

**S.Y.B.Sc**

**Computer Science**

**2019 – 2020**

**This is to certify that PRASHANT SINGH of S.Y.B.Sc Roll No. 70 has successfully completed the practical of Paper – II (sem – III)** **Core Java during the Academic Year 2019-2020 as specified by the MUMBAI UNIVERSITY.**

**Prof. (Ms.) Saba Shaikh Prof. (Mrs.) Anita Gaikwad**

**Professor-In-charge In-charge**

**Dept. of Computer Science Dept. of Computer Science**

Name : PRASHANT SINGH. Subject : CORE JAVA(PAPER-2)

Class: SYBSC(COMPUTER SCIENCE) Exam Seat no: 070

Academic Year : 2019-2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **INDEX** |  |  |
| **No** | **Date** | **TITLE** | **Page**  **No** | **SIGN** |
| **1.** |  | Write a program to create a class and object in java. |  |  |
| **2.** |  | Demonstrate Java inheritance using extends keyword. |  |  |
| **3.** |  | Demonstrate polymorphism in Java.  a. Bank Example  b. Shape Example |  |  |
| **4.** |  | Demonstrate concept of method overloading in Java. |  |  |
| **5** |  | Demonstrate String manipulation in java. |  |  |
| **6.** |  | Demonstrate the concept of multithreading in Java. |  |  |
| **7.** |  | Demonstrate creating your own exception in Java. |  |  |
| **8.** |  | Create a package: Animals. In package animals create interface Animal with suitable behaviors.  Implement the interface Animal in the same package animals. |  |  |
| **9.** |  | Demonstrate the concept of networking (client-server communication) in Java. |  |  |
| **10.** |  | Using various swing components design Java application to accept a student's resume. (Design form). |  |  |
| **11.** |  | Design a form to accept a number and display result in label whether it is even or odd. |  |  |
| **12.** |  | Write a Java List example and demonstrate methods of Java List interface. |  |  |

**PRACTICAL-1**

**Aim** : Write a program to create a class and object in java.

**Source Code-**

**TestEmployee.java**

class Employee

{

String name;

int age;

String designation;

double salary;

public Employee(String name)

{

name = name;

}

public void empAge(int empAge)

{

age = empAge;

}

public void empDesignation(String empDesig)

{

designation = empDesig;

}

public void empSalary(double empSalary)

{

salary = empSalary;

}

public void printEmployee()

{

System.out.println("Name:"+ name );

System.out.println("Age:" + age );

System.out.println("Designation:" + designation );

System.out.println("Salary:" + salary);

}

}

public class TestEmployee

{

public static void main(String args[])

{

Employee empOne = new Employee("James Smith");

Employee empTwo = new Employee("Mary Anne");

// Invoking methods for each object created

empOne.empAge(26);

empOne.empDesignation("Senior Software Engineer");

empOne.empSalary(1000);

empOne.printEmployee();

empTwo.empAge(21);

empTwo.empDesignation("Software Engineer");

empTwo.empSalary(500);

empTwo.printEmployee();

}

}

**Output –**

Name:null

Age:26

Designation:Senior Software Engineer

Salary:1000.0

Name:null

Age:21

Designation:Software Engineer

Salary:500.0

**PRACTICAL-2**

**Aim :** Demonstrate Java inheritance using extends keyword.

**Source Code –**

**CalcDemo.java**

class Calculation {

int z;

public void addition(int x, int y) {

z = x + y;

System.out.println("The sum of the given numbers:"+z);

}

public void Subtraction(int x, int y) {

z = x - y;

System.out.println("The difference between the given numbers:"+z);

}

}

class MyCalculation extends Calculation

{

public void multiplication(int x, int y)

{

z = x \* y;

System.out.println("The product of the given numbers:"+z);

}

}

public class CalcDemo

{

public static void main(String args[]) {

int a = 20, b = 10;

MyCalculation demo = new MyCalculation();

demo.addition(a, b);

demo.Subtraction(a, b);

demo.multiplication(a, b);

