**HARSHINI KULKARNI**

**2GI20CS044**

**25/10/2021**

**TERMWORK 1.1**

1.1) It is required to store and analyze data about 6 car manufacturer’s sales data in all the 12 months of a year. Demonstrate how you would store the data in a two dimensional matrix and do the following

1. Write a function to Find for a given car manufacturer, the month in which, maximum no. of cars are sold.
2. Write a function to Find the average number of cars sold for each car manufacturer
3. Write a function to Find the total number of cars sold for each car manufacturer.
4. Write a function to find standard deviation for a given car manufacturer

Assume – row index 0 - ‘Maruti Suzuki’, 1 – ‘Hundai’ 2 – ‘Tata Motors’ 3-‘KIA’ 4 – ‘BMW’ 5 – ‘Renault’

Col index 0 –‘Jan’, 1-‘Feb’………………………………….11 –‘Dec’

Demonstrate the working of the program with appropriate values for each car manufacturer and the months.

**CODE:**

import java.util.Scanner;

class car {

public static void main(String[] args)

{

Scanner in=new Scanner(System.in);

int carSale[][]=new int[6][12];

int n;

for(int i=0;i<6;i++)

{

System.out.println("Enter the Sale per month for manufacturer "+(i+1)+":");

for(int j=0;j<12;j++)

{

carSale[i][j]=in.nextInt();

}

}

System.out.println("Enter the manufacturer no. to find the maximum car sold:");

n=in.nextInt();

max(carSale,n); //calling max method

System.out.println("\nThe average cars sold by");

average(carSale);

System.out.println("\nThe total carssold by ");

total(carSale);

}

static void max(int[][] carSale,int n)

{

int count=carSale[n-1][0],month=1;

for(int j=0;j<12;j++)

{

if(carSale[n-1][j]>count)

{

count=carSale[n-1][j];

month=j+1;

}

}

switch(month)

{

case 1:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in January.");

break;

case 2:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in February.");

break;

case 3:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in March.");

break;

case 4:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in April.");

break;

case 5:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in May.");

break;

case 6:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in June.");

break;

case 7:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in July.");

break;

case 8:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in August.");

break;

case 9:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in September.");

break;

case 10:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in October.");

break;

case 11:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in November.");

break;

case 12:System.out.println("\nThe maximum number of cars sold by manufacturer "+n+" is in December.");

break;

}

}

static void average(int[][] carSale)

{

float avg;

int sum=0;

for(int i=0;i<6;i++)

{

for(int j=0;j<12;j++)

{

sum+=carSale[i][j];

}

avg=sum/12f;

System.out.println("Manufacturer "+(i+1)+" :"+avg);

sum=0;

}

}

static void total(int[][] carSale)

{

int sum=0;

for(int i=0;i<6;i++)

{

for(int j=0;j<12;j++)

{

sum+=carSale[i][j];

}

System.out.println("Manufacturer "+(i+1)+" :"+sum);

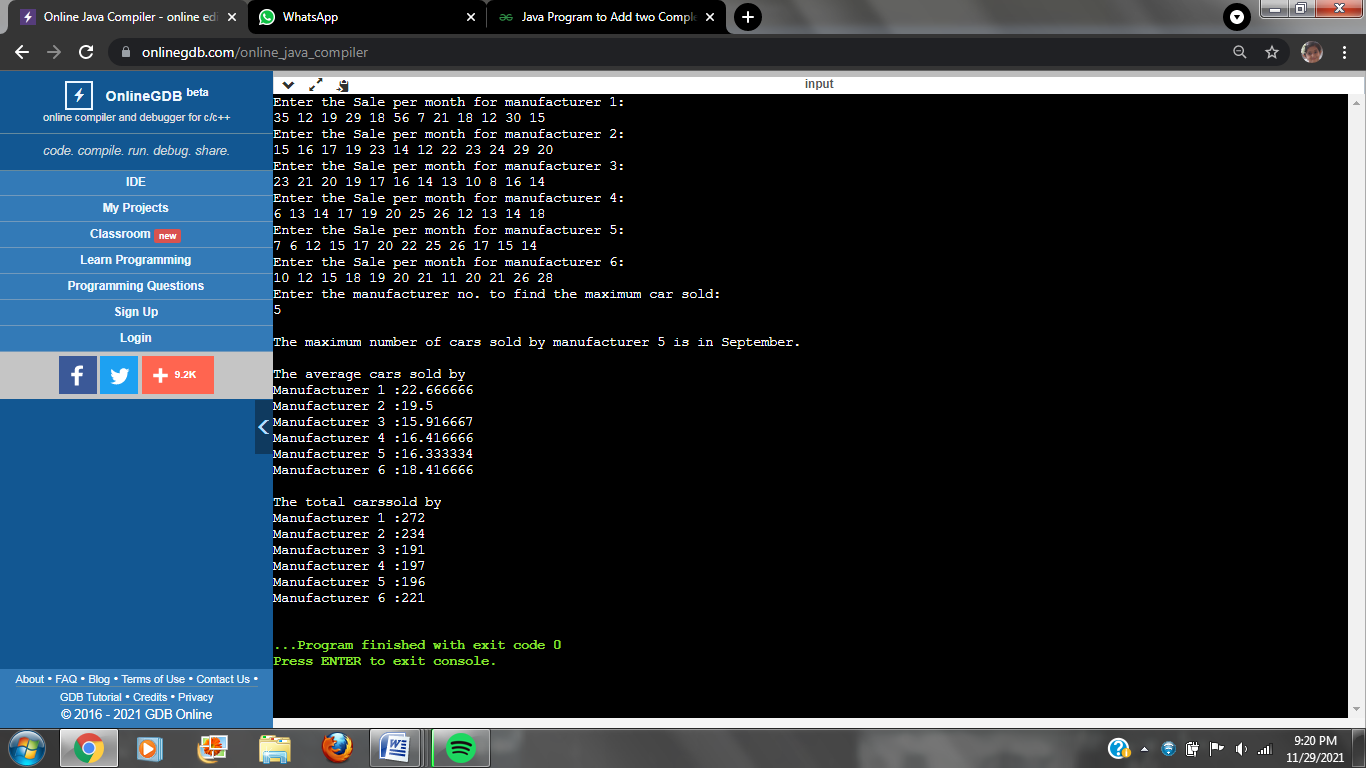
sum=0;

}

}

}

**OUTPUT:**



**HARSHINI KULKARNI  
2GI20CS044**

**25/10/2021**

**TERMWORK 1.4**

1.4) A company has 10 zonal sales offices in four zones namely, North, East, West and South. The company wants to organize the sales data of each of the office in each zone and find answers to queries such as,

1. Which office has performed the highest sales in each zone?
2. What is the average sales done by all the offices in each zone?
3. Which office among each zone is the poorly office?

You are required to answer the following:

1. How do you organize the above data?
2. How do you provide answers to the above queries?

Design a Java application for the same and demonstrate the correctness of the solution.

**CODE:**

import java.util.Scanner;

class office

{

static int maximum(int a[])

{

int m=0,pos=0;

for(int i=0;i<10;i++)

{

if(a[i]>m)

{

m=a[i];

pos=i+1;

}

}

return pos;

}

static int minimum(int a[])

{

int m=99999,pos=0;

for(int i=0;i<10;i++)

{

if(a[i]<m)

{

m=a[i];

pos=i+1;

}

}

return pos;

}

public static void main(String[]args)

{

Scanner in = new Scanner(System.in);

int data[][]=new int[4][10];

int i,j,max[]=new int[4];

int min[]=new int[4];

float avg[]=new float[4];

String zones[]={"North","South","East","West"};

System.out.println("Enter the data of the zones : ");

for(i=0;i<4;i++)

{

System.out.println("\n"+zones[i]+" : ");

int sum=0;

for(j=0;j<10;j++)

{

System.out.println("Office "+(j+1)+" : ");

data[i][j]=in.nextInt();

sum+=data[i][j];

}

avg[i]=sum/10;

max[i] = maximum(data[i]);

min[i] = minimum(data[i]);

}

System.out.println("\nOffice Number with highest sales zone wise : ");

for(i=0;i<4;i++)

System.out.println(zones[i]+" : "+max[i]);

System.out.println("\nOffice with poorest sales zone wise : ");

for(i=0;i<4;i++)

System.out.println(zones[i]+" : "+min[i]);

System.out.println("\nAverage sales zone wise : ");

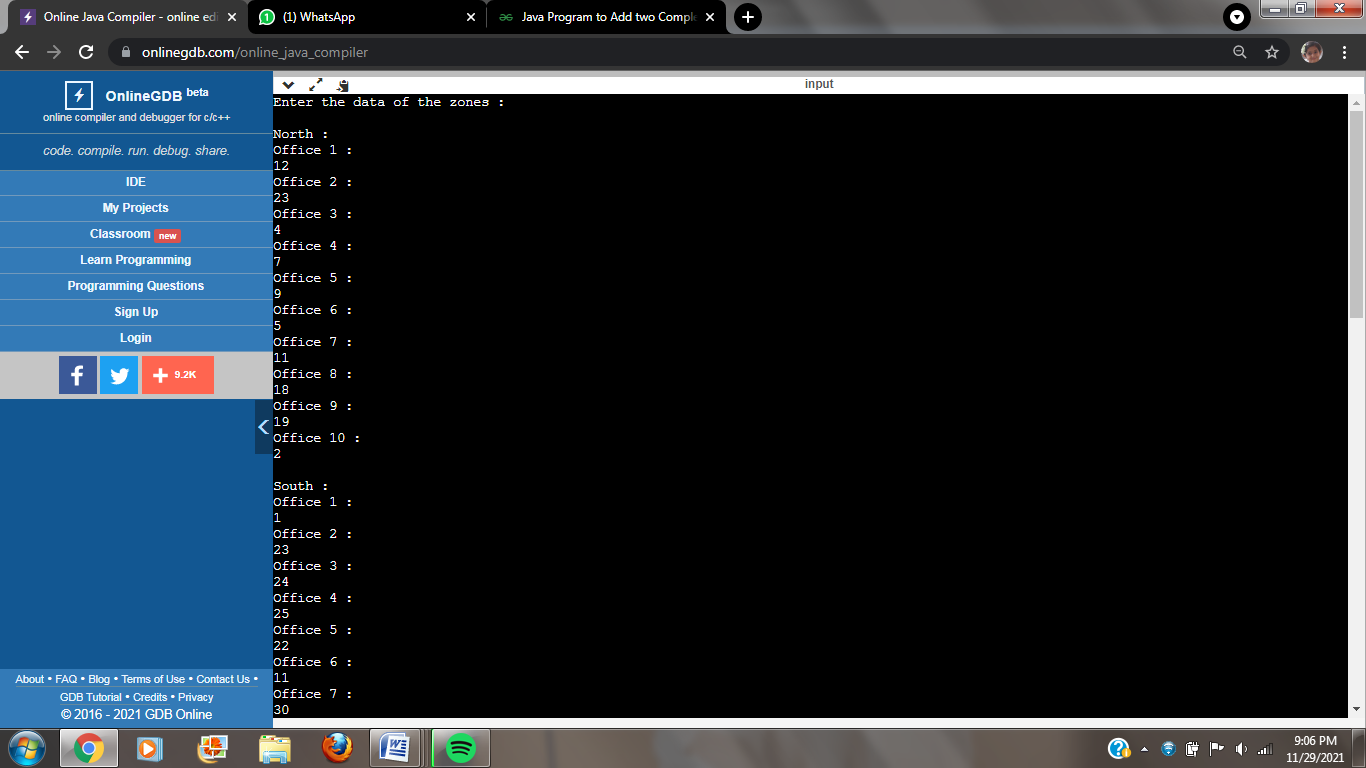
for(i=0;i<4;i++)

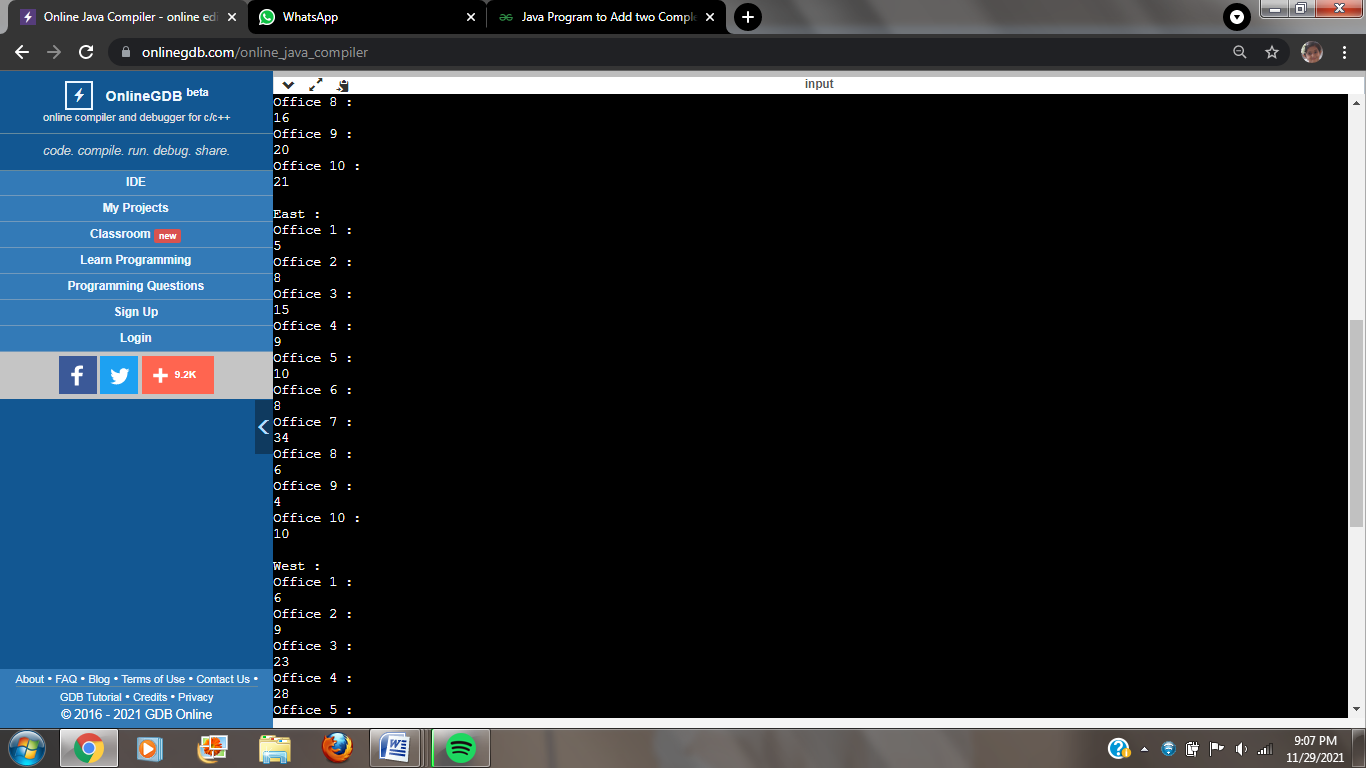
System.out.println(zones[i]+" : "+avg[i]);

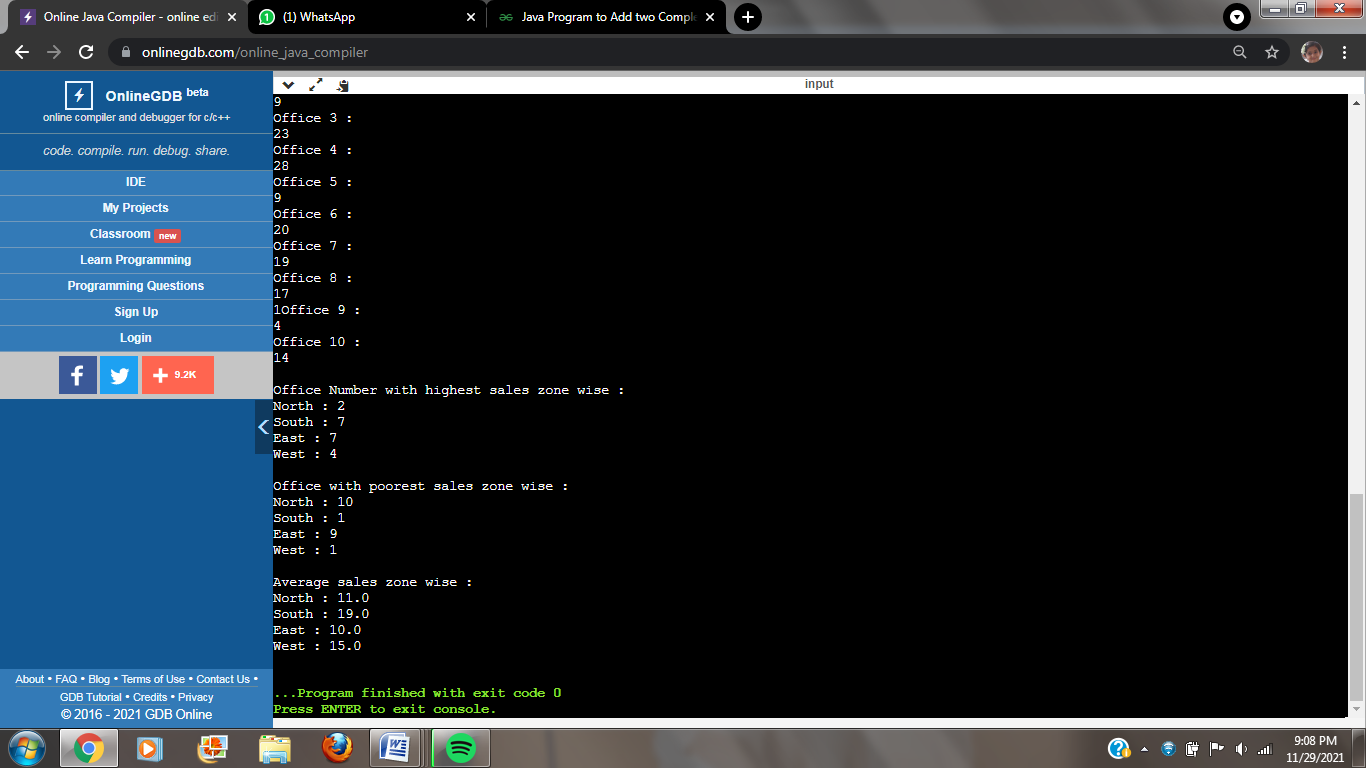
}

}

**OUTPUT:**

****

****

****

**HARSHINI KULKARNI**

**2GI20CS044**

**08/11/2021**

**TERMWORK 2.1**

2.1) Design a class by name myCircle to model Circle geometrical object with its center and radius that enables:

1. Initializing the center, radius and
2. Compute area, perimeter, and diameter of the circle object/s.

TASK 1: Identify member variable/s and their types

TASK 2: Identify Constructor/s along with their arguments (if any) to initialize the member variables

TASK 3: Identify the methods along with their arguments and return types.

TASK 4: Identify member variable getters/setters (if needed)

**CODE:**

import java.util.\*;

class myCircle

{

float x,y,rad;

myCircle()

{

Scanner in=new Scanner(System.in);

System.out.println("Enter radius:");

rad=in.nextFloat();

x=0;

y=0;

}

float diameter()

{

return 2\*rad;

}

float perimeter()

{

return 3.142f\*(diameter());

}

float area()

{

return 3.142f\*rad\*rad;

}

}

class termwork21

{

public static void main (String[] args)

{

myCircle c1=new myCircle();

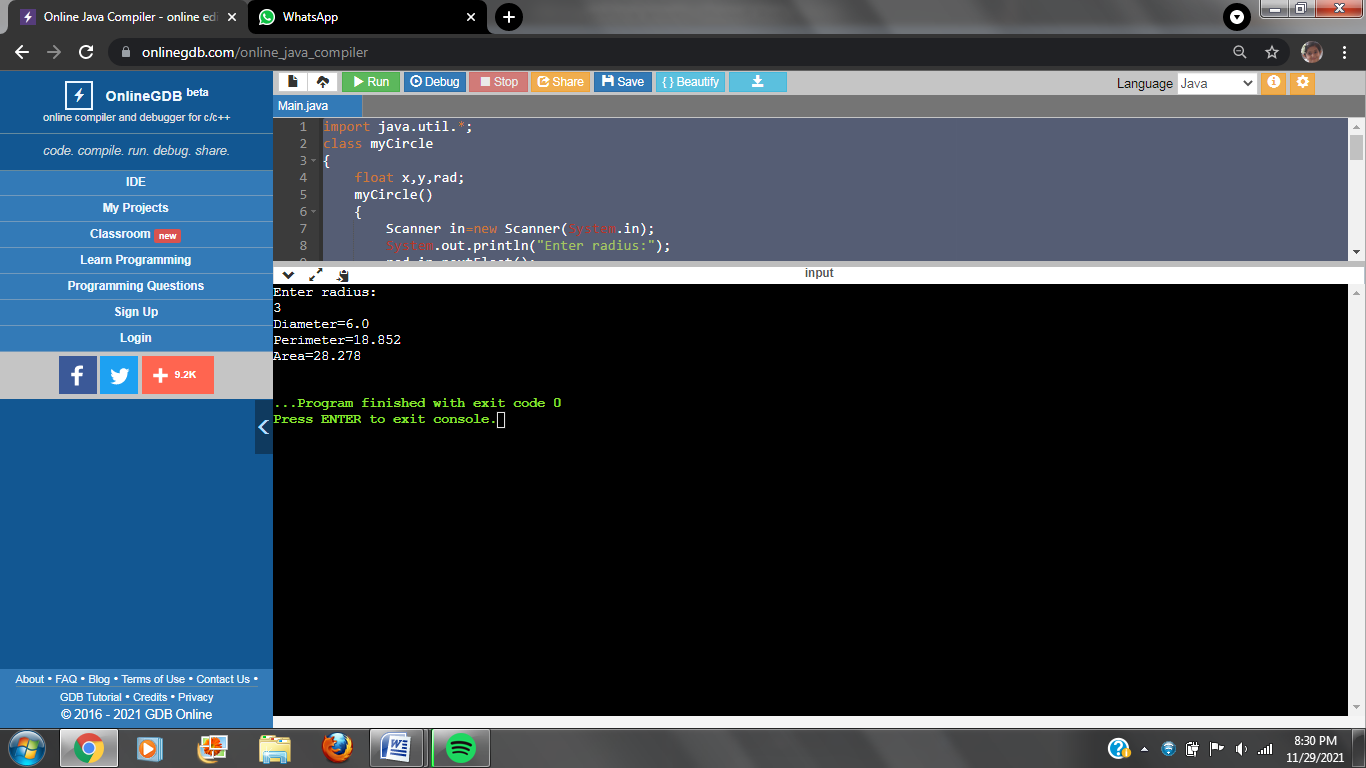
System.out.println("Diameter="+c1.diameter());

System.out.println("Perimeter="+c1.perimeter());

System.out.println("Area="+c1.area());

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**08/11/2021**

**TERMWORK 2.2**

2.2) Define a class to represent the student details such as name, roll number, marks obtained in three internal assessment tests.

1. Identify type and declare the instance variables
2. Identify and develop the constructors to initialize the instance variables
3. Write a method computeAverage() to compute the average as the average of best two marks
4. Write a method to display the student details

Write the corresponding Driver class to instantiate an array of student objects and demonstrate the working of the application by invoking appropriate methods.

**CODE:**

import java.util.\*;

class Student

{

String name;

int rollno;

float m1,m2,m3;

Student()

{

Scanner in=new Scanner(System.in);

System.out.println("Enter name:");

name=in.nextLine();

System.out.println("Enter rollno:");

rollno=in.nextInt();

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

m1=in.nextFloat();

m2=in.nextFloat();

m3=in.nextFloat();

}

Student(String n,int r,float mr1,float mr2,float mr3)

{

name=n;

rollno=r;

m1=mr1;

m2=mr2;

m3=mr3;

}

float computAverage()

{

if(m1>m2 && m2>m3)

return (m1+m2)/2;

else if(m2>m3 && m3>m1)

return (m2+m3)/2;

else

return (m1+m3)/2;

}

void disp()

{

System.out.println("Name:\n"+name+"\nRoll no:"+rollno);

System.out.println("Marks:\n"+m1+"\n"+m2+"\n"+m3);

}

}

public class Termwork22

{

public static void main(String args[])

{

Student s1=new Student();

s1.disp();

System.out.println("Average marks:"+s1.computAverage());

System.out.println("\n");

Student s2=new Student("Harry",28,30,28,29);

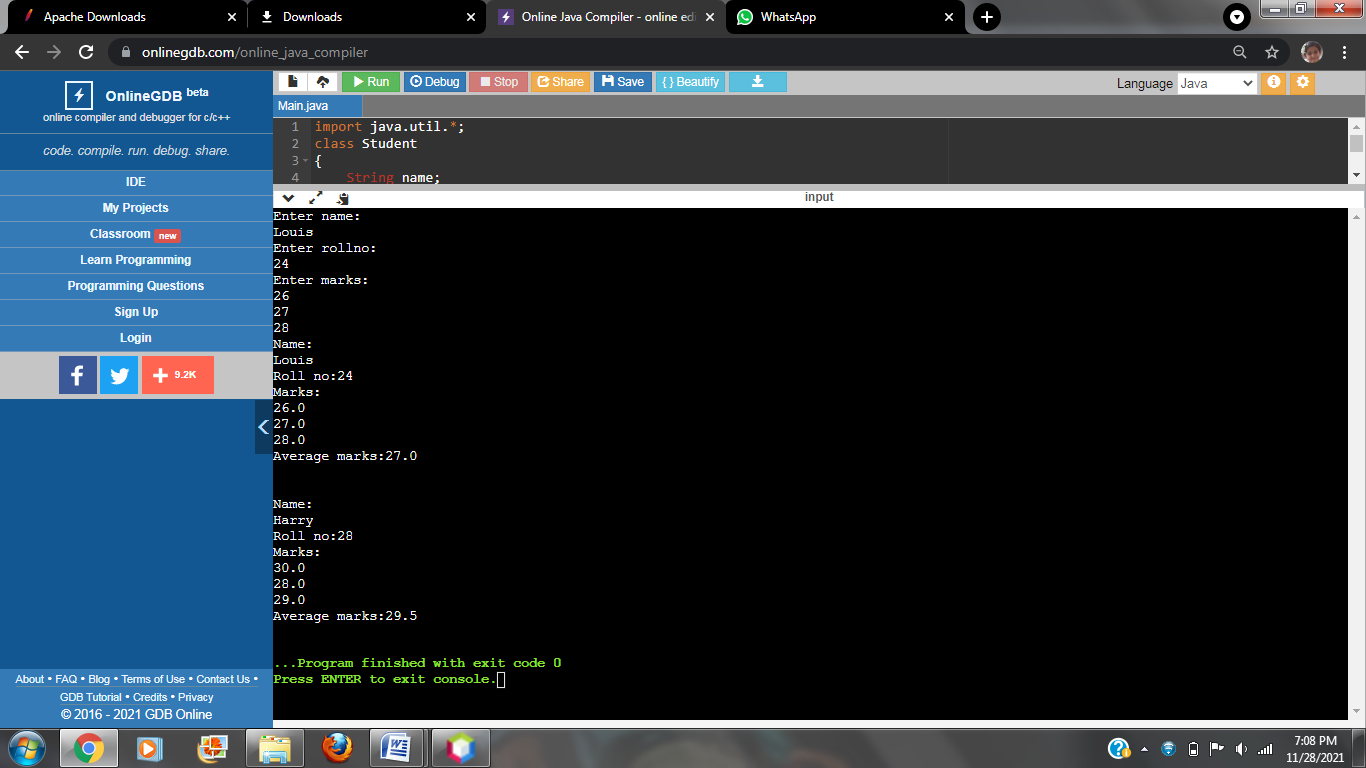
s2.disp();

System.out.println("Average marks:"+s2.computAverage());

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**08/11/2021**

**TERMWORK 3.2**

3.2) Define a class to represent a rectangle in which constructors and parameterized methods are used. It also has a method to compute area of rectangle.

