PRAKHAR LOHIYA

16BCE0721

JAVA LAB 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ASSIGNMENT 5

1. Given a positive integer n. Write a program using java to print the pyramid pattern as described in the example below.

Input: n = 5

Output:

1

3\*2

4\*5\*6

10\*9\*8\*7

11\*12\*13\*14\*15

**CODE:**

public class Pyramid\_Pattern {

    static void printPattern(int n)

    {

        int j, k = 0;

        for (int i=1;i<=n;i++) {

            // if row number is odd

            if (i%2 != 0) {

                for (j=k+1;j<k+i;j++)

                    System.out.print(j + "\*");

                System.out.println(j++);

                k = j;

            }

            else {

                k = k+i-1;

                for (j=k;j>k-i+1;j--)

                    System.out.print(j + "\*");

                System.out.println(j);

            }

        }

    }

    public static void main(String args[])

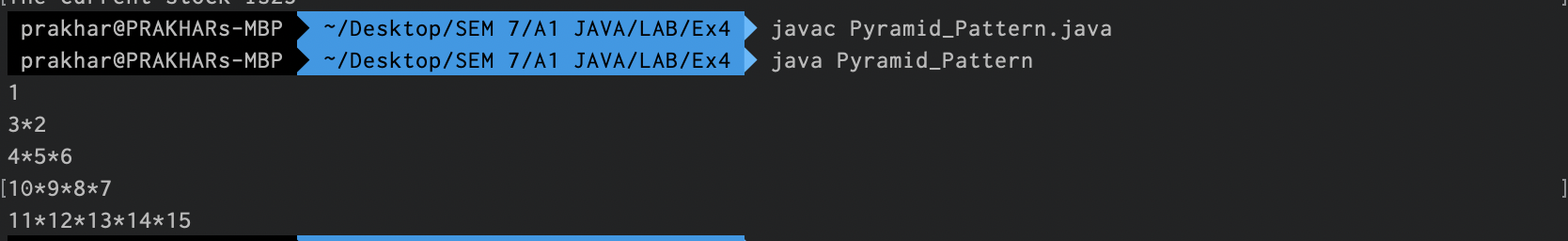
        {

            int n = 5;

            printPattern(n);

        }

}

**OUTPUT:**

2. Write a JAVA program to print the pattern given below by taking the user input (n).

If n = 4, the pattern is given below.

\*\*\*\*\*\*\*\*

\*\*\* \*\*\*

\*\* \*\*

\* \*

\*\* \*\*

\*\*\* \*\*\*

\*\*\*\*\*\*\*\*

**CODE:**

import java.util.Scanner;

class Pattern{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();

        for(int i=0;i<n;i++){

            for(int j=i;j<n;j++){

                System.out.print("\*");

            }

            for(int j=n-i;j<n;j++){

                System.out.print("  ");

            }

            for(int j=i;j<n;j++){

                System.out.print("\*");

            }

            System.out.println();

        }

        for(int i=1;i<n;i++){

            for(int j=n-i;j<=n;j++){

                System.out.print("\*");

            }

            for(int j=i+1;j<n;j++){

                System.out.print("  ");

            }

            for(int j=n-i;j<=n;j++){

                System.out.print("\*");

            }

            System.out.println();

        }

    }

}

**OUTPUT:**

3. The bank association has decided to automate the credit card application process in using common methods.

**CreditCardAPP**

getpersonaldetails();

calculatePerYearIncome();

PrintEligibility();

AXIS

HDFC

ICICI

CreditCardAPP Interface methods: - The methods inside the credit card app has to be redefined in individual bank classes.

* getpersonalDetails – Method receives the name, mob, address, nominee, age, email for a customer.
* CalculatePerYearIncome – Take input of gross salary.
  + NetSalPerMonth = GrossSalary – (GrossSalary \* 0.2)
  + PerYearIncome = NetSalPerMonth \*12
* PrintEligibility – This is decided based on previous loan EMI. Get the input of EMI that an employee pays per month
  + EligibilityScore = 3. If employee does not pays any EMI.
  + EligibilityScore= 2. If employee pays EMI for less than 20% of his perYearIncome.
  + EligibilityScore=1. If employee pays EMI for less than 40% of his perYearIncome.
  + EligibilityScore=0. If employee pays EMI for less than 60% of his perYearIncome.