}

}

**Output –**

The sum of the given numbers:30

The difference between the given numbers:10

The product of the given numbers:200

**TestInheritance.java**

class Person

{

int id;

String name;

Person(int id,String name)

{

this.id=id;

this.name=name;

}

}

class Emp extends Person

{

float salary;

Emp(int id,String name,float salary)

{

super(id,name);//reusing parent constructor

this.salary=salary;

}

void display()

{

System.out.println(id+" "+name+" "+salary);

}

}

class TestInheritance

{

public static void main(String[] args)

{

Emp e1=new Emp(1,"ankit",45000f);

e1.display();

}

}

**Output -**

1 ankit 45000.0

**PRACTICAL-3**

**Aim:** Demonstrate polymorphism in Java.

**a**. Bank Example.

**b**. Shape Example.

**Source Code-**

1. **Bank Example**

**TestBank.java**

class Bank

{

int getRateOfInterest(){return 0;}

}

class SBI extends Bank

{

int getRateOfInterest(){return 8;}

}

class ICICI extends Bank

{

int getRateOfInterest(){return 7;}

}

class AXIS extends Bank

{

int getRateOfInterest(){return 9;}

}

class TestBank

{

public static void main(String args[])

{

SBI s=new SBI();

ICICI i=new ICICI();

AXIS a=new AXIS();

System.out.println("SBI Rate of Interest: "+s.getRateOfInterest());

System.out.println("ICICI Rate of Interest: "+i.getRateOfInterest());

System.out.println("AXIS Rate of Interest: "+a.getRateOfInterest());

}

}

**Output:**

SBI Rate of Interest: 8

ICICI Rate of Interest: 7

AXIS Rate of Interest: 9

**TestShape1(Abstract)**

abstract class Shape

{

public abstract double area();

public abstract double perimeter();

}

class Rectangle extends Shape

{

private final double width, length;

public Rectangle()

{

this(1,1);

}

public Rectangle(double width, double length)

{

this.width = width;

this.length = length;

}

public double area()

{

return width \* length;

}

public double perimeter()

{

return 2 \* (width + length);

}

}

class Circle extends Shape

{

private final double radius;

final double pi = Math.PI;

public Circle()

{

this(1);

}

public Circle(double radius)

{

this.radius = radius;

}

public double area()

{

return pi \* Math.pow(radius, 2);

}

public double perimeter()

{

return 2 \* pi \* radius;

}

}

public class TestShape1

{

public static void main(String[] args)

{

// Rectangle test

double width = 5, length = 7;

Shape rectangle = new Rectangle(width, length);

System.out.println("Rectangle width: " + width + " and length: " + length);

System.out.println("Resulting area: " + rectangle.area());

System.out.println("Resulting perimeter: " + rectangle.perimeter() + "\n");

// Circle test

double radius = 5;

Shape circle = new Circle(radius);

System.out.println("Circle radius: " + radius);

System.out.println("Resulting Area: " + circle.area());

System.out.println("Resulting Perimeter: " + circle.perimeter() + "\n");

}

}

**Output:**

Rectangle width: 5.0 and length: 7.0

Resulting area: 35.0

Resulting perimeter: 24.0

Circle radius: 5.0

Resulting Area: 78.53981633974483

Resulting Perimeter: 31.41592653589793

**TestShape2 (Interface)**

interface Shape

{

public double area();

public double perimeter();

}

class Rectangle implements Shape

{

private final double width, length;

public Rectangle()

{

this(1,1);

}

public Rectangle(double width, double length)

{

this.width = width;

this.length = length;

}

public double area()

{

return width \* length;

}

public double perimeter()

{

return 2 \* (width + length);

}

}

class Circle implements Shape

{

private final double radius;

final double pi = Math.PI;

public Circle()

{

this(1);

}

public Circle(double radius)

{

this.radius = radius;

}

public double area()

{

return pi \* Math.pow(radius, 2);

}

public double perimeter()

{

return 2 \* pi \* radius;

}

}

class TestShape2

{

public static void main(String[] args)

{

// Rectangle test

double width = 5, length = 7;

Shape rectangle = new Rectangle(width, length);

System.out.println("Rectangle width: " + width + " and length: " + length);

System.out.println("Resulting area: " + rectangle.area());

System.out.println("Resulting perimeter: " + rectangle.perimeter() + "\n");

// Circle test

double radius = 5;

Shape circle = new Circle(radius);

System.out.println("Circle radius: " + radius);

System.out.println("Resulting Area: " + circle.area());

System.out.println("Resulting Perimeter: " + circle.perimeter() + "\n");

}

}

**Output:**

Rectangle width: 5.0 and length: 7.0

Resulting area: 35.0

Resulting perimeter: 24.0

Circle radius: 5.0

Resulting Area: 78.53981633974483

Resulting Perimeter: 31.41592653589793

**PRACTICAL-4**

**Aim:** Demonstrate concept of method overloading in Java.

**Source Code-**

**TestOverloading.java**

class Adder

{

static int add(int a,int b)

{

return a+b;

}

static int add(int a,int b,int c)

{

return a+b+c;

}

}

class TestOverloading

{

public static void main(String args[])

{

System.out.println("Addition of two number="+Adder.add(12,16));

System.out.println("Addition of three number="+Adder.add(15,16,17));

}

}

**Output:**

Addition of two number=28

Addition of three number=48

**PRACTICAL- 5**

**Aim: Demonstrate String manipulation in java**.

**Source Code :**

**StringManipulation.java**

public class StringManupulation

public static void main(String args[])

{

String s="Sachin";

System.out. println(s.toUpperCase());//SACHIN

System.out. println(s.toLowerCase ());//sachin

System.out. println(s);//Sachin(no change in original)

System.out.println(s.startsWith("Sa"));

System.out.println(s.endsWith("n");

System.out.printIn(s.charAt(0));//s

System.out.println(s.charAt(3));//h

System.out.println(s.length())//6

int a=10;

String s1 = String.valueOf(a);

System.out.println(s1+10);

String s2="Java is a programming language. Java is a platform. Java is an Island.";

String replaceString-s2.replace("Java","Kava");

System.out.println(replaceString);

}

**Output :**

SACHIN

sachin

Sachin

true

true

S

h

6

1010

Kava is a programming language. Kava is a platform. Kava is an Island.

**PRACTICAL-6**

**Aim:** Demonstrate the concept of multithreading in Java.

**Source Code-**

**TestSynchronization.java**

class Table

{

synchronized void printTable(int n)

{

//synchronized method

for(int i=1;i<=5;i++)

{

System.out.println(n\*i);

try

{

Thread.sleep(400);

}

catch(Exception e)

{

System.out.println(e);}

}

}

}

class MyThread1 extends Thread

{

Table t;

MyThread1(Table t)

{

this.t=t;

}

public void run()

{

t.printTable(5);

}

}

class MyThread2 extends Thread

{

Table t;

MyThread2(Table t)

{

this.t=t;

}

public void run()

{

t.printTable(100);

}

}

class TestSynchronization

{

public static void main(String args[])

{

Table obj = new Table();//only one object

MyThread1 t1=new MyThread1(obj);

MyThread2 t2=new MyThread2(obj);

t1.start();

t2.start();

}

}

**Output:** 5

10

15

20

25

100

200

300

400

500

**PRACTICAL-7**

**Aim:** Demonstrate creating your own exception in Java.

**Source Code**-

class TestCustomException

{

static void validate(int age)throws InvalidAgeException

{

if(age<18)

throw new InvalidAgeException("not valid");

else

System.out.println("welcome to vote");

}

public static void main(String args[])

{

Try

{

validate(19);

}

catch(Exception m)

{

System.out.println("Exception occured: "+m);

}

System.out.println("rest of the code...");

}

}

**Output:**

welcome to vote

rest of the code...

**PRACTICAL-8**

**Aim:** Create a package: Animals. In package animals create interface Animal with suitable

behaviors.Implement the interface Animal in the same package animals.

**Source Code-**

**Animal.java**

package animals;

interface Animal

{

public void eat();

public void travel();

}

**/\***

**javac -d . Animal.java**

**javac -d . MammalInt.java**

**java animals.MammalInt**

**\*/**

**MammalInt.java**

package animals;

public class MammalInt implements Animal

{

public void eat()

{

System.out.println("Mammal eats");

}

public void travel()

{

System.out.println("Mammal travels");

}

public int noOfLegs()

{

return 0;

}

public static void main(String args[])

{

MammalInt m = new MammalInt();

m.eat();

m.travel();

}

}

**Output:**

Mammal eats

Mammal travels

**PRACTICAL-9**

**Aim:** Demonstrate the concept of networking (client-server communication) in Java.

**Source Code-**

**MyServer.java**

import java.io.\*;

import java.net.\*;

public class MyServer

{

public static void main(String[] args)

{

try

{

ServerSocket ss=new ServerSocket(6666);

Socket s=ss.accept();//establishes connection

DataInputStream dis=new DataInputStream(s.getInputStream());

String str=(String)dis.readUTF();

System.out.println("message= "+str);

ss.close();

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**Note:**

**1) First compile MyServer.java file then compile MyClient.java file .**

**2) Then first interpret MyClient.class file then interpret MyServer.class file.**

**MyClient.java**

import java.io.\*;

import java.net.\*;

class MyClient

{

public static void main(String[] args)

{

try

{

Socket s=new Socket("localhost",6666);

DataOutputStream dout=new DataOutputStream(s.getOutputStream());

dout.writeUTF("Hello Server");

dout.flush();

dout.close();

s.close();

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**Output:**

message= Hello Server

**PRACTICAL-10**

**Aim:** Using various swing components design Java application to accept a student's resume.

(Design form).

**Source Code-**

**StudentResume.java**

import java.applet.Applet;

import java.awt.\*;

import java.awt.Font;

class StudentResume extends Frame

{

Button btnSubmit;

Label lblName,lblAge,lblAddr,lblGender,lblQua;

TextField txtName,txtAge;

TextArea txtAddr;

CheckboxGroup ChkGrp;

Checkbox chkMale,chkFemale;

Checkbox chkMca,chkBca,chkBba,chkMba;

StudentResume()

{

setLayout(new FlowLayout());

**//setLayout(new GridLayout(3,2));**

lblName = new Label("Name: ");

lblAge = new Label("Age: ");

lblAddr = new Label("Address : ");

lblGender = new Label("Gender: ");

lblQua = new Label("Qualification: ");

txtName = new TextField(20);

txtAge = new TextField(20);

txtAddr = new TextArea();

ChkGrp = new CheckboxGroup();

chkMale = new Checkbox("Male",ChkGrp,false);

chkFemale = new Checkbox("Female",ChkGrp,false);

chkMca = new Checkbox("MCA");

chkBca = new Checkbox("BCA");

chkMba = new Checkbox("MBA");

chkBba = new Checkbox("BBA");

btnSubmit = new Button("SUBMIT");

add(lblName);

add(txtName);

add(lblAge);

add(txtAge);

add(lblAddr);

add(txtAddr);

add(lblGender);

add(chkMale);

add(chkFemale);

add(lblQua);

add(chkBca);

add(chkBba);

add(chkMca);

add(chkMba);

add(btnSubmit);

setSize(500,500);

setVisible(true);

}

public static void main(String args[])

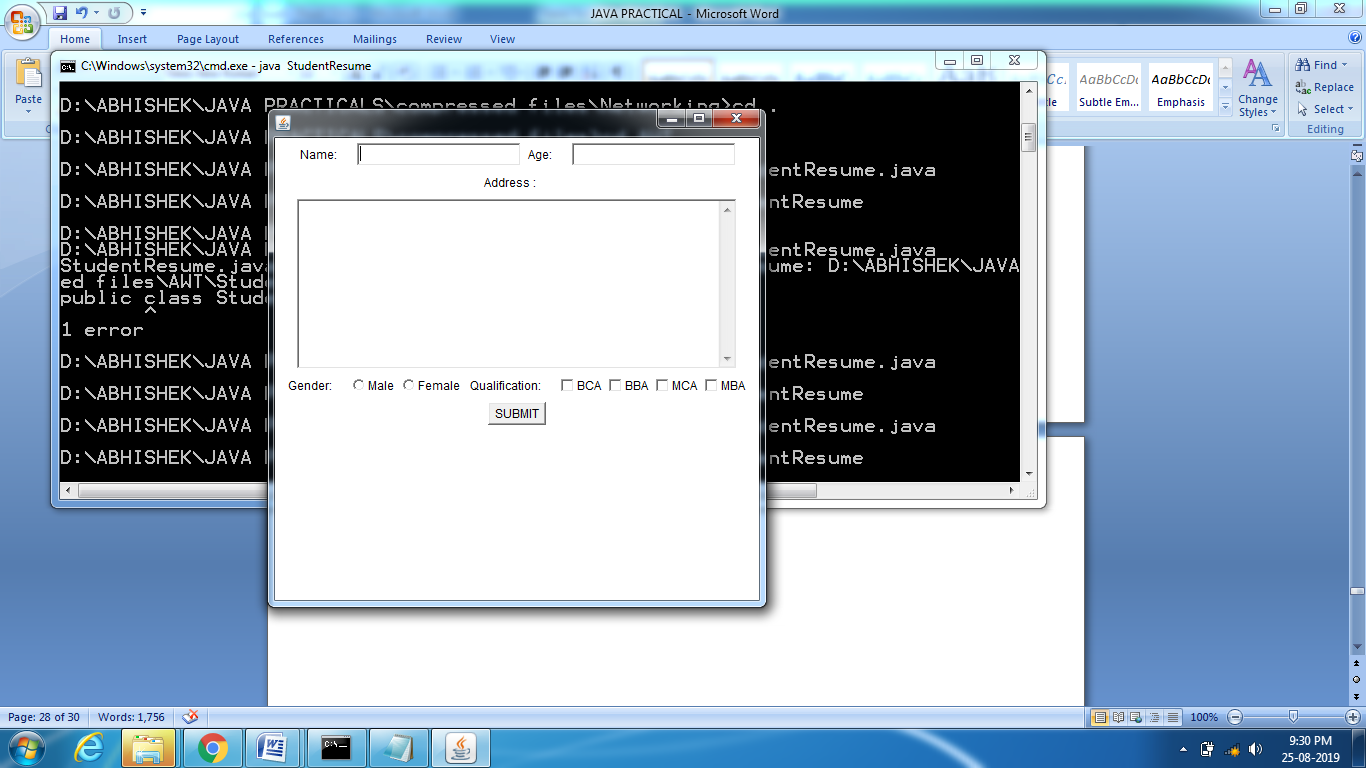
{

new StudentResume();

}

}

**Output:**



**PRACTICAL-11**

**Aim:** Design a form to accept a number and display result in label whether it is even or odd.

**Source Code-**

**EvenOdd.java**

import java.awt.\*;

import java.awt.event.\*;

class EvenOdd extends Frame implements ActionListener

{

Label lblnum,lblResult;

TextField txtnum;

Button btnDisplay;

EvenOdd()

{

setLayout(new FlowLayout());

lblnum=new Label("Enter number");

lblResult=new Label("");

txtnum=new TextField();

btnDisplay=new Button("Even/Odd");

add(lblnum);

add(txtnum);

add(btnDisplay);

add(lblResult);

btnDisplay.addActionListener(this);

setSize(300,300);

setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

int num=Integer.parseInt(txtnum.getText());

if(e.getSource()==btnDisplay)

