Rajalakshmi Engineering College

Name: Punith Kumar.S
Email: 241501155@rajalakshmi.edu.in
Roll no: 241501155
Phone: 9600149411
Branch: REC
Department: AI & ML - Section 3
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are working as a developer for CityCab, a taxi service company that wants to build a ride fare management system.

Each customer booking has:

A Booking ID (integer)A Customer Name (string)A Distance Travelled in km (double)

The fare calculation rules are:

Base Fare = 50 units (flat charge for every ride).Per km charge = 10 units/km.If the distance is greater than 20 km, a 10% discount is applied on the total fare.

You are required to implement this system using:

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

Finally, display each booking's details and final fare.

Input Format

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

For each booking:

- The next line contains the booking ID (integer).
- The following line contains the customer's name (string).
- The next line contains the distance travelled (double).

Output Format

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

1. Booking ID: <booking_id>
2. Customer Name: <customer_name>
3. Final Fare: <final_fare> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Rahul Sharma
15

Output: Booking ID: 1234
Customer Name: Rahul Sharma
Final Fare: 200.0

Answer

```
import java.util.*;

class Booking {
```

```
    private int id;
    private String name;
    private double distance;

    Booking(int id, String name, double distance) {
        this.id = id;
        this.name = name;
        this.distance = distance;
    }

    public int getId() { return id; }
    public String getName() { return name; }
    public double getDistance() { return distance; }

    public double calculateFare() {
        double fare = 50 + distance * 10;
        if (distance > 20) fare -= fare * 0.1;
        return fare;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());
        for (int i = 0; i < n; i++) {
            int id = Integer.parseInt(sc.nextLine());
            String name = sc.nextLine();
            double distance = Double.parseDouble(sc.nextLine());
            Booking b = new Booking(id, name, distance);
            System.out.println("Booking ID: " + b.getId());
            System.out.println("Customer Name: " + b.getName());
            System.out.printf("Final Fare: %.1f\n", b.calculateFare());
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Punith Kumar.S
Email: 241501155@rajalakshmi.edu.in
Roll no: 241501155
Phone: 9600149411
Branch: REC
Department: AI & ML - Section 3
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Neha is working as a developer for CityElectricity Board, which wants to build a household electricity billing system.

Each customer's electricity account has:

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

The electricity bill is calculated based on these rules:

For the first 100 units 5 units charge per unitFor the next 100 units (101–200) 7 units charge per unitFor units above 200 10 units charge per unitIf the total bill exceeds 2000 units, a 5% discount is applied on the final bill.

Neha 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> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1001
Ravi Kumar
80
Output: Customer ID: 1001
Customer Name: Ravi Kumar
Final Bill: 400.0

Answer

```
import java.util.*;
```

```
class Customer {
    private int id;
    private String name;
    private double units;

    Customer(int id, String name, double units) {
        this.id = id;
        this.name = name;
        this.units = units;
    }

    public int getId() { return id; }
    public String getName() { return name; }
    public double getUnits() { return units; }

    public double calculateBill() {
        double bill = 0;
        if (units <= 100) bill = units * 5;
        else if (units <= 200) bill = 100 * 5 + (units - 100) * 7;
        else bill = 100 * 5 + 100 * 7 + (units - 200) * 10;
        if (bill > 2000) bill -= bill * 0.05;
        return bill;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());
        for (int i = 0; i < n; i++) {
            int id = Integer.parseInt(sc.nextLine());
            String name = sc.nextLine();
            double units = Double.parseDouble(sc.nextLine());
            Customer c = new Customer(id, name, units);
            System.out.println("Customer ID: " + c.getId());
            System.out.println("Customer Name: " + c.getName());
            System.out.printf("Final Bill: %.1f\n", c.calculateBill());
        }
    }
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Punith Kumar.S
Email: 241501155@rajalakshmi.edu.in
Roll no: 241501155
Phone: 9600149411
Branch: REC
Department: AI & ML - Section 3
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 5_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are working as a developer for CityBank, which wants to build a basic account management system.

Each customer at the bank has:

An Account Number (integer)A Customer Name (string)An Initial Balance (double)

The bank allows two types of transactions:

Deposit – increases the balance.Withdrawal – decreases the balance only if enough funds are available.

If the withdrawal amount is greater than the balance, the withdrawal should not happen, and the balance should remain the same.

You are required to implement this system using:

A class with attributes for account details.A constructor to initialize account 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 account details after all transactions.

Input Format

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

For each customer:

- The next line contains the account number (integer).
- The following line contains the customer name (string).
- The next line contains the initial balance (double).
- The next line contains the deposit amount (double).
- The next line contains the withdrawal amount (double).

Output Format

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

1. Account Number: <account_number>
2. Customer Name: <customer_name>
3. Final Balance: <final_balance> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Rahul Sharma
5000
2000
3000
Output: Account Number: 1234
Customer Name: Rahul Sharma

Final Balance: 4000.0

Answer

```
import java.util.*;

class Account {
   private int accNo;
   private String name;
   private double balance;

   public Account(int accNo, String name, double balance) {
      this.accNo = accNo;
      this.name = name;
      this.balance = balance;
   }

   public int getAccNo() {
      return accNo;
   }
   public String getName() {
      return name;
   }
   public double getBalance() {
      return balance;
   }

   public void setName(String name) {
      this.name = name;
   }
   public void setBalance(double balance) {
      this.balance = balance;
   }

   public void deposit(double amount) {
      if (amount >= 0) balance += amount;
   }

   public void withdraw(double amount) {
      if (amount <= balance) balance -= amount;
   }

   public void display() {
```

```
        System.out.println("Account Number: " + accNo);
        System.out.println("Customer Name: " + name);
        System.out.println("Final Balance: " + String.format("%.1f", balance));
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < n; i++) {
            int accNo = sc.nextInt();
            sc.nextLine();
            String name = sc.nextLine();
            double balance = sc.nextDouble();
            double deposit = sc.nextDouble();
            double withdraw = sc.nextDouble();
            Account acc = new Account(accNo, name, balance);
            acc.deposit(deposit);
            acc.withdraw(withdraw);
            acc.display();
        }
        sc.close();
    }
}
```

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