**OOP Lab Program 1 , 2**

1. **a) Design, Develop and Implement a Java program to calculate gross salary net salary taking the following data**

**DA=40%of basic**

**HRA=20%of basic**

**CCA=Rs250/-**

**PF=10%of basic**

**PT=Rs100/-**

**Income tax = 10% of gross**

**Gross income: Basic + DA + HRA + CCA**

**Deductions = PF+PT+IT**

**Net income = Gross income – Deductions**

import java.util.Scanner;

public class EmployeeSalary {

public static void main(String[] args) {

String name,id;

double bSalary, DA, HRA, PF, IT,grossIncome, netIncome, deductions;

double CCA = 250, PT = 100;

Scanner s = new Scanner(System.in);

System.out.println("Enter name of the employee");

name = s.nextLine();

System.out.println("Enter Employee ID");

id = s.nextLine();

System.out.println("Enter basic salary");

bSalary = s.nextDouble();

DA = (0.4)\*bSalary;

HRA = (0.2)\*bSalary;

PF = (0.1)\*bSalary;

grossIncome = bSalary + DA + HRA+CCA;

IT = (0.1)\*grossIncome;

deductions = PF+PT+IT;

netIncome = grossIncome - deductions;

System.out.println("The Gross income of employee "+name+" with ID "+id+" is "+grossIncome);

System.out.println("The Net income of employee "+name+" with ID "+id+" is "+netIncome);

s.close();

}

}

**1) b) Design, Develop and Implement a Java program that prints all real solutions to the quadratic equation ax2 + bx + c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.**

public class QuadraticEquation {

int a, b, c;

double root1, root2, d;

Scanner s = new Scanner(System.in);

void input()

{

System.out.println("Quadratic equation is in the form : ax^2 + bx + c");

System.out.print("Enter a:");

a = s.nextInt();

System.out.print("Enter b:");

b = s.nextInt();

System.out.print("Enter c:");

c = s.nextInt();

}

void discriminant() {

d= (b\*b)-(4\*a\*c);

}

void calculateRoots() {

if(d>0)

{

System.out.println("Roots are real and unequal");

root1 = ( - b + Math.sqrt(d))/(2\*a);

root2 = (-b - Math.sqrt(d))/(2\*a);

System.out.println("First root is:"+root1);

System.out.println("Second root is:"+root2);

}

else if(d == 0)

{

System.out.println("Roots are real and equal");

root1 = (-b+Math.sqrt(d))/(2\*a);

System.out.println("Root:"+root1);

}

else

{

System.out.println("No real solutions. Roots are imaginary");

}

}

}

public class TestQE {

public static void main(String[] args) {

QuadraticEquation qe = new QuadraticEquation();

qe.input();

qe.discriminant();

qe.calculateRoots();

}

}

**2). a) Design, Develop and Implement a Java program to add two given matrices using multidimensional arrays.**

import java.util.Scanner;

class Matrix

{

int m, n, p, q, sum = 0, i,j,k;

Scanner in = new Scanner(System.in);

int First[][] = new int[10][10];

int Second[][] = new int[10][10];

int Result[][] = new int[10][10];

void input()

{

System.out.println("Enter the number of rows and columns of First matrix");

m = in.nextInt();

n = in.nextInt();

System.out.println("Enter elements of First matrix");

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

{

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

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

}

System.out.println("Enter the number of rows and columns of Second matrix");

p = in.nextInt();

q = in.nextInt();

if (n != p)

{

System.out.println("The matrices can't be multiplied with each other.");

System.exit(0);

}

else

{

System.out.println("Enter the number of rows and columns of Second matrix");

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

{

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

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

}

}

}

void add()

{

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

{

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

{

Result[i][j]=First[i][j]+Second[i][j];

}

}

}

void display()

{

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

{

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

System.out.print(Result[i][j]+" ");

System.out.println();

}

}

}

public class Demo

{

public static void main(String args[])

{

Matrix MM=new Matrix();

MM.input();

MM.add();

MM.display();

}

}

**2) b) Design, Develop and Implement a Java program to add and subtract two complex numbers and using the concept of constructor overloading.**

public class ComplexNumber {

double real;

double imag;

ComplexNumber()

{

real = 0.0;

imag = 0.0;

}

ComplexNumber(double a)

{

real = a;

imag = 0;

}

ComplexNumber(double a, double b)

{

real = a;

imag = b;

}

ComplexNumber(ComplexNumber ob)

{

real = ob.real;

imag = ob.imag;

}

void add(ComplexNumber c1, ComplexNumber c2)

{

double realSum = c1.real+c2.real;

double imagSum = c1.imag+c2.imag;

System.out.println("Sum is "+realSum+"+i"+imagSum);

}

void sub(ComplexNumber c1, ComplexNumber c2)

{

double realDiff = c1.real-c2.real;

double imagDiff = c1.imag-c2.imag;

System.out.println("Difference is "+realDiff+"-i"+imagDiff);

}

}

public class ComplexNumberRun {

public static void main(String args[])

{

ComplexNumber cn1 = new ComplexNumber();

ComplexNumber cn2 = new ComplexNumber(10);

ComplexNumber cn3 = new ComplexNumber(20,30);

ComplexNumber cn4 = new ComplexNumber(cn3);

cn1.add(cn1,cn4);

cn2.sub(cn3,cn2);

}

}