{

if(num%2==0)

lblResult.setText("The number is even");

else

lblResult.setText("The number is odd");

}

}

public static void main(String args[])

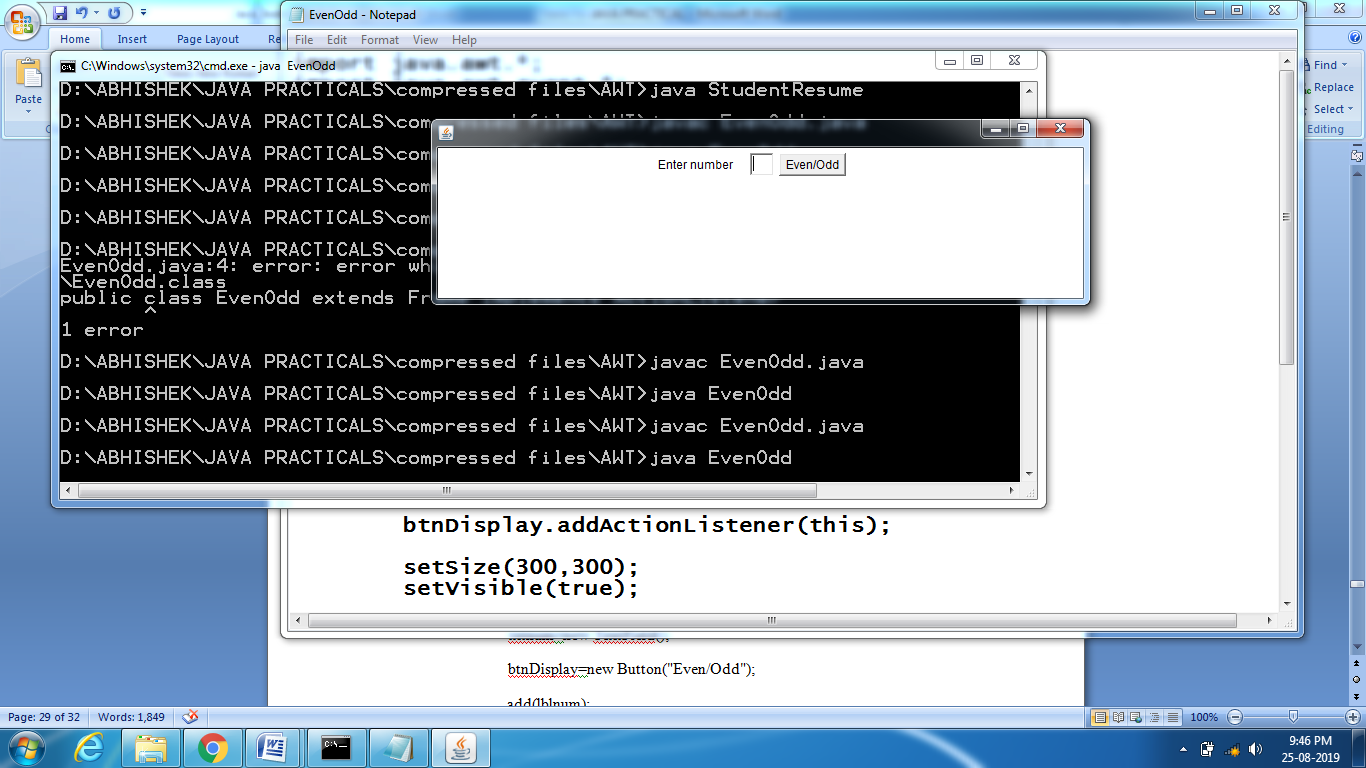
{

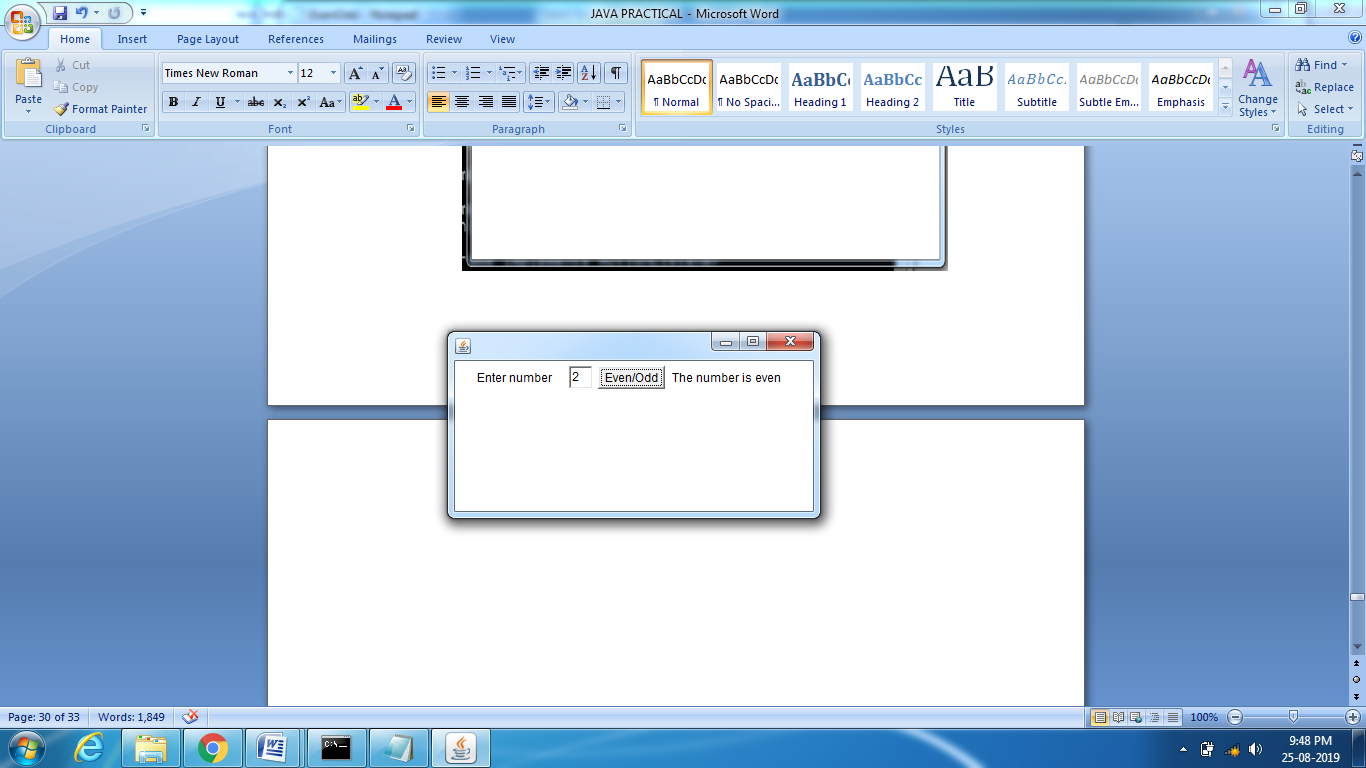
new EvenOdd();