Write a Java program that creates the banking class for individual banks like ICICI, HDFC and AXIS. Redefine the methods of the interfaces in all classes.

**CODE:**

import java.util.Scanner;

import javafx.scene.effect.MotionBlur;

interface CreditCardAPP{

    public void getpersonaldetails(String name, String mob, String add, String nominee, int age, String email);

    public void calculatePerYearIncome(double grossSalary);

    public void PrintEligibility(double EMI);

}

class ICICI implements CreditCardAPP{

    String name;

    String mob;

    String add;

    String nominee;

    int age;

    String email;

    double netSalary;

    double perYearIncome;

    int eligibilityScore;

    public void getpersonaldetails(String name, String mob, String add, String nominee, int age, String email){

        this.name = name;

        this.mob = mob;

        this.add = add;

        this.nominee = nominee;

        this.age = age;

        this.email = email;

    }

    public void calculatePerYearIncome(double grossSalary) {

        netSalary = grossSalary - (grossSalary\*0.2);

        perYearIncome = netSalary\*12;

        System.out.println("The annual income is "+perYearIncome);

        // return perYearIncome;

    }

    public void PrintEligibility(double EMI){

        EMI = EMI\*12;

        if(EMI==0)

            eligibilityScore = 3;

        else if(EMI>0 && EMI<=0.2\*perYearIncome)

            eligibilityScore = 2;

        else if(EMI>0.2\*perYearIncome && EMI <=0.4\*perYearIncome)

            eligibilityScore = 1;

        else

            eligibilityScore = 0;

        System.out.println("##### ICICI BANK #####");

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

        System.out.println("Mobile  :"+mob);

        System.out.println("Address :"+add);

        System.out.println("Nominee :"+nominee);

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

        System.out.println("E-mail  :"+email);

        System.out.println();

        System.out.println("Eligibility Score = "+eligibilityScore);

        System.out.println();

    }

}

class HDFC implements CreditCardAPP{

    String name;

    String mob;

    String add;

    String nominee;

    int age;

    String email;

    double netSalary;

    double perYearIncome;

    int eligibilityScore;

    public void getpersonaldetails(String name, String mob, String add, String nominee, int age, String email){

        this.name = name;

        this.mob = mob;

        this.add = add;

        this.nominee = nominee;

        this.age = age;

        this.email = email;

    }

    public void calculatePerYearIncome(double grossSalary) {

        netSalary = grossSalary - (grossSalary\*0.2);

        perYearIncome = netSalary\*12;

        System.out.println("The annual income is "+perYearIncome);

        // return perYearIncome;

    }

    public void PrintEligibility(double EMI){

        EMI = EMI\*12;

        if(EMI==0)

            eligibilityScore = 3;

        else if(EMI>0 && EMI<=0.2\*perYearIncome)

            eligibilityScore = 2;

        else if(EMI>0.2\*perYearIncome && EMI <=0.4\*perYearIncome)

            eligibilityScore = 1;

        else

            eligibilityScore = 0;

        System.out.println("##### HDFC BANK #####");

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

        System.out.println("Mobile  :"+mob);

        System.out.println("Address :"+add);

        System.out.println("Nominee :"+nominee);

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

        System.out.println("E-mail  :"+email);

        System.out.println();

        System.out.println("Eligibility Score = "+eligibilityScore);

        System.out.println();

    }

}

class AXIS implements CreditCardAPP{

    String name;

    String mob;

    String add;

    String nominee;

    int age;

    String email;

    double netSalary;

    double perYearIncome;

    int eligibilityScore;

    public void getpersonaldetails(String name, String mob, String add, String nominee, int age, String email){

        this.name = name;

        this.mob = mob;

        this.add = add;

        this.nominee = nominee;

        this.age = age;

        this.email = email;

    }

    public void calculatePerYearIncome(double grossSalary) {

        netSalary = grossSalary - (grossSalary\*0.2);

        perYearIncome = netSalary\*12;

        // System.out.println("The annual income is "+perYearIncome);

        // return perYearIncome;

    }

    public void PrintEligibility(double EMI){

        EMI = EMI\*12;

        if(EMI==0)

            eligibilityScore = 3;

        else if(EMI>0 && EMI<=0.2\*perYearIncome)

            eligibilityScore = 2;

        else if(EMI>0.2\*perYearIncome && EMI <=0.4\*perYearIncome)

            eligibilityScore = 1;

        else

            eligibilityScore = 0;

        System.out.println("##### AXIS BANK #####");

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

        System.out.println("Mobile  :"+mob);

        System.out.println("Address :"+add);

        System.out.println("Nominee :"+nominee);

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

        System.out.println("E-mail  :"+email);

        System.out.println();

        System.out.println("The annual income is "+perYearIncome);

        System.out.println();

        System.out.println("Eligibility Score = "+eligibilityScore);

        System.out.println();

    }

}

class Bank{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

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

        String name = sc.nextLine();

        System.out.println("Enter Mobile No: ");

        String mob = sc.nextLine();

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

        String add = sc.nextLine();

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

        String nominee = sc.nextLine();

        System.out.println("Enter Age ");

        int age = Integer.parseInt(sc.nextLine());

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

        String email = sc.nextLine();

        System.out.println("Enter Gross Salary: ");

        double grossSalary = sc.nextDouble();

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

        double EMI = sc.nextDouble();

        ICICI i = new ICICI();

        HDFC h = new HDFC();

        AXIS a = new AXIS();

        i.getpersonaldetails(name, mob, add, nominee, age, email);

        h.getpersonaldetails(name, mob, add, nominee, age, email);

        a.getpersonaldetails(name, mob, add, nominee, age, email);

        i.calculatePerYearIncome(grossSalary);

        h.calculatePerYearIncome(grossSalary);

        a.calculatePerYearIncome(grossSalary);

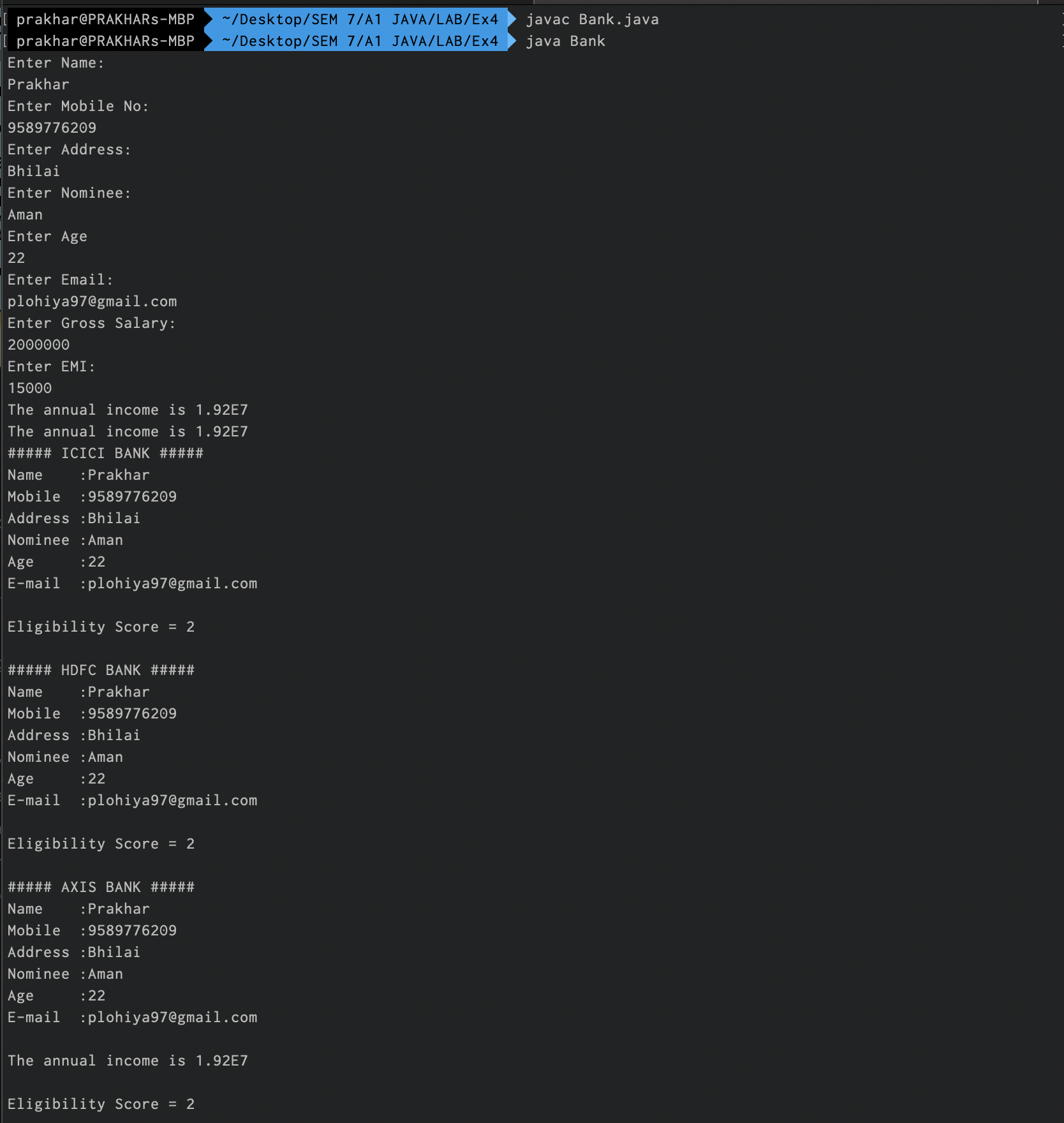
        i.PrintEligibility(EMI);