1. First make a class rectangle in which we declare the parameterized constructor.
2. Then demonstrate the use of parameterized method.
3. Use a method to compute area of rectangle.
4. Create a class to demonstrate the call of the methods in previously created class rectangle holding constructors, parameterized methods and method to compute area of rectangle.

**CODE:**

import java.util.\*;

class Rectangle

{

float l,b;

Rectangle()

{

l=0;

b=0;

}

Rectangle(float l1,float b1)

{

l=l1;

b=b1;

}

void read(float l2,float b2)

{

l=l2;

b=b2;

}

float area()

{

return l\*b;

}

void disp()

{

System.out.println("Length="+l+"\nBreadth="+b);

System.out.println("Area of rectangle="+area());

}

}

public class Main

{

public static void main (String[] args)

{

Rectangle r1=new Rectangle(12,11);

r1.disp();

Rectangle r2=new Rectangle();

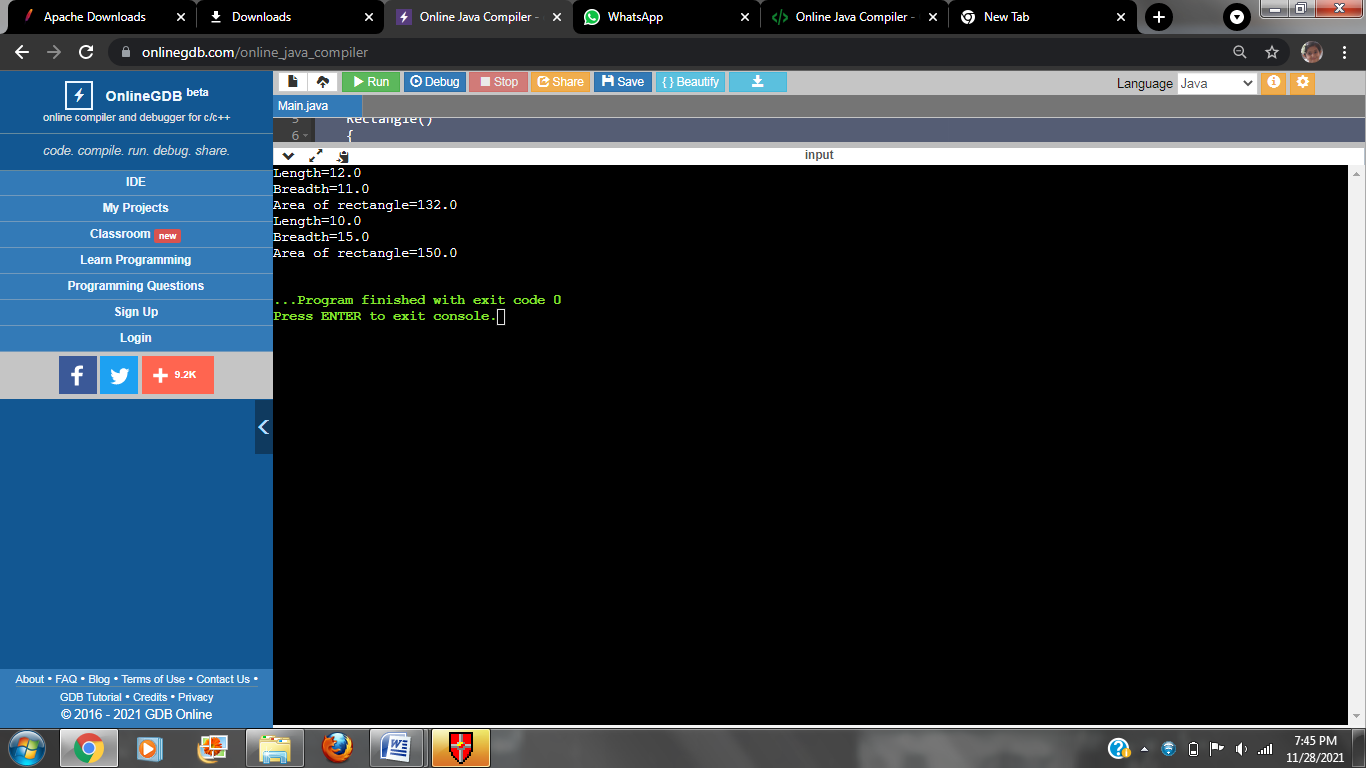
r2.read(10,15);

r2.disp();

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**08/11/2021**

**TERMWORK 3.3**

3.3) Write a Java program to represent a Complex number. Include member functions to:

1. Initialize a complex number to a default value of zero (default constructor)
2. Initialize a complex number to a user defined value (parameterized constructor)
3. Add two complex numbers and return the result. (Parameterized method)
4. Subtract two complex numbers and return the result. (Parameterized method)
5. Display a complex number. (non-parameterized method)

**CODE:**

import java.util.\*;

class Complex

{

int real,img;

Complex()

{

real=0;

img=0;

}

Complex(int r, int i)

{

real=r;

img=i;

}

Complex add(Complex c1, Complex c2)

{

Complex c=new Complex();

c.real=c1.real+c2.real;

c.img=c1.img+c2.img;

return c;

}

Complex sub(Complex c1, Complex c2)

{

Complex c=new Complex();

c.real=c1.real-c2.real;

c.img=c1.img-c2.img;

return c;

}

void disp()

{

if(img>0)

System.out.println(""+real+"+i"+img);

else

System.out.println(""+real+"-i"+(-img));

}

}

class termwork33

{

public static void main(String args[])

{

Complex c1=new Complex(3,5);

Complex c2=new Complex(4,6);

Complex c3=new Complex();

System.out.println("C1:");

c1.disp();

System.out.println("C2:");

c2.disp();

c3=c3.add(c1,c2);

System.out.println("Addition of c1 and c2:");

c3.disp();

c3=c3.sub(c1,c2);

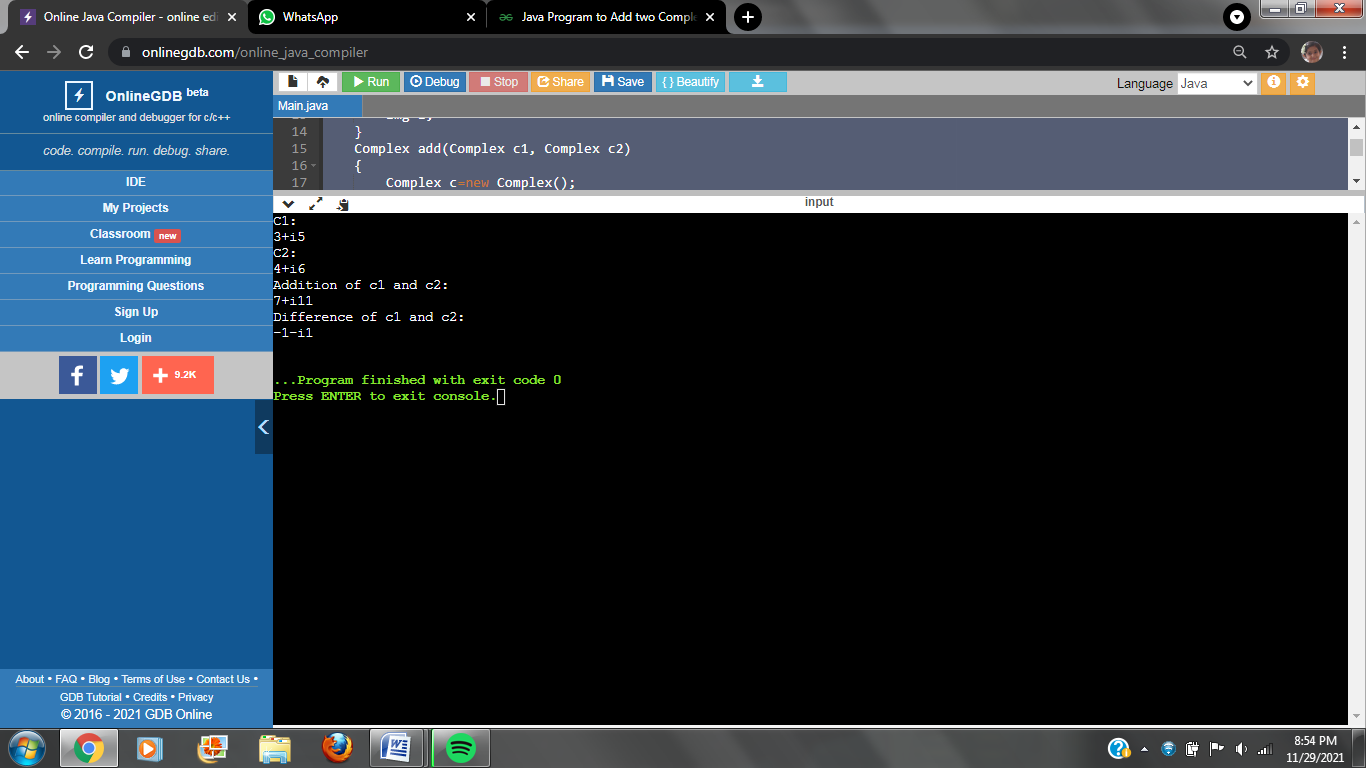
System.out.println("Difference of c1 and c2:");

c3.disp();

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI  
2GI20CS044**

**29/11/2021**

**TERMWORK 4.2:**

4.2)The class Cylinder inherits all the instance variables (radius and color) and methods (getRadius(), getArea(), among others) from its superclass Circle. It further defines a variable called height, three methods getHeight(), setHeight() and getVolume() and its own constructors. Implement the hierarchy as shown below:

Circle

radius:double = 1.0

color:String = “Red”

Circle()

Circle(radius:double)

Circle(radius:double, color:String)

getRadius():double

setRadius(radius:double):void

getColor():String

getColor(color:String):void

getArea():double

Cylinder

height:double = 1.0

Cylinder()

Cylinder(height:double)

Cylinder(height:double, radius:double)

Cylinder(height:double, radius:double,

color:String)

getHeight():double

setHeight(height:double):void

getVolume():double

**CODE:**

import java.util.\*;

class Circle

{

double radius;

String color;

Circle()

{

radius=1.0;

color="Turquoise";

}

Circle(double radius,String color)

{

this.radius=radius;

this.color=color;

}

double getRadius()

{

return radius;

}

void setRadius(double radius)

{

this.radius=radius;

}

String getColor()

{

return color;

}

void setColor(String color)

{

this.color=color;

}

double getArea()

{

return 3.142\*radius\*radius;

}

}

class Cylinder extends Circle

{

double height;

Cylinder()

{

super();

height=1.0;

}

Cylinder(double height)

{

super();

this.height=height;

}

Cylinder(double height,double radius,String color)

{

super(radius,color);

this.height=height;

}

double getHeight()

{

return height;

}

void setHeight(double height)

{

this.height=height;

}

double getVolume()

{

return getArea()\*height;

}

}

class Main

{

public static void main(String args[])

{

Circle c=new Circle(3.0,"Teal");

System.out.println("Radius of Circle 1="+c.getRadius());

System.out.println("Color of circle 1="+c.getColor());

c.setColor("Sapphire");

System.out.println("Changed color of circle 1="+c.getColor());

System.out.println("Area of circle 1="+c.getArea());

Circle ci=new Circle();

System.out.println("\nRadius of Circle 2="+ci.getRadius());

System.out.println("Color of circle 2="+ci.getColor());

ci.setColor("Sapphire");

System.out.println("Changed color of circle 2="+ci.getColor());

System.out.println("Area of circle 2="+ci.getArea());

Cylinder c1=new Cylinder(2,3,"Mauve");

System.out.println("\nRadius of cylinder 1="+c1.getRadius());

System.out.println("Height of cylinder 1="+c1.getHeight());

System.out.println("Color of cylinder 1="+c1.getColor());

System.out.println("Volume of cylinder 1="+c1.getVolume());

Cylinder c2=new Cylinder();

System.out.println("\nRadius of cylinder 2="+c2.getRadius());

System.out.println("Height of cylinder 2="+c2.getHeight());

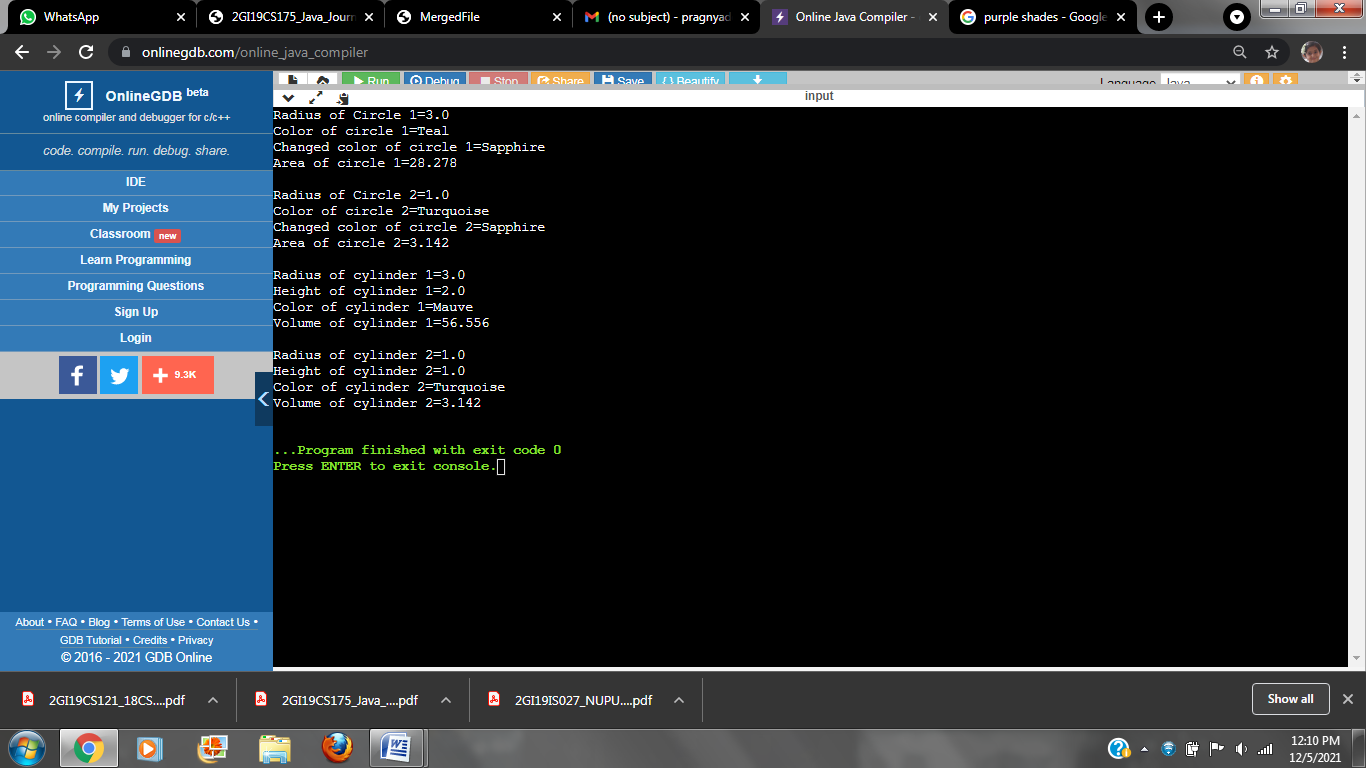
System.out.println("Color of cylinder 2="+c2.getColor());

System.out.println("Volume of cylinder 2="+c2.getVolume());

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI  
2GI20CS044**

**29/11/2021**

**TERMWORK 4.4**

* 1. Implement a BANK class to demonstrate the inheritance in Java by implementing getRateOfInterestmember function for three different banks, as shown below.

Java method overriding example of bank

**CODE:**

import java.util.\*;

class Bank

{

protected String name;

protected double balance;

protected int accountNumber;

protected float inamt;

Bank( String name,double balance, int accountNumber , float inamt)

{

this.balance=balance;

this.name=name;

this.accountNumber=accountNumber;

this.inamt=inamt;

}

void disp()

{

System.out.println("Account holder:"+name+"\nAccount number:"+accountNumber);

System.out.println("Balance:"+balance);

}

public void getRateofInterest(int t)

{

System.out.println("The interest amount is "+inamt);

}

}

class SBI extends Bank

{

SBI(String name,double balance, int accountNumber , float r)

{

super(name,balance,accountNumber ,r);

}

public void getRateofInterest(int t)

{

super.getRateofInterest(t);

float i=(float)(inamt/(balance\*t)\*100);

System.out.println("Rate of interest ="+i+"%");

}

}

class icici extends Bank

{

icici(String name,double balance, int accountNumber ,float inamt)

{

super(name,balance,accountNumber ,inamt);

}

public void getRateofInterest(int t)

{

super.getRateofInterest(t);

float i=(float)(inamt/(balance\*t)\*100);

System.out.println("Rate of interest ="+i+"%");

}

}

class Axis extends Bank

{

Axis(String name,double balance, int accountNumber , float inamt)

{

super(name,balance,accountNumber ,inamt);

}

public void getRateofInterest(int t)

{

super.getRateofInterest(t);

float i=(float)(inamt/(balance\*t)\*100);

System.out.println("Rate of interest ="+i+"%");

}

}

class Main

{

public static void main(String args[])

{

SBI s=new SBI("Hozier",45100,123,2500);

s.disp();

s.getRateofInterest(3);

Axis a=new Axis("Harry",28000,12,1500);

a.disp();

a.getRateofInterest(2);

icici i=new icici("Louis",29000,24,1800);

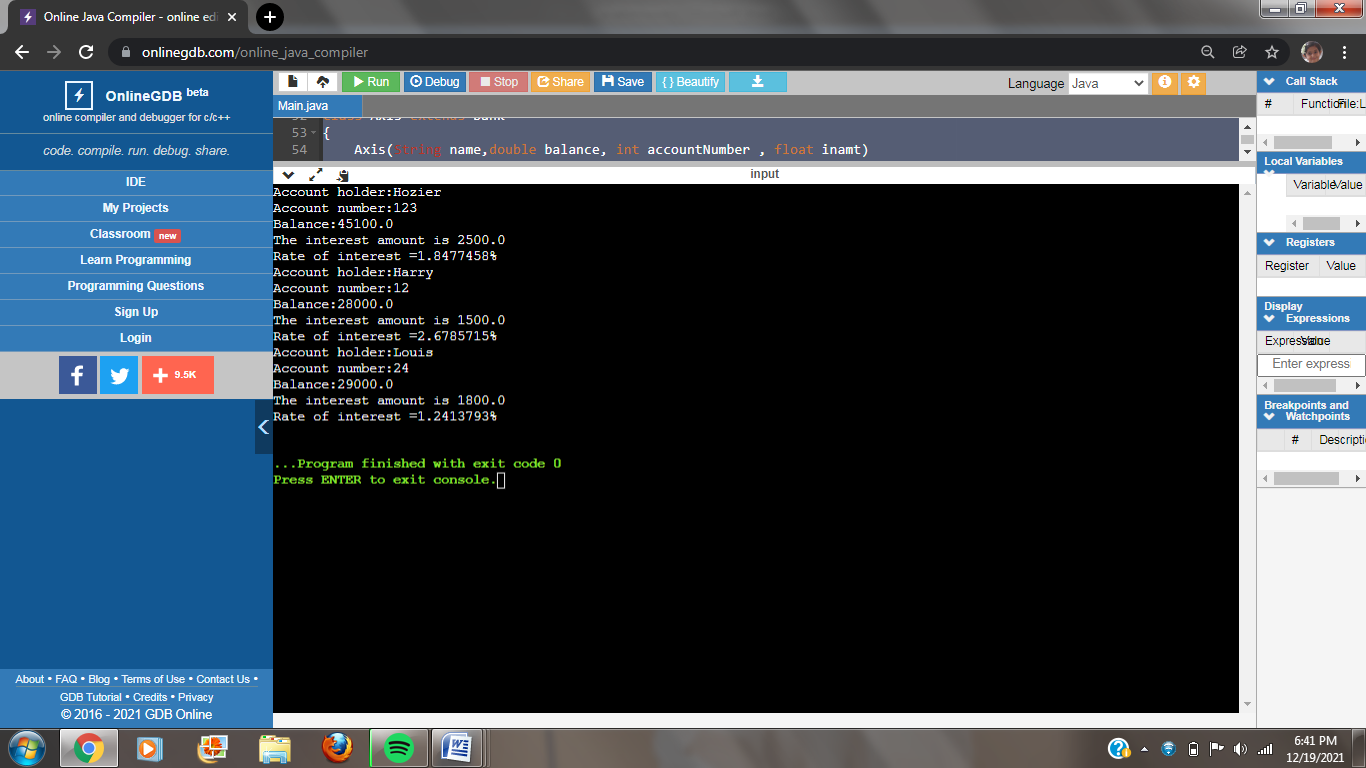
i.disp();

i.getRateofInterest(5);

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI  
2GI20CS044**

**05/11/2021**

**TERMWORK 5b.3**

5b.3) Design a base class where Hospital provides no. of patients admitted in it. Number of patients varies with the different hospitals, For example Health India hospital has 1657 patients, IVY hospital has 2965 patients  and Apollo Hospital has 1631 patients. Hospital parent class which has one method getNumberOfPatients() and sub class HealthIndia, IVY and Apolo class which extends parent class & override its method.

**CODE:**

import java.util.\*;

class Hospital

{

protected int nop;

Hospital(int nop)

{

this.nop=nop;

}

int getNumberofPatients()

{

return nop;

}

}

class HealthIndia extends Hospital

{

HealthIndia(int nop)

{

super(nop);

}

int getNumberofPatients()

{

return nop;

}

}

class IVY extends Hospital

{

IVY(int nop)

{

super(nop);

}

int getNumberofPatients()

{

return nop;

}

}

class Apollo extends Hospital

{

Apollo(int nop)

{

super(nop);

}

int getNumberofPatients()

{

return nop;

}

}

class Main

{

public static void main(String args[])

{

HealthIndia h=new HealthIndia(1657);

System.out.println("Number of patients in HealthIndia is "+h.getNumberofPatients());

IVY i=new IVY(2965);

System.out.println("Number of patients in IVY is "+i.getNumberofPatients());

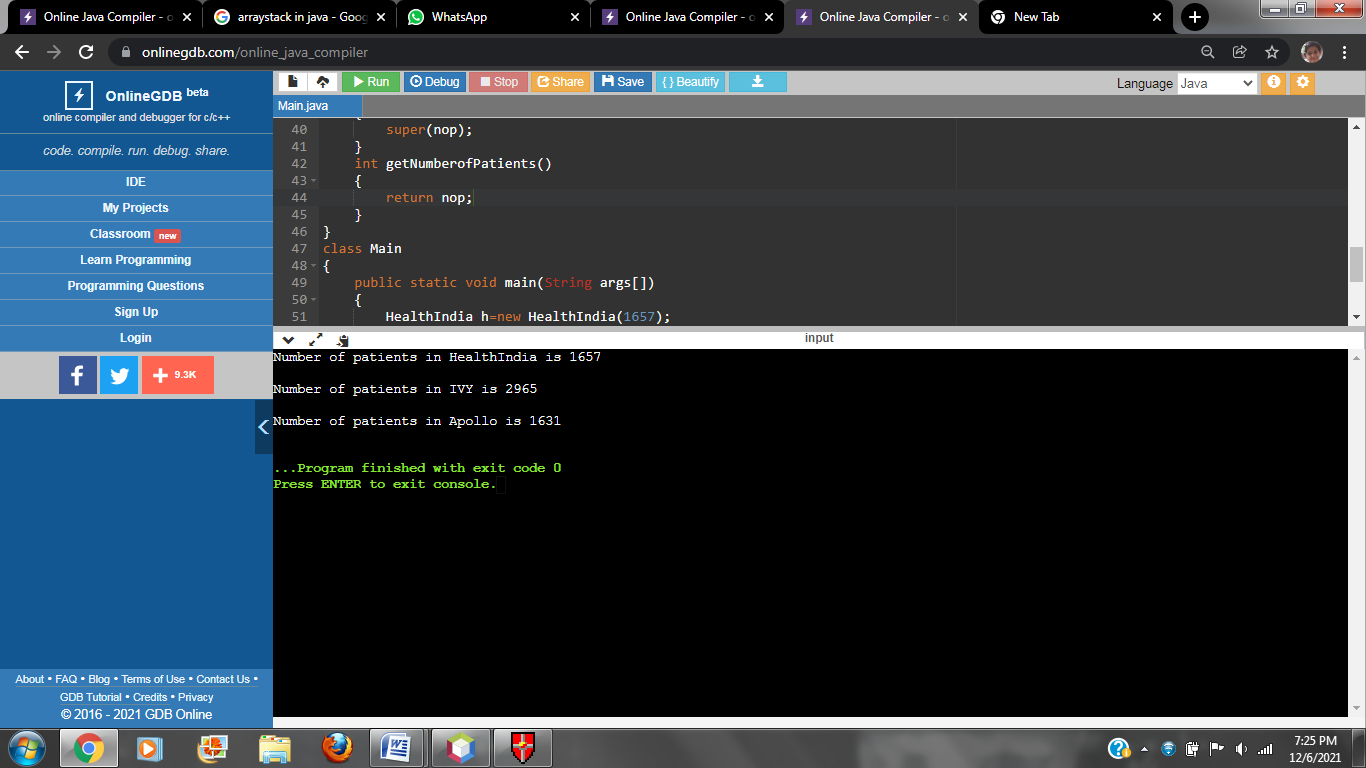
Apollo a=new Apollo(1631);

System.out.println("Number of patients in Apollo is "+a.getNumberofPatients());

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**05/12/2021**

**TERMWORK 5.2**

* 1. Implement a linear search function by using method overloading concept for an array of integers, double and character elements.

**CODE:**

import java.util.Scanner;

class Mysearch{

int linearsearch(int []a,int key){

for(int i=0;i<a.length;i++){

if(a[i]==key)

return i;

}

return -1;

}

int linearsearch(double []a,double key){

for(int i=0;i<a.length;i++){

if(a[i]==key)

return i;

}

return -1;

}

int linearsearch(char []a,char key){

for(int i=0;i<a.length;i++){

if(a[i]==key)

return i;

}

return -1;

}

}

public class Main

{

public static void main(String[] args)

{

Mysearch m=new Mysearch();

Scanner in=new Scanner(System.in);

System.out.println("1:Integers\n2:Double\n3:Character\n");

int c=in.nextInt();

System.out.println("how many elements to be entered: ");

int n=in.nextInt();

switch(c){

case 1:int[]a=new int[n];

for(int i=0;i<n;i++)

a[i]=in.nextInt();

System.out.println("Enter the Element to be searched: ");

int key=in.nextInt();

int pos=m.linearsearch(a, key);

if(pos==-1){

System.out.println("Element not found");

}

else{

System.out.println("Element found at "+(pos+1)+" Position");

}

break;

case 2:double[]b=new double[n];

for(int i=0;i<n;i++)

b[i]=in.nextDouble();

System.out.println("Enter the Element to be searched: ");

double key1=in.nextDouble();

int pos1=m.linearsearch(b, key1);

if(pos1==-1){

System.out.println("Element not found");

}

else{

System.out.println("Element found at "+(pos1+1)+" Position");

}

break;

case 3: char[]d=new char[n];

for(int i=0;i<n;i++)

d[i]=in.next().charAt(0);

System.out.println("Enter the Element to be searched: ");

char key2=in.next().charAt(0);

int pos2=m.linearsearch(d, key2);

if(pos2==-1){

System.out.println("Element not found");

}

else{

System.out.println("Element found at "+(pos2+1)+" Position");

}

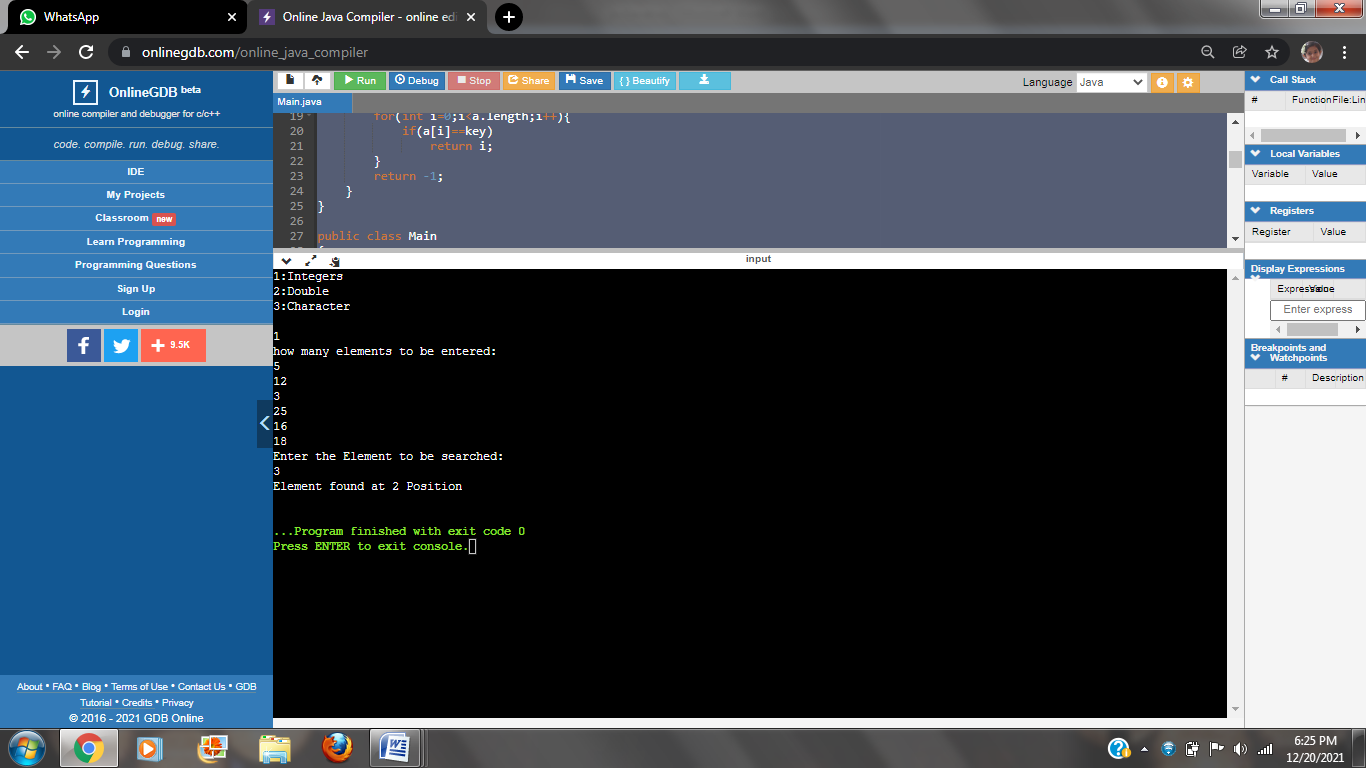
break;

}

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**06/12/2021**

**TERMWORK 6.2:**

6.2) Implement the following inheritance hierarchy.

Shape (Abstract class)

area,perimeter: float

type: String

abstract computeArea()

abstract computePerimeter()

Rectangle

(Concrete class)

length, width: float

Circle

(Concrete class)

radius: float

Triangle

(Concrete class)

base, height: float

**CODE:**

import java.util.\*;

abstract class Shape

{

protected float area,perimeter;

protected String type;

public abstract void CompArea();

public abstract void CompPerimeter();

}

class Rectangle extends Shape

{

private float length,width;

public Rectangle(float length,float width,String type)

{

this.length=length;

this.width=width;

this.type=type;

}

public void CompArea()

{

area=length\*width;

System.out.println("Area of Rectangle="+area+"\nType:"+type);

}

public void CompPerimeter()

{

perimeter=2\*(length+width);

System.out.println("Perimeter of Rectangle="+perimeter+"\n");

}

}

class Circle extends Shape

{

private float radius;

public Circle(float radius,String type)

{

this.radius=radius;

this.type=type;

}

public void CompArea()

{

area=3.142f\*radius\*radius;

System.out.println("Area of Circle="+area+"\nType:"+type);

}

public void CompPerimeter()

{

perimeter=2\*3.142f\*radius;

System.out.println("Perimeter of Circle="+perimeter+"\n");

}

}

class Triangle extends Shape

{

private float base,height;

public Triangle(float base,float height,String type)

{

this.base=base;

this.height=height;

this.type=type;

}

public void CompArea()

{

area=0.5f\*base\*height;

System.out.println("Area of Triangle="+area+"\nType:"+type);

}

public void CompPerimeter()

{

float c;

Scanner in=new Scanner(System.in);

System.out.println("Enter 3rd side:");

c=in.nextFloat();

perimeter=base+height+c;

System.out.println("Perimeter of Triangle="+perimeter);

}

}

class Main

{

public static void main (String args[])

{

Rectangle r=new Rectangle(4,5,"Parallelogram");

r.CompArea();

r.CompPerimeter();

Circle c=new Circle(3,"Circle");

c.CompArea();

c.CompPerimeter();

Triangle t=new Triangle(3,4,"Right angled");

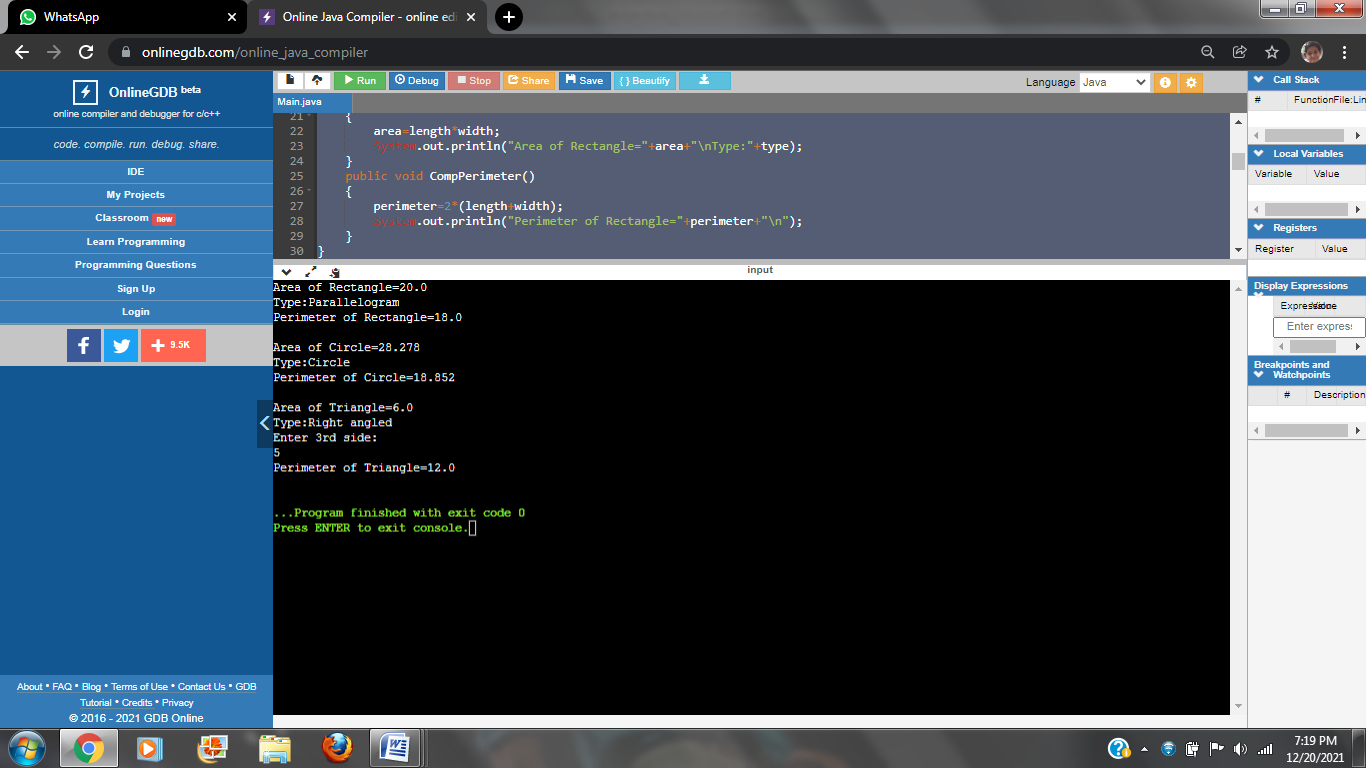
t.CompArea();

t.CompPerimeter();

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**06/12/2021**

**TERMWORK 6.3**

**6.3)** Write a Java program that defines an abstract class called Account and accepts the following customer account information:  **1) Customer Name   
 2) Account Number   
 3) Balance** and provides below operations on customer account:   
 **1) Deposit   
 2) Withdraw   
 3) Display Balance   
 4) Display full account details**

­ There are two types of accounts – Savings and Current. The Current account has an overdraft facility limited to Rs. 75,000 per account. The following constraints hold on these accounts:

Savings Account:

1. The total number of deposits for a Savings account cannot exceed three per day.
2. The amount deposited into a savings account cannot exceed Rs.5000 in each transaction.
3. The maximum withdrawal amount is Rs.25,000 per transaction.
4. The minimum balance to be maintained is Rs. 10,000.

Current account:

1. The amount withdrawn cannot exceed the overdraft limit once the account balance is zero.
2. Maximum number of withdrawals is two.
3. No limit on the number of deposits.
4. Each deposit cannot exceed Rs. 25,000.

Test the program by creating objects of the Savings and Current Accounts.

**CODE:**

import java.util.\*;

abstract class Account

{

protected String cusName,acNo;

protected float bal;

public abstract void Deposit(float dep);

public abstract void Withdraw(float wdw);

public void DispBal()

{

System.out.println("Balance="+bal);

}

public void disp()

{

System.out.println("Customer details are:");

System.out.println("Customer Name:"+cusName+"\nAccount No.:"+acNo);

System.out.println("Balance:"+bal);

}

}

class Savings extends Account

{

private int n=0;

public Savings(String c,String ac,float b)

{

cusName=c;

acNo=ac;

bal=b;

}

public void Deposit(float dep)

{

n++;

if(n<=3)

{

if(dep<=5000)

{

bal=bal+dep;

System.out.println("Amount has been deposited");

DispBal();

}

else

System.out.println("Amount exceeds the limit");

}

else

System.out.println("You've reached maximum limit of deposition");

}

public void Withdraw(float wdw)

{

if((bal-wdw)<10000)

System.out.println("Insufficient balance");

else

{

if(wdw>25000)

System.out.println("Amount exceeds maximum withdrawal limit");

else

{

bal=bal-wdw;

System.out.println("Amount successfully withdrawn");

DispBal();

}

}

}

}

class Current extends Account

{

private int n=0;

public Current(String c,String ac,float b)

{

cusName=c;

acNo=ac;

bal=b;

}

public void Deposit(float dep)

{

bal=bal+dep;

System.out.println("Amount has been deposited");

DispBal();

}

public void Withdraw(float wdw)

{

n++;

if(n<=2)

{

if((bal-wdw)<-75000){

n--;

System.out.println("Amount exceeds overdraft amount");

}

else

{

bal=bal-wdw;

System.out.println("Amount successfully withdrawn");

DispBal();

}

}

else

System.out.println("You've exceeded withdrawal limit");

}

}

class Main

{

public static void main(String args[])

{

Savings s=new Savings("Hozier","221B",28000);

s.disp();

System.out.println("Account Type:Savings");

s.Deposit(2000);

s.Deposit(1000);

s.Deposit(2500);

s.Deposit(3000);

s.Withdraw(1000);

s.disp();

System.out.println("");

Current c=new Current("Louis","28Lou",0);

c.disp();

System.out.println("Account Type:Current");

c.Withdraw(20000);

c.Withdraw(56000);

c.Withdraw(1000);

c.Withdraw(1000);

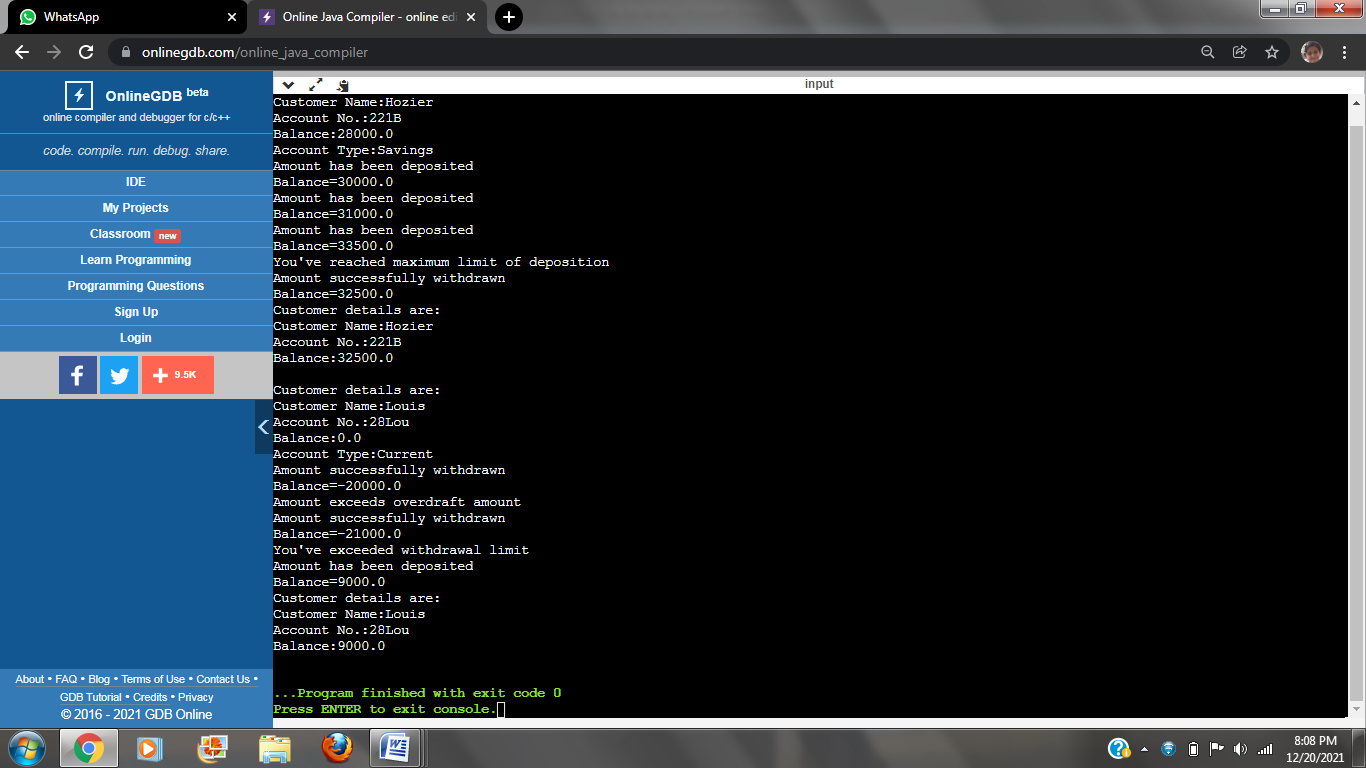
c.Deposit(30000);

c.disp();

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**13/12/2021**

**TERMWORK 7.2**

**7.2)** Write a JAVA program which has  
  
 i. An Interface class for Stack Operations (viz., push(), pop(), peek(),display())  
 ii. A Class that implements the Stack Interface and creates a fixed length Stack.  
 iii. A Class that implements the Stack Interface and creates a Dynamic Length Stack.  
 iv. A Class that uses both the above Stacks through Interface reference and does the Stack operations that demonstrates the runtime binding.

**Code:**

interface stackop

{

void push(int item);

int pop();

}

class FixedStack implements stackop

{

private int stk[ ];

private int top;

FixedStack(int size)

{

stk=new int[size];

top=-1;

}

public void push(int item)

{

if(top==stk.length-1)

{

System.out.println("Stack Overflows");

int t[ ]=new int[stk.length \* 2];

for(int i=0;i<stk.length;i++)

t[i]=stk[i];

stk=t;

stk[++top]=item;

}

else

stk[++top]=item;

}

public int pop()

{

if(top<0)

{

System.out.println("Stack Underflows");

return 0;

}

else

return stk[top--];

}

}

class DynStack implements stackop

{

private int stk[ ];

private int top;

DynStack(int size)

{

stk=new int[size];

top=-1;

}

public void push(int item)

{

if(top==stk.length-1)

{

System.out.println("Stack Overflows.");

int t[ ]=new int[stk.length \* 2];

for(int i=0;i<stk.length;i++)

t[i]=stk[i];

stk=t;

stk[++top]=item;

}

else

stk[++top]=item;

}

public int pop()

{

if(top<0)

{

System.out.println("Stack Underflows.");

return 0;

}

else

return stk[top--];

}

}

class StackTest

{

public static void main(String args[ ])

{

FixedStack fs=new FixedStack(3);

DynStack ds=new DynStack(5);

stackop mystk;

for(int i=0;i<3;i++)

fs.push(i);

System.out.println("Fixed length Stack Contents are.");

for(int i=0;i<3;i++)

System.out.println(fs.pop());

for(int i=0;i<7;i++)

ds.push(i);

System.out.println("Dynamic length Stack Contents are");

for(int i=0;i<7;i++)

System.out.println(ds.pop());

mystk=fs;

for(int i=0;i<3;i++)

mystk.push(i);

System.out.println("Fixed length Stack Contents using interface reference");

for(int i=0;i<3;i++)

System.out.println(mystk.pop());

mystk=ds;

for(int i=0;i<7;i++)

mystk.push(i);

System.out.println("Dynamic length Stack Contents using interface reference");

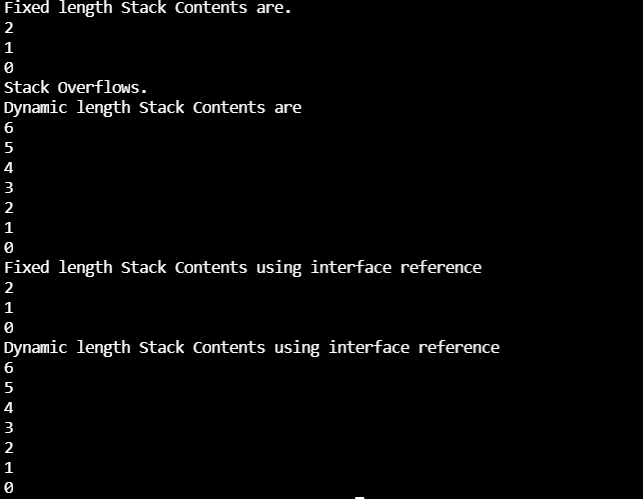
for(int i=0;i<7;i++)

System.out.println(mystk.pop());

}

}

**OUTPUT:**

****

**HARSHINI KULKARNI**

**2GI20CS044**

**13/12/2021**

**TERMWORK 7.3**

**7.3)**Design an interface IAnimal that has walk, and sleep methods, an interface IBird that has fly, and peck methods, an interface IHuman that has eat and speak methods. Construct a Bird class that implements IAnimal and IBird interfaces and also construct Human class that implements IAnimal and IHuman interfaces. Demonstrate the working of these methods by invoking the methods using appropriate reference variables.

DemoClass

Human

Bird

IBird

+fly( ) + peck( )

IAnimal

+sleep( ) +walk( )

IHuman

+speak( ) +eat( )

**Code:**

interface IAnimal

{

abstract void sleep();

abstract void walk();

}

interface IBird

{

abstract void fly();

abstract void peck();

}

interface IHuman

{

abstract void eat();

abstract void speak();

}

class Bird implements IAnimal, IBird

{

public void sleep()

{

System.out.println("Bird is sleeping");

}

public void walk()

{

System.out.println("Bird is walking");

}

public void fly()

{

System.out.println("Bird is flying");

}

public void peck()

{

System.out.println("Bird is pecking");

}

}

class Human implements IAnimal,IHuman

{

public void sleep()

{

System.out.println("Human is sleeping");

}

public void walk()

{

System.out.println("Human is walking");

}

public void eat()

{

System.out.println("Human is eating");

}

public void speak()

{

System.out.println("Human is speakng");

}

}

class animal

{

public static void main(String[] args)

{

Bird b=new Bird();

b.fly();

b.peck();

b.sleep();

b.walk();

Human h= new Human();

h.sleep();

h.walk();

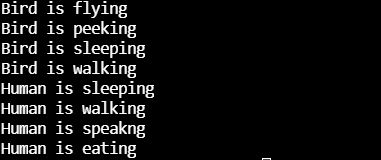
h.speak();

h.eat();

}

}

**OUTPUT:**



**HARSHINI KULKARNI**

**2GI20CS044**

**20/12/2021**

**TERMWORK 8.4**

8.4) Write java program that takes the value of num variable and checks it is odd, then the throw keyword will raise the user defined exception and the catch block will get executed. OddNumberException class is derived from the Exception class. To implement user defined exception throw an exception object explicitly.

**Code:**

import java.util.Scanner;

class OddNumberException extends Exception {

OddNumberException(String m){

super(m);

}

public String toString() {

return "You just entered an odd number.";

}

}

public class termwork84 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter an integer: ");

int num = sc.nextInt();

try { if(num%2!=0) {

throw new OddNumberException("odd number encountered");

}

else

System.out.println("You entered an even number.");

}

catch(OddNumberException e) {

System.out.println(e.getMessage());

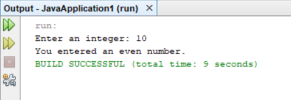
System.out.println(e);

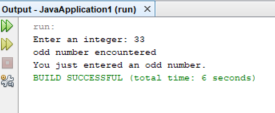
}

}

}

**Output:**





**HARSHINI KULKARNI**

**2GI20CS044**

**20/12/2021**

**TERMWORK 8.2:**

8.2) Design a class CustomerAccount that has acctNum, custName and balance as member variables and a constructor to initialize these. Implement withDraw and depositAmount methods that accepts amount as argument and it must throw an user defined exception called InsufficientBalalance/InvalidAmount exception when amount is greater than balance/ amount is negative respectively. Design two classes InsufficientBalance and InvalidAmount that extend the Exception class and override toString method. Demonstrate the working of the user defined exceptions by instantiating an object of customerAccount class and invoking withDraw and depoistAmount in try… catch.. finally block.

Exception

InsufficientBalance

+getMessage()

InvalidAmount

+getMessage( )

CustomerAccount

-accNum -balance -custName

+withDraw(float amount ) throws InsufficientBalance

+depositAmount(float amount ) throws InvalidAmount

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

CustomerAccount newCustomer=new CustomerAccount( );

try{ newCustomer.withDraw(7000);

newCustomer.depositAmount(-500);

}catch(InsufficientBalance e) { e.getMessage( );}

catch(InvalidAmount e) { e.getMessage( ); } finally{ println(“Transaction failed…”); }

} }

**CODE:**import java.util.Scanner;

class CustomerAccount{

int accNo;

String custName;

double balance;

CustomerAccount(int accNo,String custName,double balance){

this.accNo = accNo;

this.custName = custName;

this.balance = balance;

}

void withdrawAmt(double Amt)

{

try{

if(balance-Amt <0)

throw new InsufficientBalException("Insufficient Balance");

balance-=Amt;

System.out.println("Balance in the account is "+ balance);

}

catch (InsufficientBalException e)

{

System.out.println(e.getMessage());

System.out.println(e);

System.out.println();

}

}

void depositAmt(double Amt)

{

try

{

if(Amt<0)

throw new InvalidAmtException("Invalid amount");

balance+=Amt;

System.out.println("Balance in the account is "+ balance);

}

catch( InvalidAmtException e)

{

System.out.println(e.getMessage());

System.out.println(e);

System.out.println();

}

}

}

class InsufficientBalException extends Exception{

InsufficientBalException(String msg)

{

super(msg);

}

public String toString()

{

return "Amount can't be withdrawm as there is insufficient Balance";

}

}

class InvalidAmtException extends Exception{

InvalidAmtException(String msg)

{

super(msg);

}

public String toString()

{

return "Can't deposit as amount is negative";

}

}

public class Main {

public static void main(String[] args) {

CustomerAccount c =new CustomerAccount(101,"Harry Styles",1000);

c.depositAmt(-500);

c.withdrawAmt(1200);

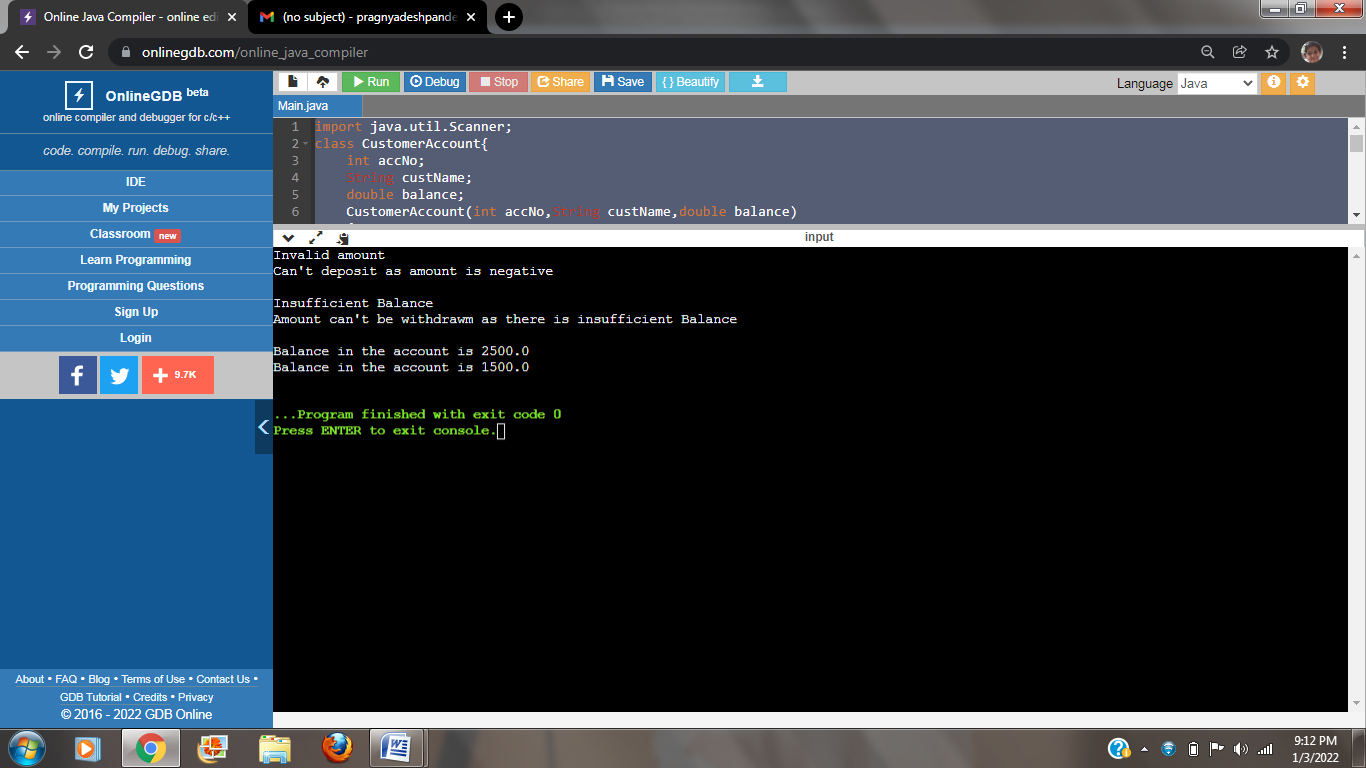
c.depositAmt(1500);

c.withdrawAmt(1000);

}

}

**OUTPUT:**



**HARSHINI KULKARNI**

**2GI20CS044**

**20/12/2021**

**TERMWORK 9.1:**

9.1)Read a string containing 3\_4 words using Scanner class object. Split it into words and for

each word check if it is palindrome by writing a function isPalindrome(String the myWord, int

s, int e) which return true if its palindrome else return false. Where s is start index and e is

end index of the input myWord. Print it in uppercase if it is palindrome else reverse the string

and print it in lowercase.  Use appropriate string functions to implement the above problem

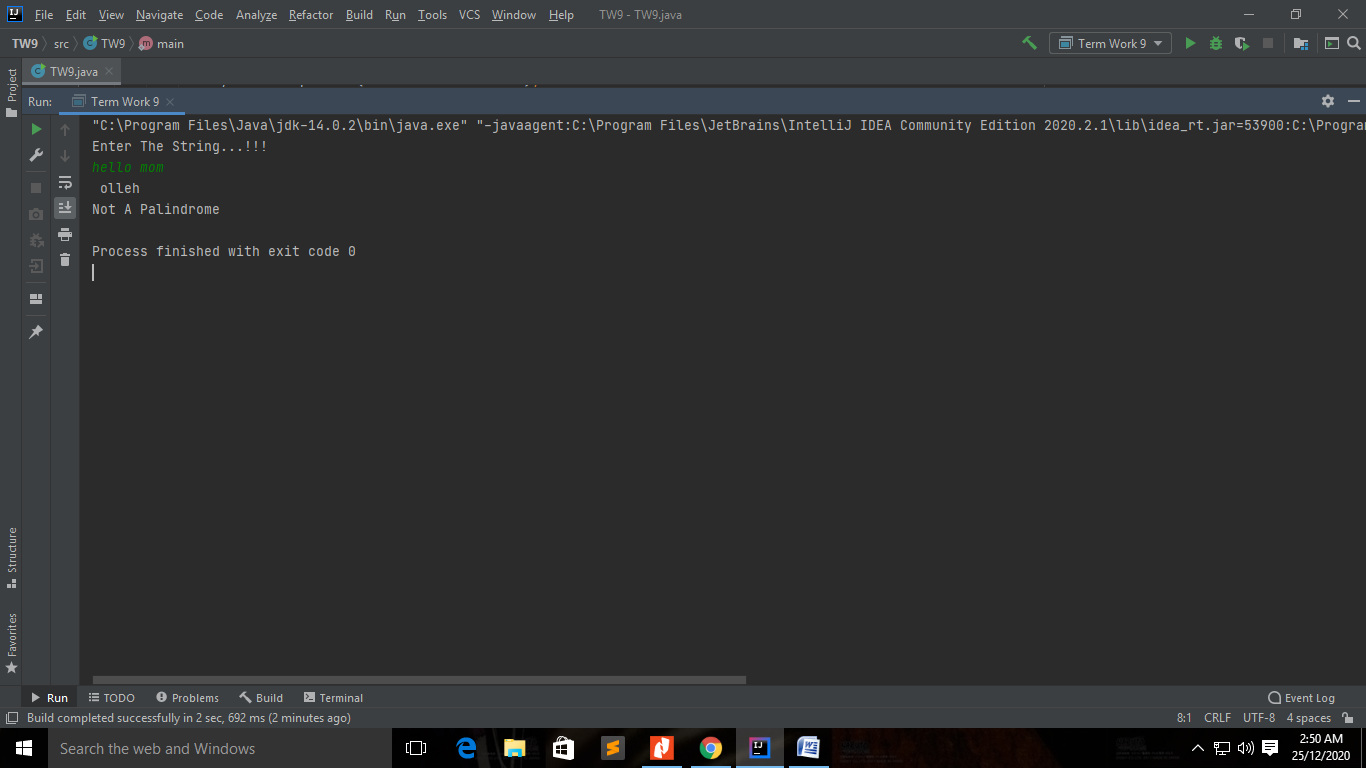
statement.

**CODE:**

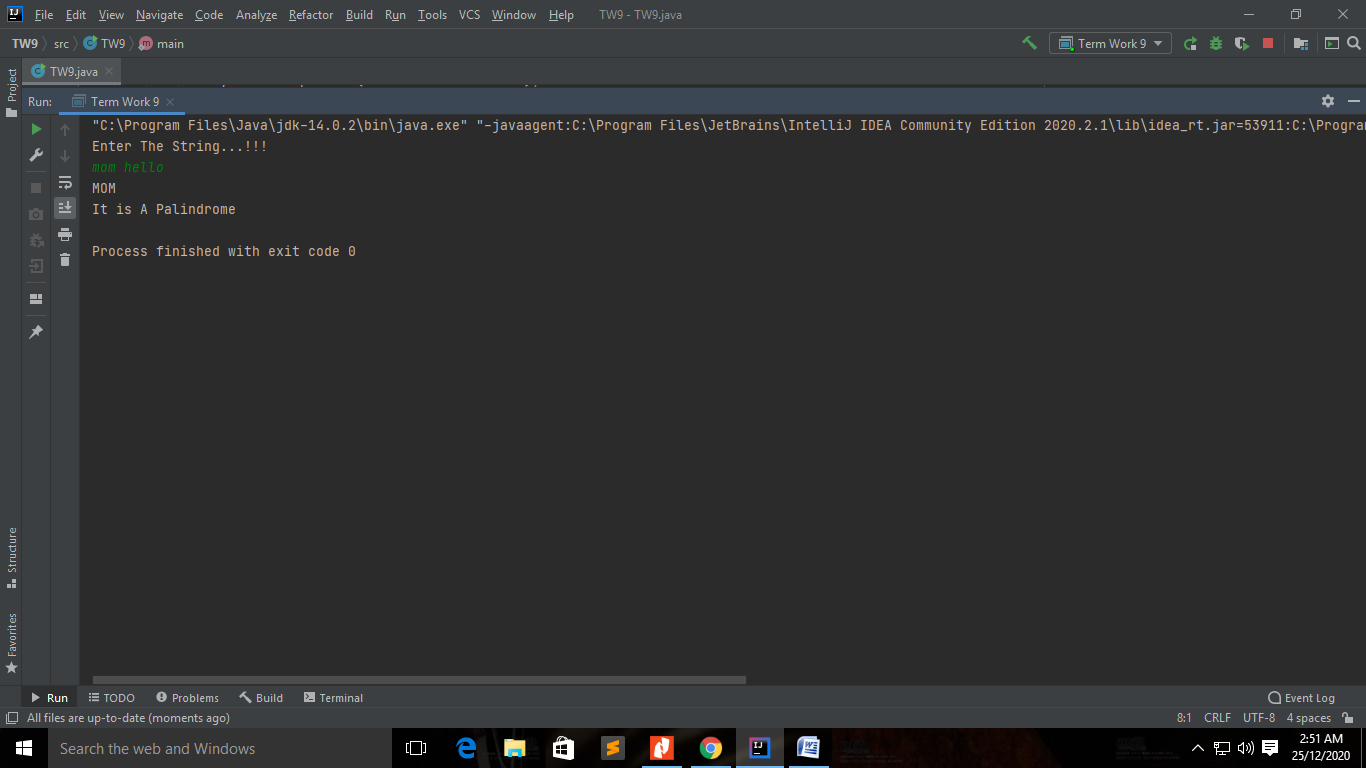
import java.util.Scanner;  
public class TW9 {  
 public static void main(String[] args) {  
 String str;  
 String[] words;  
 Scanner input = new Scanner(System.*in*);  
 System.*out*.println("Enter The String...!!!");  
 str = input.next();  
 words = str.split(" ");  
 for (String s : words) {  
 if (*isPalindrome*(s, 0, s.length() - 1)) {  
 System.*out*.println(s.toUpperCase());  
 System.*out*.println("It is A Palindrome");  
 } else {  
 System.*out*.println(*reverseString*(s).toLowerCase());  
 System.*out*.println("Not A Palindrome");  
 }  
 }  
 }  
 public static boolean isPalindrome (String word,int s, int t){  
 if (word.charAt(s) == word.charAt(t)) {  
 if (s < t)  
 return *isPalindrome*(word, s + 1, t - 1);  
 else if (s == t || s == t + 1)  
 return true;  
 }  
 return false;  
 }  
  
 public static String reverseString(String s){  
 String rs = " ";  
 for (int i = s.length() - 1; i >= 0; i--)  
 rs = rs + s.charAt(i);  
 return rs;  
 }  
 }

**OUTPUT:**

Case 1:



Case 2:



**HARSHINI KULKARNI**

**2GI20CS044**

**20/12/2021**

**TERMWORK 9.3**

9.3) Write a Java program that creates a simple book database (use an array of N objects).

Each book is represented with a ID, title, author (First Name &amp; last name), Genre (category –

technical, Sci Fi, Fiction, Comedy etc) and a Publisher name. Define methods to perform the

following tasks:

1. Given a title, returns a status to indicate whether or not the book exists in database.

2. Given a string “str”, lists the details of all the books whose title contains str.

3. Given a genre, lists publishers who have published books in that genre.

4. Given a character “c”, lists all authors who name starts with “c”.

**Code:**

class book{

int id;

String title, author, genre, publisher;

book(int id, String title, String author, String genre, String publisher ){

this.id = id;

this.title = title;

this.author = author;

this.genre = genre;

this.publisher = publisher;

}

}

public class TW9 {

public static void main(String[] args){

book[] b = new book[4];

b[0] = new book(101, "Java Programming", "james Gosling", "Technical", "PHI");

b[1] = new book(102, "Java Made Easy", "Dennis Ritchie", "Technical", "MIT");

b[2] = new book(103, "Programming with C", "Balaguruswamy", "Technical", "VTU");

b[3] = new book(104, "Programming with C++", "Balaguruswamy", "Technical", "PHI");

if(checkTitle(b,"Fun with Python") == true)

System.out.println("The title exits in the database");

else

System.out.println("The title does not exits in the database");

listTitles(b, "Java");

checkGenre(b, "Comedy");

listAuthors(b, 'D');

}

static boolean checkTitle(book[] b , String title){

for(book b1: b)

if(b1.title.equals(title) == true)

return true;

return false;

}

static void listTitles(book[] b , String str){

int i=0;

System.out.println(str + " appears in the following titles: ");

for(book b1: b){

if(b1.title.indexOf(str) != -1){

System.out.println(b1.title);

i++;

}

}

if(i==0)

System.out.println("Not Found");

}

static void checkGenre(book[] b , String genre){

int i=0;

System.out.println(" Following publishers have published books on " + genre + ":");

for(book b1: b){

if(b1.genre.equals(genre) == true){

System.out.println(b1.publisher);

i++;

}

}

if(i==0)

System.out.println("Not Found");

}

static void listAuthors(book[] b , char c){

int i=0;

System.out.println(" Auhtors whose names start with " + c + ":");

for(book b1: b){

if(b1.author.charAt(0) == c){

System.out.println(b1.author);

i++;

}

}

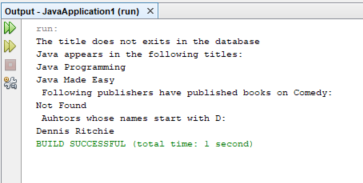
if(i==0)

System.out.println("Not Found");

}

}

**OUTPUT:**



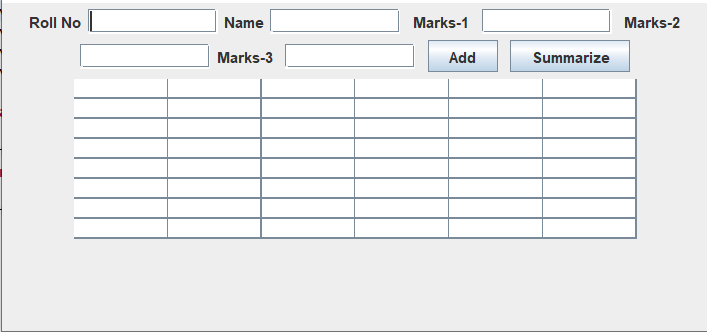
**HARSHINI KULKARNI**

**2GI20CS044**

**03/01/2022**

**TERMWORK-10**

Design and develop a GUI application to accepts student details as shown in the GUI below. Allow the user to add records one after the other. Once the user clicks on close, compute the average score and display the details using JTable component.



# Code:

package termwork\_10\_pp1; import java.util.ArrayList; import java.awt.EventQueue; import javax.swing.JFrame; import java.awt.FlowLayout; import javax.swing.JLabel; import javax.swing.JTextField; import javax.swing.JTable;

import javax.swing.table.DefaultTableModel; import javax.swing.JButton; import java.awt.event.ActionListener; import java.awt.event.ActionEvent;

public class GUI3 extends javax.swing.JFrame{ class StudentRecord{

String name;

int rollNo, m1, m2, m3;

StudentRecord(int rollNo, String name, int m1, int m2, int m3){ this.name = name;

this.rollNo = rollNo; this.m1 = m1; this.m2 = m2; this.m3 = m3;

}

}

ArrayList<StudentRecord> records = new ArrayList<StudentRecord>(); private JFrame frame;

private JTextField textField; private JTextField textField\_1; private JTextField textField\_2; private JTextField textField\_3; private JTextField textField\_4;

private JTable table;

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) { EventQueue.invokeLater(new Runnable() { public void run() {

try {

GUI3 window = new GUI3(); window.frame.setVisible(true);

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

}

}

});

}

/\*\*

\* Create the application.

\*/

public GUI3() { initialize();

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() { frame = new JFrame();

frame.setBounds(100, 100, 450, 300); frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); frame.getContentPane().setLayout(new FlowLayout(FlowLayout.CENTER, 5, 5));

JLabel jbl1 = new JLabel("Roll No"); frame.getContentPane().add(jbl1);

textField = new JTextField(); frame.getContentPane().add(textField); textField.setColumns(10);

JLabel jbl2 = new JLabel("Name"); frame.getContentPane().add(jbl2); textField\_1 = new JTextField(); frame.getContentPane().add(textField\_1); textField\_1.setColumns(10);

JLabel label = new JLabel(""); frame.getContentPane().add(label);

JLabel jbl3 = new JLabel("Marks-1"); frame.getContentPane().add(jbl3); JLabel label\_1 = new JLabel(""); frame.getContentPane().add(label\_1);

textField\_2 = new JTextField(); frame.getContentPane().add(textField\_2); textField\_2.setColumns(10);

JLabel label\_2 = new JLabel(""); frame.getContentPane().add(label\_2); JLabel jbl4 = new JLabel("Marks-2"); frame.getContentPane().add(jbl4);

textField\_3 = new JTextField(); frame.getContentPane().add(textField\_3); textField\_3.setColumns(10);

JLabel jbl5 = new JLabel("Marks-3"); frame.getContentPane().add(jbl5); JLabel label\_3 = new JLabel(""); frame.getContentPane().add(label\_3);

textField\_4 = new JTextField(); frame.getContentPane().add(textField\_4); textField\_4.setColumns(10);

JButton b1 = new JButton("Add");

b1.addActionListener(new ActionListener() { public void actionPerformed(ActionEvent e) {

StudentRecord s1 = new StudentRecord(Integer.parseInt(textField.getText()), textField\_1.getText(), Integer.parseInt(textField\_2.getText()), Integer.parseInt(textField\_3.getText()), Integer.parseInt(textField\_4.getText()));

records.add(s1); textField.setText(""); textField\_1.setText(""); textField\_2.setText(""); textField\_3.setText(""); textField\_4.setText("");

}

});

JLabel label\_4 = new JLabel(""); frame.getContentPane().add(label\_4); frame.getContentPane().add(b1);

JButton b2 = new JButton("Summarize"); b2.addActionListener(new ActionListener() { public void actionPerformed(ActionEvent e) { int rowCount = 0;

for(StudentRecord s : records) { table.setValueAt(s.rollNo, rowCount, 0);

table.setValueAt(s.name, rowCount, 1);

table.setValueAt(s.m1, rowCount, 2);

table.setValueAt(s.m2, rowCount, 3);

table.setValueAt(s.m3, rowCount, 4); float avg = (s.m1 + s.m2 + s.m3) / 3.0f;

table.setValueAt(String.format("%.2f", avg), rowCount, 5); rowCount++;

}

}

});

JLabel label\_5 = new JLabel(""); frame.getContentPane().add(label\_5); frame.getContentPane().add(b2);

table = new JTable(); table.setModel(new DefaultTableModel( new Object[][] {

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

{null, null, null, null, null, null},

},

new String[] {

"Roll No", "Name", "Marks 1", "Marks 2", "Marks 3", "Average"

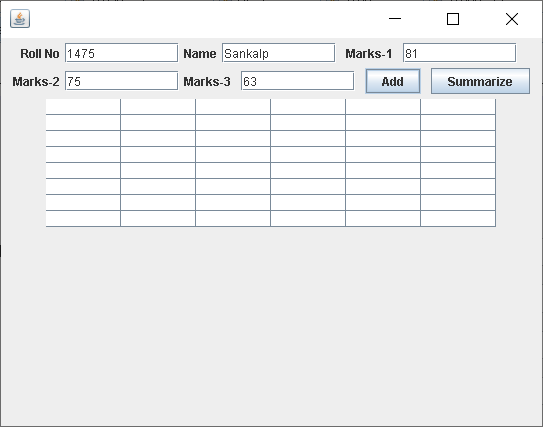
}

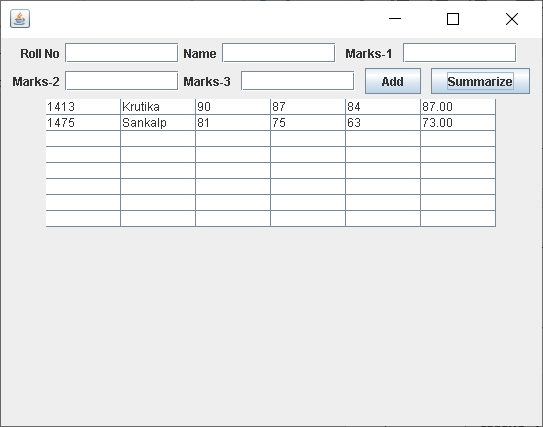
));

frame.getContentPane().add(table)

}

**INPUT AND OUTPUT:**





**HARSHINI KULKARNI**

**2GI20CS044**

**03/01/2022**

**TERMWORK 10.3**

10.3) Design and implement a Home loan Emi calculator using appropriate Swing

components. The GUI should like as under:

The formula to compute Home loan EMI for a given Principal amount PA and interest

rate IR for a period of T years is

EMI = ( PA + (PA \* IR \* T))/12\*T ;

Hint : Use SlideBars for Amount and Loan Period LoanType - ComboBox

**Code:**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class emiCalculator extends JFrame {

private JTextField jtfInvestmentAmount;

private JTextField jtfAnnualInterestRate;

private JTextField jtfNumberOfYears;

private JTextField jtfFutureValue;

private JButton jbtCompute;

private JButton jbtReset;

public emiCalculator() {

setTitle("Loan Calculator");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(500,300);

setLocationRelativeTo(null);

setLayout(new GridLayout(5,2,5,5));

JLabel jlInvestmentAmount = new JLabel(" Investment Amount");

JLabel jlNumberOfYears = new JLabel(" Number of Years");

JLabel jlAnnualInterestRate = new JLabel(" Annual Interest Rate");

JLabel jlFutureValue = new JLabel(" Future Value");

jtfInvestmentAmount = new JTextField();

jtfNumberOfYears = new JTextField();

jtfAnnualInterestRate = new JTextField();

jtfFutureValue = new JTextField();

jtfFutureValue.setEditable(false);

jbtCompute = new JButton("Compute");

jbtReset = new JButton("Reset");

add (jlInvestmentAmount);

add (jtfInvestmentAmount);

add (jlNumberOfYears);

add (jtfNumberOfYears);

add (jlAnnualInterestRate);

add (jtfAnnualInterestRate);

add (jlFutureValue);

add (jtfFutureValue);

add (jbtCompute);

add (jbtReset);

ListenerClass listener = new ListenerClass();

jbtCompute.addActionListener(listener);

jbtReset.addActionListener(listener);

setVisible(true);

}

public static void main(String[] args) {

new emiCalculator();

}

private void computeValue() {

try {

double annualInterestRate = Double.parseDouble(jtfAnnualInterestRate.getText());

double monthlyInterestRate = annualInterestRate / 1200.0;

int NumberOfYears = Integer.parseInt(jtfNumberOfYears.getText());

double investmentAmount = Double.parseDouble(jtfInvestmentAmount.getText());

double futureValue = investmentAmount \* Math.pow(1.0 + monthlyInterestRate, NumberOfYears \* 12);

jtfFutureValue.setText(String.format("%.2f", futureValue));

} catch (Exception e) {

JOptionPane.showMessageDialog(null, " Please enter numeric values.");

}

}

private void resetForm() {

jtfInvestmentAmount.setText("");

jtfAnnualInterestRate.setText("");

jtfNumberOfYears.setText("");

jtfFutureValue.setText("");

}

private class ListenerClass implements ActionListener {

public void actionPerformed(ActionEvent e) {

if (e.getSource() == jbtCompute) {

computeValue();

}

if (e.getSource() == jbtReset) {

resetForm();

}

}

}

}

**Output:**