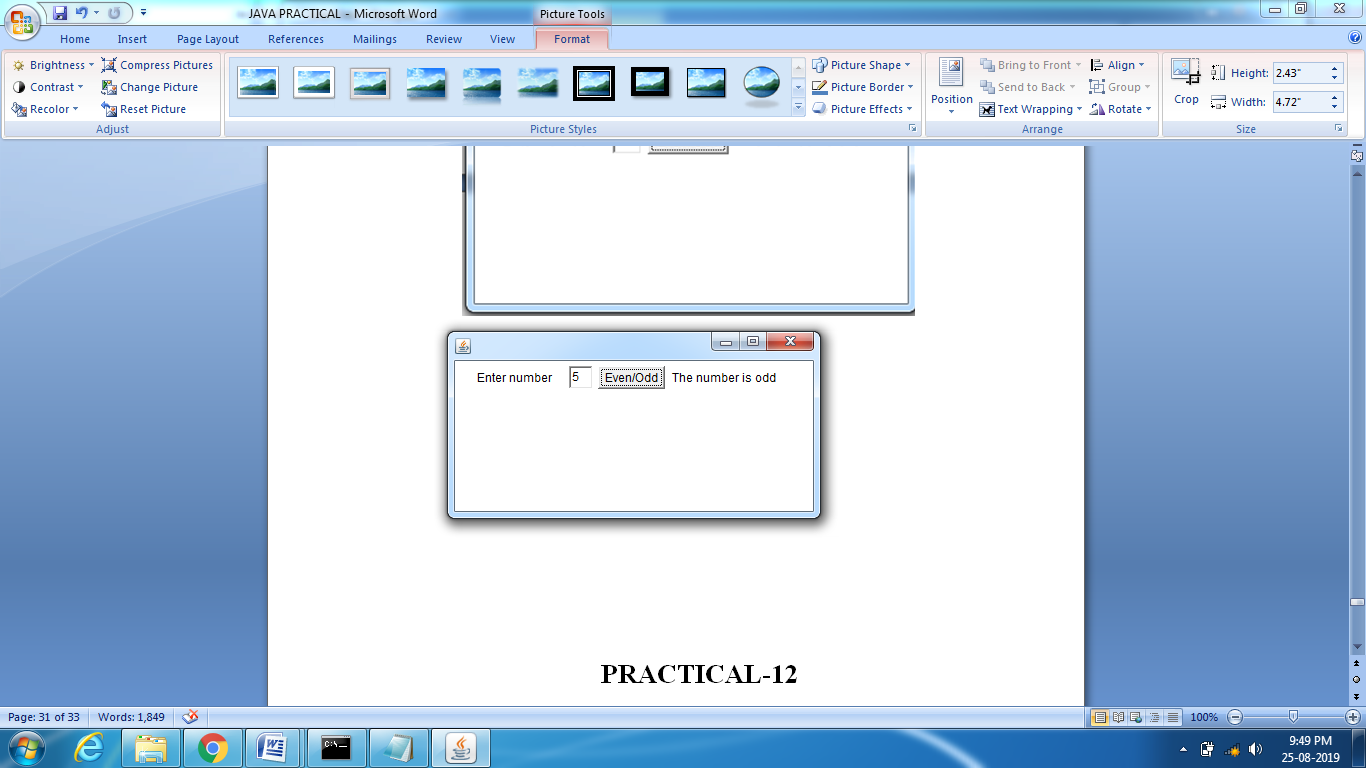
}

}

**Output:**





****

**PRACTICAL-12**

**Aim:** Write a Java List example and demonstrate methods of Java List interface

**Source Code-**

**ListDemo.java**

import java.util.\*;

public class ListDemo

{

public static void main (String[] args)

{

**// Let us create a list**

List l1 = new ArrayList();

l1.add(0, 1);

**// adds 1 at 0 index**

l1.add(1, 2);

**// adds 2 at 1 index**

System.out.println(l1);

**// [1, 2]**

**// Let us create another list**

List l2 = new ArrayList();

l2.add(1);

l2.add(2);

l2.add(3);

**// will add list l2 from 1 index**

l1.addAll(1, l2);

System.out.println(l1);

l1.remove(1);

**// remove element from index 1**

System.out.println(l1);

**// [1, 2, 3, 2]**

**// prints element at index 3**

System.out.println(l1.get(3));

l1.set(0, 5);

**// replace 0th element with 5**

System.out.println(l1);

**// [5, 2, 3, 2]**

}

}

**Output:**

[1, 2]

[1, 1, 2, 3, 2]

[1, 2, 3, 2]

2

[5, 2, 3, 2]