        h.PrintEligibility(EMI);

        a.PrintEligibility(EMI);

    }

}

**OUTPUT:**

1. A super market has various products for sale. Design an abstract class to have methods like billing, stock availability for a specific product, printSaleReport. Create the subclasses according to category of products like, cosmetics, groceries, kitchenTools, snacks & chocolates. Redefine the methods in abstract class. Apply the dynamic polymorphism concept and override the printSaleReport method of the abstract class.

**CODE:**

abstract class SuperMarket{

    abstract void billing();

    abstract void stock();

    abstract void printSaleReport();

}

class Cosmetics extends SuperMarket{

    int count\_c = 25;

    void billing(){

        System.out.println("The per-piece price of cosmetics is 100");

        // return 100;

    }

    void stock(){

        count\_c--;

        System.out.println("The current stock is"+count\_c);

    }

    void printSaleReport(){

        System.out.println("Products Sold: 1");

        System.out.println("Price: 100");

    }

}

class Groceries extends SuperMarket{

    int count\_g = 100;

    void billing(){

        System.out.println("The cost of groceries is 500");

        // return 100;

    }

    void stock(){

        count\_g = count\_g-5;

        System.out.println("The current stock is"+count\_g);

    }

    void printSaleReport(){

        System.out.println("Products Sold: 5");

        System.out.println("Price: 500");

    }

}

class KitchenTools extends SuperMarket{

    int count\_k = 40;

    void billing(){

        System.out.println("The cost of groceries is 200");

        // return 100;

    }

    void stock(){

        count\_k = count\_k--;

        System.out.println("The current stock is"+count\_k);

    }

    void printSaleReport(){

        System.out.println("Products Sold: 1");

        System.out.println("Price: 200");

    }

}

class SnacksAndChocolate extends SuperMarket{

    int count\_snc = 500;

    void billing(){

        System.out.println("The cost of groceries is 20");

        // return 100;

    }

    void stock(){

        count\_snc = count\_snc--;

        System.out.println("The current stock is"+count\_snc);

    }

    void printSaleReport(){

        System.out.println("Products Sold: 1");

        System.out.println("Price: 20");

    }

}

class Market{

    public static void main(String[] args) {

        SuperMarket c = new Cosmetics();

        SuperMarket g = new Groceries();

        SuperMarket k = new KitchenTools();

        SuperMarket s = new SnacksAndChocolate();

        // Cosmetics

        c.billing();

        c.printSaleReport();

        c.stock();

        // Groceries

        g.billing();

        g.printSaleReport();

        g.stock();

        // Kitchen Tools

        k.billing();

        k.printSaleReport();

        k.stock();

        // Snacks And Choclates

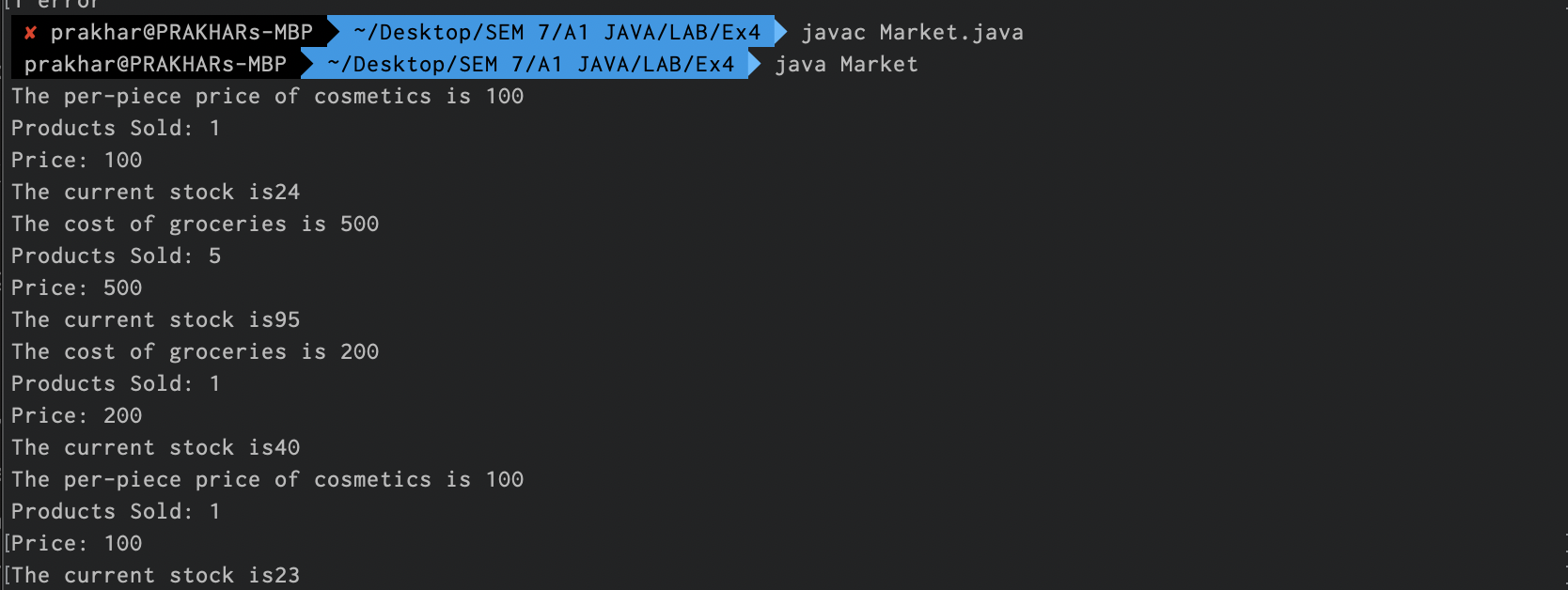
        c.billing();

        c.printSaleReport();

        c.stock();

    }

}

**OUTPUT:**

1. Design a class to display the schedule of trains in MGR Central railway station. The class can have its own member variable like, train\_no, src, dest, time, traveltime, platformno, traveltype (A-Arrival, D-Departure). Create an array of objects in main function. Perform the following tasks.
   1. SearchFunction ( ) – Takes trainno has input and perfroms a search with all objects. If found display all variable details of the class like train\_no, src, dest, time, traveltime, platformno, traveltype. If not found throw arrayIndexOutofBound exception and handle it.
   2. SortFunction ( ) – Takes the input of sorting type (TM- time based, TN – Train no based). Sort and display the records. If the user enters other than TN or TM input throw arithematic or arrayIndexOutofBound exception and handle it.

**CODE:**

import java.util.Scanner;

class Train{

    int train\_no;

    String src, dest;

    int time;

    int travelTime;

    int platform\_no;

    String travelType; // A or D

    Train(int train\_no, String src, String dest, int time, int travelTime, int platform\_no, String travelType){

        this.train\_no = train\_no;

        this.src = src;

        this.dest = dest;

        this.time = time;

        this.travelTime = travelTime;

        this.platform\_no = platform\_no;

        this.travelType = travelType;

    }

}

class MGRStation{

    public static void SearchFunction(Train[] obj, int number){

        int count = 0, t=0;

        try{

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

                if(number == obj[i].train\_no){

                    count = 1;

                    t=i;

                    break;

                }

            }

            if(count == 1){

                System.out.println("######## Train Found ########");

                System.out.println("Train no    : "+obj[t].train\_no);

                System.out.println("Source      : "+obj[t].src);

                System.out.println("Destination : "+obj[t].dest);

                System.out.println("Time        : "+obj[t].time);

                System.out.println("Duration    : "+obj[t].travelTime);

                System.out.println("Platform no : "+obj[t].platform\_no);

                System.out.println("Travel Type : "+obj[t].travelType);

            }

            else{

                throw new ArrayIndexOutOfBoundsException("Train Not Found");

            }

        }

        catch(ArrayIndexOutOfBoundsException e){

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

        }

    }

    public static void main(String []args){

        Scanner sc = new Scanner(System.in);

        Train t1 = new Train(12435, "Katpadi", "Chennai", 1000, 2, 10, "A");

        Train t2 = new Train(22679, "Chennai", "Bhilai", 1600, 24, 2, "D");

        Train t3 = new Train(12636, "Chennai", "Hyderabad", 4000, 12, 1, "D");

        Train obj[] = new Train[3];

        obj[0] = t1;

        obj[1] = t2;

        obj[2] = t3;

        System.out.println("Enter Train number");

        int search = Integer.parseInt(sc.nextLine());

        SearchFunction(obj, search);

    }

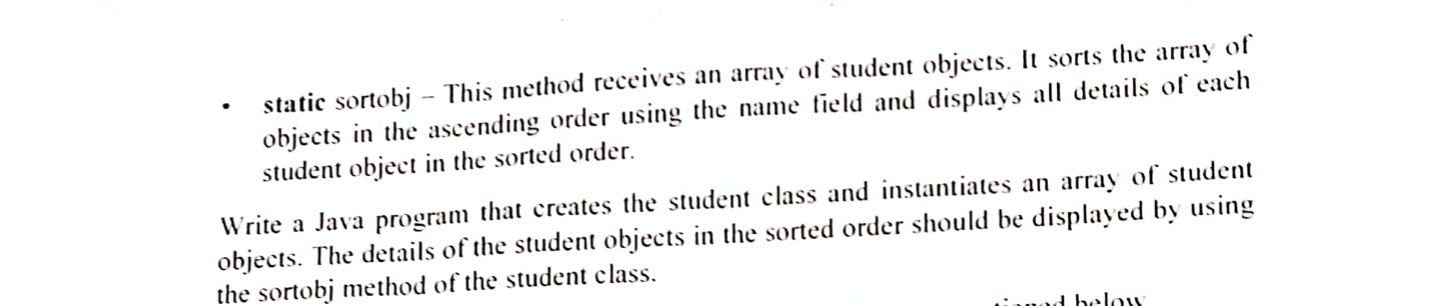
}

**OUTPUT:**

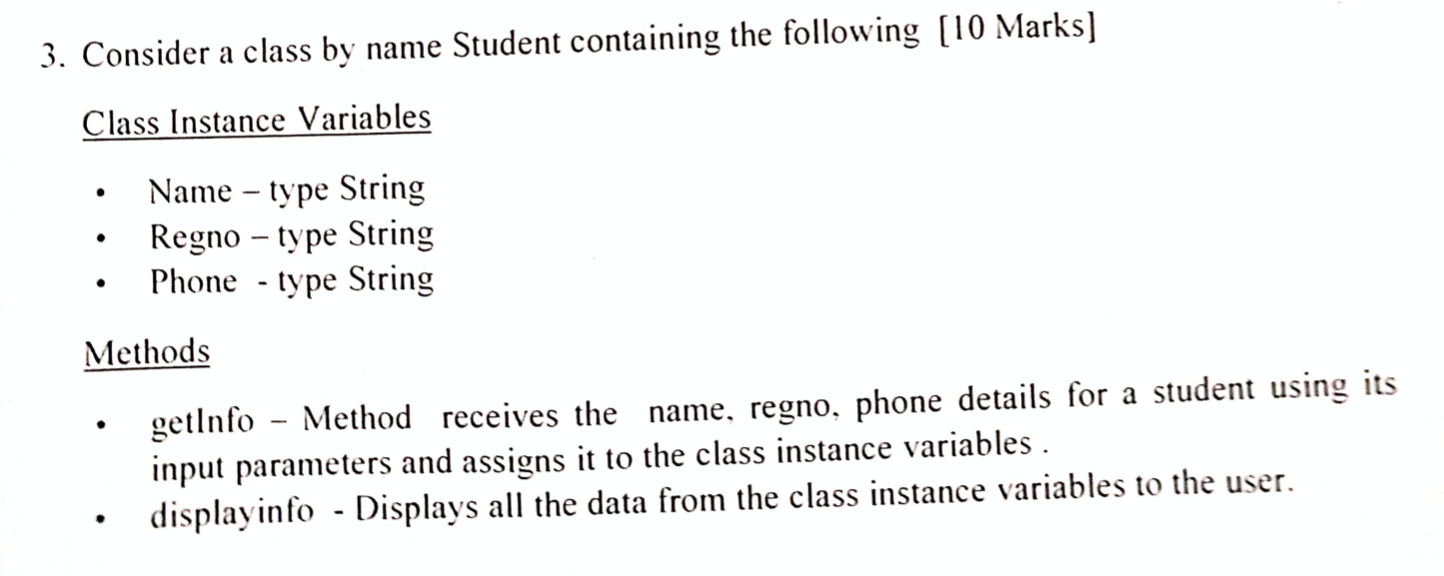
PAPER 2:

**CODE:**

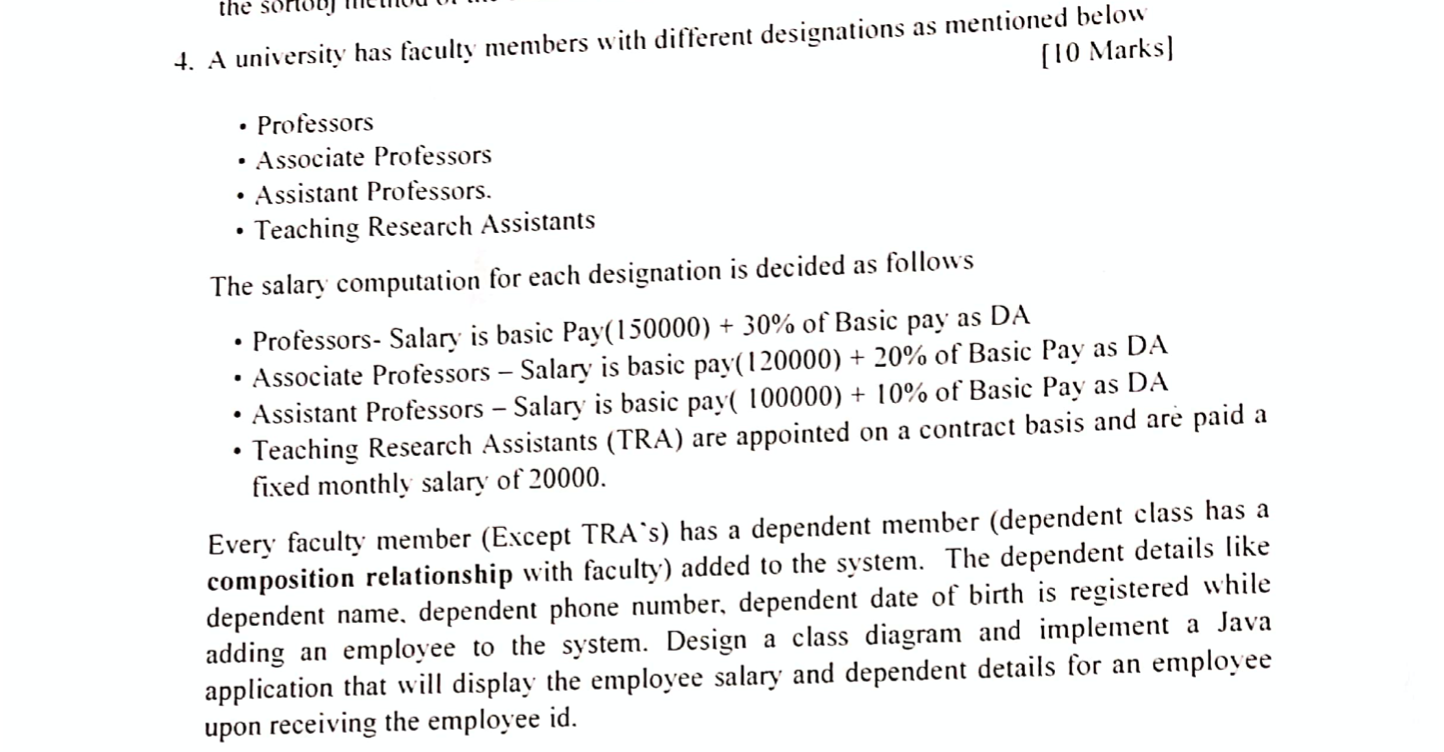
**OUTPUT:**

****

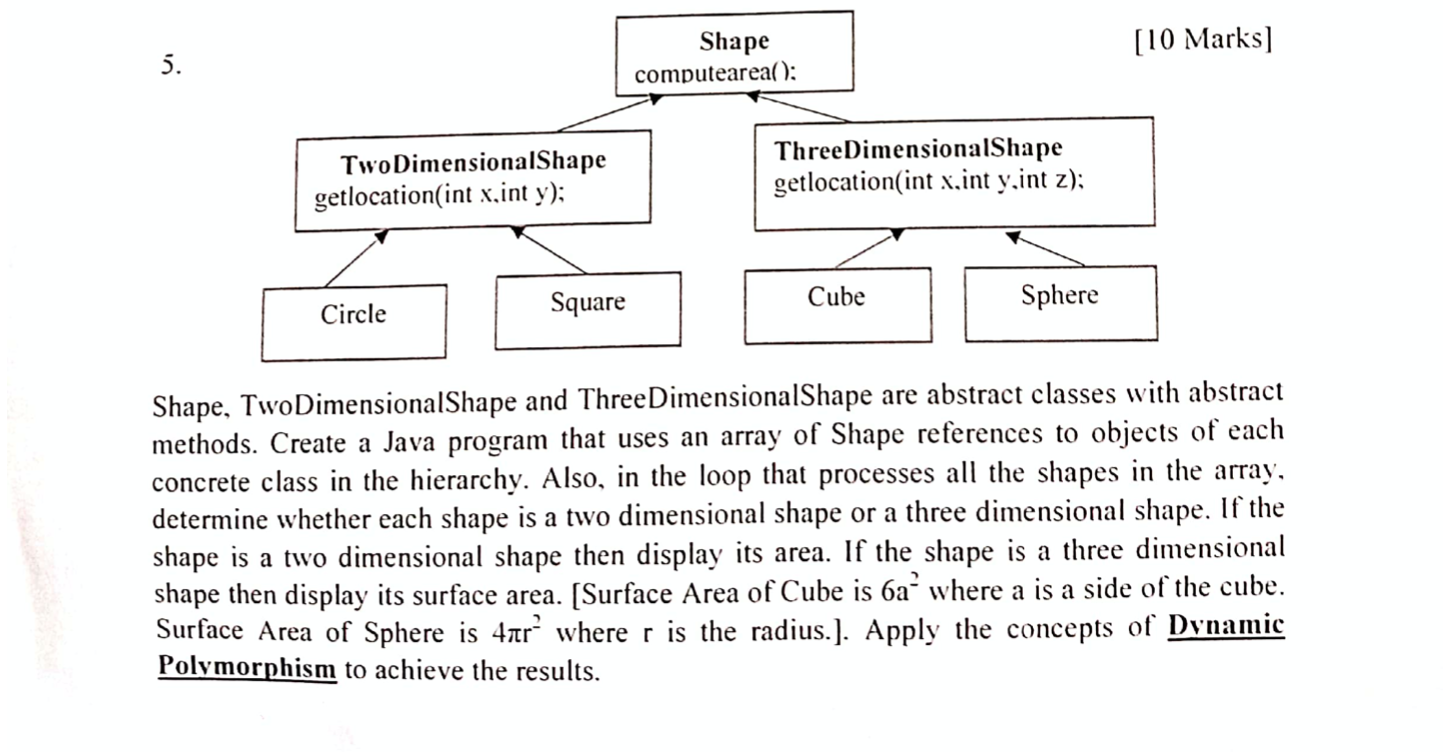
**CODE:**



**OUTPUT:**

****

**CODE:**

**OUTPUT:**

**CODE:**

**OUTPUT:**