName : %s\nMarks :
%d\n",t1.getId(),t1.getName(),t1.getMarks());
                courses c = t1;
                System.out.print("Courses : ");
                System.out.println(Arrays.toString(c.getCourses()));
                placement p = t1;
                System.out.print("Attended Companies : ");
                System.out.println(Arrays.toString(p.getAttendedCompanies()));
                System.out.println("The no. of books can take in library:
"+t1.getNoOfBookCanTake());

        }
}
```

OUTPUT:

```
-- Inheritance and Interface --

Enter the id:
101
Enter the name:
maha
Enter the marks:
466

ID : 101
Name : maha
Marks : 466
Courses : [OPERATING SYSTEM, C PROGRAMMING]
Attended Companies : [TCS, ZOHO]
The no. of books can take in library: 2
```

RESULT:

PROGRAM:

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.Iterator;
//example of java synchronized method
class Table{

   static int maxval=Integer.MIN_VALUE,resval=0;
   public synchronized void printTable(int a,int b){  //synchronized method
      int cnt,max=Integer.MIN_VALUE,result=0;

      for(int i=a;i<=b;i++)
      {
         cnt=0;
         for(int j=2;j<i;j++)
         {
            if(i%j==0)
            {
               cnt++;
            }
         }
         if(cnt>max)
         {
            result = i;
            max = cnt;
         }
      }
      if(maxval<=max)
      {
         maxval=max;
         resval=result;
      }
      System.out.printf("The number that has maximum number of divisors from
%d to %d is : %d",a,b,result);
      System.out.println();
```

```
        System.out.println("Count = "+max);

    }
    public int[] getFinalResult()
    {
        return new int[]{maxval,resval};
    }

}

class MyThread extends Thread{
    Table t;
    int a,b;
    MyThread(Table t,int a,int b){
        this.t=t;
        this.a=a;
        this.b=b;
    }
    public void run(){
        t.printTable(a,b);
    }
}


public class TestSynchronization2{
    public static void main(String args[]) throws Exception{
        Scanner sc = new Scanner(System.in);
        int value, kvalue;
        int[] result = new int[2];
        Table obj = new Table();//only one object
        System.out.println("Enter the value: ");
        value = sc.nextInt();

        MyThread[] t = new MyThread[10];
        //1000
        kvalue = value/10; //100
        int j=1,k=kvalue;
```

```
        for(int i=0;i<10;i++)
        {
           t[i] = new MyThread(obj,j,k);
           t[i].start();
           j+=kvalue;//1 101
           k+=kvalue;//100 200
        }

        for(int l=0;l<10;l++)
         t[l].join();

        result = obj.getFinalResult();
        System.out.printf("Result: %d Count: %d \n",result[1],result[0]);
    }
}
```

OUTPUT:

```
-- Multithreading --

Enter the value:
1000
The number that has maximum number of divisors from 1 to 100 is : 60
Count = 10
The number that has maximum number of divisors from 901 to 1000 is : 960
Count = 26
The number that has maximum number of divisors from 801 to 900 is : 840
Count = 30
The number that has maximum number of divisors from 701 to 800 is : 720
Count = 28
The number that has maximum number of divisors from 601 to 700 is : 630
Count = 22
The number that has maximum number of divisors from 401 to 500 is : 420
Count = 22
The number that has maximum number of divisors from 501 to 600 is : 504
Count = 22
The number that has maximum number of divisors from 301 to 400 is : 360
Count = 22
The number that has maximum number of divisors from 201 to 300 is : 240
Count = 18
The number that has maximum number of divisors from 101 to 200 is : 180
Count = 16
Result: 840 Count: 30
```

RESULT:

PROGRAM:

```
import java.io.File;
import java.io.FileReader;
import java.io.FileNotFoundException;
import java.io.IOException;


public class ExceptionDemo {

   static void arithmeticException(int a,int b)
   {
    try{
    System.out.println("Result = "+(a/b));
    }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }
   }

   static void nullPointerException(String str)
   {
    System.out.println("Length = "+str.length());
   }

   static void fileNotFoundException()
   {
    try{
      File file = new File("E://file.txt");
      FileReader fr = new FileReader(file);
    }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }
   }

   static void numberFormatException(String str)
   {
    try{
```

```
    int num = Integer.parseInt(str);
    System.out.println("Integer = "+(num-5));
    }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }
  }

  static char indexOutOfBounds(int pos)
  {
    String str = "mahadevan";
    return str.charAt(pos);
  }

  static void userDefined(int num)throws GreaterThanTenException
  {
    if(num>10)
    {
      throw new GreaterThanTenException("Greater than 10 exception");
    }
  }

  public static void main(String[] args) {
   System.out.println("\n-- Exceptions --\n");

   try{
    System.out.println("passing null to a function that requires a string causes
NullPointerException");
    ExceptionDemo.nullPointerException(null);
   }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }
    System.out.println();

    System.out.println("Opening a file might cause FileNotFoundException if the
file is not there");
    ExceptionDemo.fileNotFoundException();
    System.out.println();
```

```
    System.out.println("Dividing any value by 0 causes ArithmeticException");
    ExceptionDemo.arithmeticException(1,0);
    System.out.println();

    System.out.println("Passing a character string instead of number string
causes this error");
    ExceptionDemo.numberFormatException("hel");
    System.out.println();

    System.out.println("passing index above 8 causes this error");
    char ch=' ';
    try{
    ch = ExceptionDemo.indexOutOfBounds(13);
    System.out.println("Char ch = "+ch);
    }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }
    System.out.println();



    try{
      ExceptionDemo.userDefined(22);
    }catch(Exception e){
      System.out.println("Exception message: "+e.toString());
    }

  }
}

class GreaterThanTenException extends Exception
{
 GreaterThanTenException(){}
 GreaterThanTenException(String msg){
  super(msg);
 }
}
```

OUTPUT:

```
-- Exceptions --

passing null to a function that requires a string causes NullPointerException
Exception message: java.lang.NullPointerException

Opening a file might cause FileNotFoundException if the file is not there
Exception message: java.io.FileNotFoundException: E:/file.txt (No such file or directory)

Dividing any value by 0 causes ArithmeticException
Exception message: java.lang.ArithmeticException: / by zero

Passing a character string instead of number string causes this error
Exception message: java.lang.NumberFormatException: For input string: "hel"

passing index above 8 causes this error
Exception message: java.lang.StringIndexOutOfBoundsException: String index out of range: 13

Exception message: GreaterThanTenException: Greater than 10 exception
```

RESULT:

PROGRAM:

PackageDemo.java

```
import addition.PackageAccess;
public class PackageDemo{
  public static void main(String args[]){
      PackageAccess obj = new PackageAccess();
      System.out.println(obj.add(100, 200));
  }
}
```

PackageAccess.java

```
import addition.PackageAccess;

public class PackageDemo{
  public static void main(String args[]){
      PackageAccess obj = new PackageAccess();
      System.out.println("Using Package to access the function add(),
Result:"+obj.add(100, 200));
  }
}
```

OUTPUT:

```
Using Package to access the function add(), Result:300
```

RESULT:

PROGRAM:

```
import java.io.*;
import java.util.Scanner;

class StreamDemo
{
	public static void main(String[] args)throws IOException
	{
		String yourFile = "input.txt";

		// Scanner sc = new Scanner(System.in);
		// String yourContent=sc.nextLine();
		String yourContent="Genius Ganesh";
		File tmpDir = new File(yourFile);


		if(tmpDir.exists()){
			FileOutputStream fos = new FileOutputStream(yourFile);

			fos.write(yourContent.getBytes());

			fos.flush();

			fos.close();
			FileInputStream fis = new FileInputStream(yourFile);
			int data;
			 int count =0;
			while((data=fis.read()) != -1)
			{
				System.out.print((char)data);count++;
			}
			System.out.println(count);
		}
	}
}
```

OUTPUT:

```
FileInputStream & FileOutputStream

Writing "Hello world" to a file....

Reading that content from that file...

Hello world
11
```

RESULT:

PROGRAM:

```
import java.sql.*;
import java.util.Scanner;
class OracleCon{

   static Scanner sc;
   private static Connection con=null;
   private static Statement stmt=null;
   static{
      sc = new Scanner(System.in);
   }


   public static void main(String args[]){
      try{
         //step1 load the driver class
         Class.forName("oracle.jdbc.driver.OracleDriver");

         //step2 create  the connection object
         con = DriverManager.getConnection(
         "jdbc:oracle:thin:@localhost:1521:xe","system","thepassword");

         //step3 create the statement object
         stmt=con.createStatement();

         String createSql = "create table emp(id number(10),name
varchar2(40),age number(3))";
         int j = stmt.executeUpdate(createSql);

         if(j == 0)
         {
            System.out.println("Table is created");
         }
         else
         {
            System.out.println("Table is not created");
```

```
        }

        System.out.println("Enter the no. of records you want to enter:");
        int rec = sc.nextInt();
        sc.nextLine();
        String name;
        int age,id,res;
        for(int i=0;i<rec;i++)
        {
           System.out.println("Enter the name:");
           name = sc.nextLine();
           System.out.println("Enter the id:");
           id = sc.nextInt();
           System.out.println("Enter the age:");
           age = sc.nextInt();
           // System.out.println("INSERT INTO EMP
VALUES("+id+","+"\'"+name+"\'"+","+age+")");
           sc.nextLine();
           res = stmt.executeUpdate("INSERT INTO EMP
VALUES("+id+","+"\'"+name+"\'"+","+age+")");
           if(res != 0)
           {
              System.out.println("Row is created");
           }
           else
           {
              System.out.println("Row is not created");
           }
        }

         String sql = "UPDATE EMP SET NAME='Hari' " +
              "WHERE id=201";

        //Step 4 : Executing The Query

        //We are using executeUpdate() method as we are executing UPDATE
statement
```

```
   int i = stmt.executeUpdate(sql);

   if(i != 0)
   {
      System.out.println("Record is updated");
   }
   else
   {
      System.out.println("Record is not updated");
   }


   ResultSet rs=stmt.executeQuery("select * from emp");
   while(rs.next())
   System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));


}
catch (Exception e)
{
   e.printStackTrace();
}
finally
{
   //STEP 5 : Closing The DB Resources

   //Closing the Statement object
   try
   {
      if(stmt!=null)
      {
         stmt.close();
         stmt=null;
      }
   }
   catch (SQLException e)
```

```
        {
            e.printStackTrace();
        }

        //Closing the Connection object

        try
        {
            if(con!=null)
            {
                con.close();
                con=null;
            }
        }
        catch (SQLException e)
        {
            e.printStackTrace();
        }
    }

  }
}
```

OUTPUT:

```
Java Program connecting to Oracle Database

Table is created
Enter the no. of records you want to enter:
2
Enter the name:
maha
Enter the id:
201
Enter the age:
22
Row is created
Enter the name:
siva
Enter the id:
202
Enter the age:
23
Row is created
Record is updated
201  Hari  22
202  siva  23
```

RESULT:

PROGRAM:

SocketDemo.java

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;

public class SocketDemo
{
   private Socket socket = null;
   private DataInputStream input = null;
   private DataOutputStream output = null;

   public SocketDemo(String address, Integer port){

      try{
         socket = new Socket(address,port);
         input = new DataInputStream(System.in);
         output = new DataOutputStream(socket.getOutputStream());
      }catch(Exception e){
         e.printStackTrace();
      }

      String line="";
      while(!(line.equals("Done"))){
         try{
            line = input.readLine();
            output.writeUTF(line);
         }catch(Exception e){
            e.printStackTrace();
         }
      }

      try{
```

```
            input.close();
            output.close();
            socket.close();
        }catch(Exception e){
            e.printStackTrace();
        }

    }
    public static void main(String[] args){
        SocketDemo client = new SocketDemo("127.0.0.1",5000);
    }
}
```

SocketDemoServer.java

```
import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class SocketDemoServer{
    private Socket socket = null;
    private ServerSocket server = null;
    private DataInputStream in = null;

    public SocketDemoServer(int port){
        try{
            server = new ServerSocket(port);
            System.out.println("Server started::");

            System.out.println("Waiting for a client .....");
            socket = server.accept();
            System.out.println("Client accepted.");

            in = new DataInputStream(new
BufferedInputStream(socket.getInputStream()));
```

```
        String line="";

        while(!line.equals("Done")){
           try{
              line = in.readUTF();
              System.out.println(line);
           }catch(Exception i){
              i.printStackTrace();
           }
        }

        System.out.println("Closing connection");
        socket.close();
        in.close();

     }catch(Exception i){
        i.printStackTrace();
     }
  }
  public static void main(String args[]){
     SocketDemoServer server = new SocketDemoServer(5000);
  }
}
```

OUTPUT:

```
client output:

Hi there
How are you?
Done

server output:

Server started::
Waiting for a client ......
Client accepted.
Hi there
How are you?
Done
Closing connection
```

RESULT:

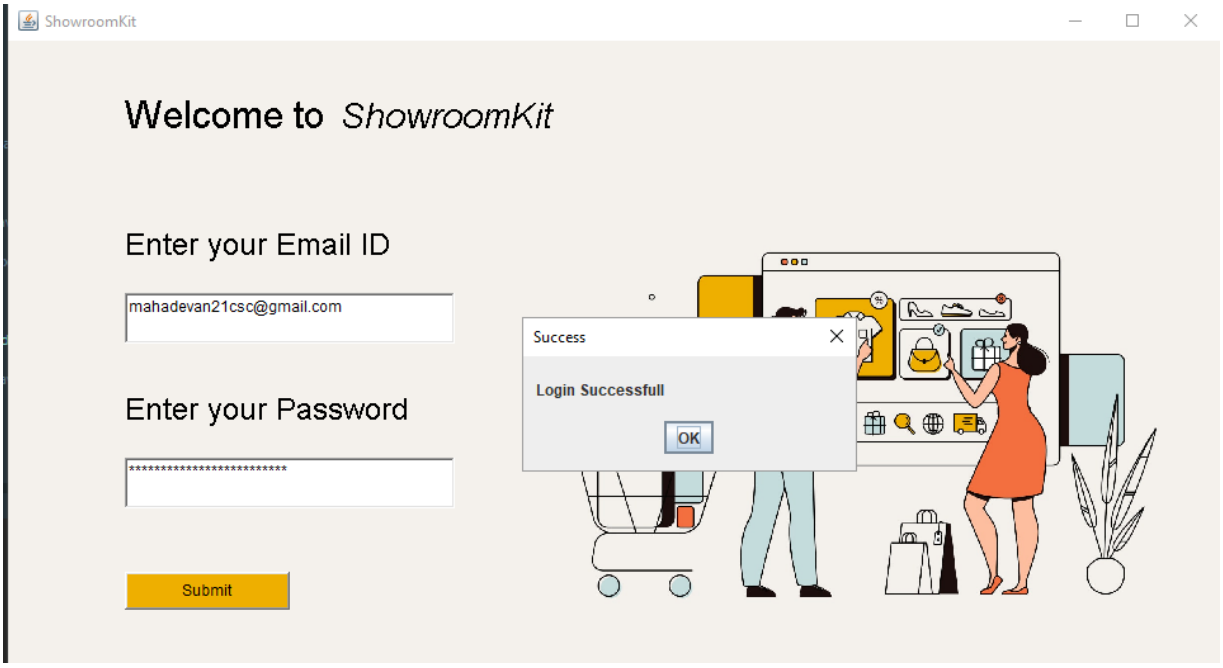
PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;

class AWTForm extends Frame implements ActionListener
{
	TextField tf1,tf2;
	Button b1;
	AWTForm()
	{
		this.addWindowListener(new WindowAdapter(){
			public void windowClosing(WindowEvent e)
			{
				System.exit(0);
			}
		});
		tf1 = new TextField(25);
		tf1.setBounds(100,230,260,40);
		tf2 = new TextField(25);
		tf2.setBounds(100,360,260,40);
		tf2.setEchoChar('*');

		b1 = new Button("Submit");
		b1.setBounds(100,450,130,30);
		b1.setBackground(new Color(238,175,0));
		b1.addActionListener(this);
		this.setTitle("ShowroomKit");
		this.setSize(1300,600);
		this.setVisible(true);
		this.setLayout(null);
		this.add(tf1);
		this.add(tf2);
		this.add(b1);
	}
```

```
	public void actionPerformed(ActionEvent ae)
	{
		if(ae.getSource()==b1)
		{
			if(tf1.getText().isEmpty())
			{
				JOptionPane.showMessageDialog(null, "Enter the
email ID","Error",JOptionPane.QUESTION_MESSAGE);
				return;
			}
			if(tf2.getText().isEmpty())
			{
				JOptionPane.showMessageDialog(null, "Enter the
password","Error",JOptionPane.QUESTION_MESSAGE);
				return;
			}
			JOptionPane.showMessageDialog(null, "Login
Successfull","Success",JOptionPane.PLAIN_MESSAGE);
		}
	}

	public void paint(Graphics g)
	{
		Image img = Toolkit.getDefaultToolkit().getImage("shopping.jpg");
		MediaTracker track = new MediaTracker(this);
		track.addImage(img,0);

		try{
			track.waitForID(0);
		}catch(InterruptedException ie){}

		this.setBackground(new Color(244,241,236));
		Font f = new Font("Arial",Font.PLAIN,30);
		g.setFont(f);
		g.setColor(Color.black);
		g.drawString("Welcome to",100,100);
```

```
            Font f2 = new Font("Arial",Font.ITALIC,28);
            g.setFont(f2);
            g.drawString("ShowroomKit",270,100);

            g.drawImage(img,440,130,500,400,null);

            Font f3 = new Font("Arial",Font.PLAIN,24);
            g.setFont(f3);
            g.drawString("Enter your Email ID",100,200);
            g.drawString("Enter your Password",100,330);


        }

        public static void main(String args[])
        {
            AWTForm f = new AWTForm();

        }
}
```

OUTPUT:



RESULT: