Rajalakshmi Engineering College

Name: SUBASH R
Email: 240701538@rajalakshmi.edu.in
Roll no: 240701538
Phone: 9150202710
Branch: REC
Department: CSE - Section 7
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 5_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Arjun is working as a developer for CityWater Supply Board, which wants to build a household water billing system.

Each household's water account has:

A Customer ID (integer)A Customer Name (string)Liters Consumed (double)

The water bill is calculated based on these rules:

For the first 500 liters 2 per literFor the next 500 liters (501–1000) 3 per literFor liters above 1000 5 per literIf the total bill exceeds 3000, a 10% discount is applied on the final bill.

Arjun has been asked to implement this system using:

A class with attributes for customer details.A constructor to initialize customer details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer's details and final bill amount.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the Customer ID (integer).
- The following line contains the Customer Name (string).
- The next line contains the Liters Consumed (double).

Output Format

For each customer, print the details in the following format:

Customer ID: <customer_id>

Customer Name: <customer_name>

Final Bill: <final_bill> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
300
Output: Customer ID: 1001
Customer Name: Ravi Kumar
Final Bill: 600.0

Answer

```
import java.util.Scanner;
```

```
class Account {
    private int cid;
    private String name;
    private double lit;
    public Account(int cid, String name, double lit) {
        this.cid = cid;
        this.name = name;
        this.lit = lit;
    }
    public int getCid() {
        return cid;
    }
    public String getName() {
        return name;
    }
    public double calcBill() {
        double bill=0;
        if(lit>1000) bill+=(500*2)+(500*3)+((lit-1000)*5);
        else if(lit>500) bill+=(500*2)+((lit-500)*3);
        else if(lit>0) bill+=lit*2;
        if(bill>3000) bill-=bill*0.1;
        return bill;
    }
}

public class Main{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int count = in.nextInt();
        in.nextLine();
        for (int i = 0; i < count; i++) {
            int cid = in.nextInt();
            in.nextLine();
            String name = in.nextLine();
            double lit = in.nextDouble();
            Account acc = new Account(cid, name, lit);
            System.out.println("Customer ID: " + acc.getCid());
            System.out.println("Customer Name: " + acc.getName());
            System.out.println("Final Bill: " + acc.calcBill());
        }
    }
```

}

Status : Correct **Marks : 10/10**

2. Problem Statement

Anjali is now working as a developer for the City Marathon Association, which wants to build a system to track and find the fastest runner among marathon participants.

Each runner’s record has:

Runner ID (integer)Runner Name (string)An array of times (in minutes) taken in 5 marathon events (integers)

The system must calculate:

The average time of each runner (sum of all times / 5).Identify the fastest runner (the one with the lowest average time).If two or more runners have the same average time, the one with the lower Runner ID is considered the fastest runner.

Anjali has been asked to implement this system using:

A class with attributes for runner details.A constructor to initialize runner details.Getter and Setter methods to retrieve and update runner details if required.A method to calculate the average time.Objects of the class to represent runners.

Finally, display each runner’s details and announce the Fastest Runner.

Input Format

The first line of input contains an integer N (number of runners).

For each runner:

- The next line contains the Runner ID (integer).
- The following line contains the Runner Name (string).
- The next line contains 5 integers separated by spaces (times in minutes for 5 marathon events).

Output Format

For each runner the output prints the following details:

- Runner ID: <runner_id>
- Runner Name: <runner_name>
- Average Time: <average_time>

Finally, print "Fastest Runner: <runner_name> with <average_time> minutes"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
240 250 245 255 260
Output: Runner ID: 1001
Runner Name: Ravi Kumar
Average Time: 250
Fastest Runner: Ravi Kumar with 250 minutes

Answer

```
import java.util.*;

class Runner {
    private int Id;
    private String Name;
    private int[] times;
    public Runner(int Id, String Name, int[] times) {
        this.Id = Id;
        this.Name = Name;
        this.times = times;
    }
    public int getId() {
        return Id;
```

```
    }
    public String getName() {
        return Name;
    }
    public double getAverage() {
        int sum = 0;
        for (int t : times) {
            sum += t;
        }
        return sum / 5.0;
    }
}

public class Main{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        Runner[] runners = new Runner[n];
        for (int i = 0; i < n; i++) {
            int id = in.nextInt();
            in.nextLine();
            String name = in.nextLine();
            int[] times = new int[5];
            String[] parts = in.nextLine().split(" ");
            for (int j = 0; j < 5; j++) {
                times[j] = Integer.parseInt(parts[j]);
            }
            runners[i] = new Runner(id, name, times);
        }
        Runner fastest = runners[0];
        for (Runner r : runners) {
            System.out.println("Runner ID: " + r.getId());
            System.out.println("Runner Name: " + r.getName());
            System.out.println("Average Time: " + (int) r.getAverage());
            if (r.getAverage() < fastest.getAverage() || (r.getAverage() ==
fastest.getAverage() && r.getId() < fastest.getId())) {
                fastest = r;
            }
        }
        System.out.println("Fastest Runner: " + fastest.getName() + " with " + (int)
fastest.getAverage() + " minutes");
    }
```

}

Status : Correct ***Marks : 10/10***

3. Problem Statement

Meera is working as a developer for CityGas Supply Board, which wants to build a household gas billing system.

Each household’s gas account has:

A Customer ID (integer)A Customer Name (string)Units Consumed in cubic meters (double)

The gas bill is calculated based on these rules:

For the first 50 units 4 per unitFor the next 100 units (51–150) 6 per unitFor units above 150 8 per unitIf the total bill exceeds 2000, a 15% discount is applied on the final bill.

Meera has been asked to implement this system using:

A class with attributes for customer details.A constructor to initialize customer details.Setter methods to update details if needed.Getter methods to retrieve details.Objects of the class to represent customers.

Finally, display each customer’s details and final bill amount.

Input Format

The first line of input contains an integer N, representing the number of customers.

For each customer:

- The next line contains the Customer ID (integer).
- The following line contains the Customer Name (string).
- The next line contains the Units Consumed (double).

Output Format

For each customer, print the details in the following format:

Customer ID: <customer_id>

Customer Name: <customer_name>

Final Bill: <final_bill> (The final bill must be rounded to one decimal place.)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
30
Output: Customer ID: 1001
Customer Name: Ravi Kumar
Final Bill: 120.0

Answer

```
import java.util.Scanner;

class Account {
   private int cid;
   private String name;
   private double unit;
   public Account(int cid, String name, double unit) {
      this.cid = cid;
      this.name = name;
      this.unit = unit;
   }
   public int getCid() {
      return cid;
   }
   public String getName() {
      return name;
   }
   public double calcBill() {
      double bill=0;
      if(unit>150) bill+=(50*4)+(100*6)+((unit-150)*8);
```

```
        else if(unit>50) bill+=(50*4)+((unit-50)*6);
        else if(unit>0) bill+=unit*4;
        if(bill>2000) bill-=bill*0.15;
        return bill;
    }
}

public class Main{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int count = in.nextInt();
        in.nextLine();
        for (int i = 0; i < count; i++) {
            int cid = in.nextInt();
            in.nextLine();
            String name = in.nextLine();
            double unit = in.nextDouble();
            Account acc = new Account(cid, name, unit);
            System.out.println("Customer ID: " + acc.getCid());
            System.out.println("Customer Name: " + acc.getName());
            System.out.println("Final Bill: " + acc.calcBill());
        }
    }
}
```

Status : Correct **Marks : 10/10**

4. Problem Statement

Anjali is working as a developer for the City Basketball Association, which wants to build a system to track and find the top scorer among basketball players.

Each player's record has:

Player ID (integer)Player Name (string)An array of points scored in 5 matches (integers)

The system must calculate:

The total score of each player (sum of all match points).Identify the highest scorer among all players.If two or more players have the same total score, the one with the lower Player ID is considered the top scorer.

Anjali has been asked to implement this system using:

A class with attributes for player details.A constructor to initialize player details.Getter and Setter methods to retrieve and update player details if required.A method to calculate the total score.Objects of the class to represent players.

Finally, display each player's details and announce the Top Scorer.

Input Format

The first line of input contains an integer N (number of players).

For each player:

- The next line contains the Player ID (integer).
- The following line contains the Player Name (string).
- The next line contains 5 integers separated by spaces (points scored in 5 matches).

Output Format

For each player the output prints the following details:

- Player ID: <player_id>
- Player Name: <player_name>
- Total Score: <total_score>

Finally, print "Top Scorer: <player_name> with <total_score> points"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
10 20 30 40 50

Output: Player ID: 1001
Player Name: Ravi Kumar
Total Score: 150
Top Scorer: Ravi Kumar with 150 points

Answer

```
import java.util.*;

class Player {
    private int Id;
    private String Name;
    private int[] scores;
    public Player(int Id, String Name, int[] scores) {
        this.Id = Id;
        this.Name = Name;
        this.scores = scores;
    }
    public int getId() {
        return Id;
    }
    public String getName() {
        return Name;
    }
    public int getSum() {
        int sum = 0;
        for (int s : scores) {
            sum += s;
        }
        return sum;
    }
}

public class Main{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        Player[] players = new Player[n];
        for (int i = 0; i < n; i++) {
```

```
            int id = in.nextInt();
            in.nextLine();
            String name = in.nextLine();
            int[] scores = new int[5];
            String[] parts = in.nextLine().split(" ");
            for (int j = 0; j < 5; j++) {
                scores[j] = Integer.parseInt(parts[j]);
            }
            players[i] = new Player(id, name, scores);
        }
        Player topper = players[0];
        for (Player r : players) {
            System.out.println("Player ID: " + r.getId());
            System.out.println("Player Name: " + r.getName());
            System.out.println("Total Score: " + r.getSum());
            if (r.getSum() > topper.getSum() || (r.getSum() == topper.getSum() &&
r.getId() < topper.getId())) {
                topper = r;
            }
        }
        System.out.println("Top Scorer: " + topper.getName() + " with " +
topper.getSum() + " points");
    }
}
```

Status : Correct **Marks : 10/10**