Rajalakshmi Engineering College

Name: Noothan H C
Email: 241501137@rajalakshmi.edu.in
Roll no: 241501137
Phone: 8217612225
Branch: REC
Department: AI & ML - Section 4
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_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Ram is working as a developer for BrightEdu Coaching Center, which wants to build a student fee management system.

Each student’s enrollment has:

An Enrollment ID (integer)A Student Name (string)The Number of Subjects (integer)

The fee calculation rules are:

Registration Fee = 1000 units (flat for every student).Per Subject Fee = 800 units.If the student enrolls in more than 5 subjects, a 20% scholarship (discount) is applied on the total fee.

Ram has been asked to implement this system using:

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

Finally, display each student's details and final fee.

Input Format

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

For each student:

- The next line contains the Enrollment ID (integer).
- The following line contains the student's name (string).
- The next line contains the Number of subjects (integer).

Output Format

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

- Enrollment ID: <enrollment_id>
- Student Name: <student_name>
- Final Fee: <final_fee> (rounded to one decimal place)

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Ravi Kumar
3
Output: Enrollment ID: 1234
Student Name: Ravi Kumar
Final Fee: 3400.0

Answer

```
import java.util.*;

class Student {
    private int enrollmentId;
```

```
    private String studentName;
    private int numSubjects;

    public Student(int enrollmentId, String studentName, int numSubjects) {
        this.enrollmentId = enrollmentId;
        this.studentName = studentName;
        this.numSubjects = numSubjects;
    }

    public void setEnrollmentId(int enrollmentId) {
        this.enrollmentId = enrollmentId;
    }

    public void setStudentName(String studentName) {
        this.studentName = studentName;
    }

    public void setNumSubjects(int numSubjects) {
        this.numSubjects = numSubjects;
    }

    public int getEnrollmentId() {
        return enrollmentId;
    }

    public String getStudentName() {
        return studentName;
    }

    public int getNumSubjects() {
        return numSubjects;
    }

    public double calculateFee() {
        double fee = 1000 + (numSubjects * 800);
        if (numSubjects > 5) {
            fee = fee - (fee * 0.20);
        }
        return fee;
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = Integer.parseInt(sc.nextLine().trim());
        for (int i = 0; i < N; i++) {
            int enrollmentId = Integer.parseInt(sc.nextLine().trim());
            String studentName = sc.nextLine().trim();
            int numSubjects = Integer.parseInt(sc.nextLine().trim());

            Student student = new Student(enrollmentId, studentName,
numSubjects);
            System.out.printf("Enrollment ID: %d\nStudent Name: %s\nFinal Fee: %.1f
\n",
                    student.getEnrollmentId(), student.getStudentName(),
student.calculateFee());
        }
        sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Noothan H C
Email: 241501137@rajalakshmi.edu.in
Roll no: 241501137
Phone: 8217612225
Branch: REC
Department: AI & ML - Section 4
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 bookingId;
    private String customerName;
    private double distance;

    public Booking(int bookingId, String customerName, double distance) {
        this.bookingId = bookingId;
        this.customerName = customerName;
        this.distance = distance;
    }

    public void setBookingId(int bookingId) {
        this.bookingId = bookingId;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public void setDistance(double distance) {
        this.distance = distance;
    }

    public int getBookingId() {
        return bookingId;
    }

    public String getCustomerName() {
        return customerName;
    }

    public double getDistance() {
        return distance;
    }

    public double calculateFare() {
        double fare = 50 + (distance * 10);
        if (distance > 20) {
            fare = fare - (fare * 0.10);
        }
        return fare;
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = Integer.parseInt(sc.nextLine().trim());
        for (int i = 0; i < N; i++) {
            int bookingId = Integer.parseInt(sc.nextLine().trim());
            String customerName = sc.nextLine().trim();
            double distance = Double.parseDouble(sc.nextLine().trim());

            Booking booking = new Booking(bookingId, customerName, distance);
            System.out.printf("Booking ID: %d\nCustomer Name: %s\nFinal Fare: %.1f
\n",
                    booking.getBookingId(), booking.getCustomerName(),
booking.calculateFare());
        }
        sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Noothan H C
Email: 241501137@rajalakshmi.edu.in
Roll no: 241501137
Phone: 8217612225
Branch: REC
Department: AI & ML - Section 4
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 customerId;
    private String customerName;
    private double unitsConsumed;

    public Customer(int customerId, String customerName, double
unitsConsumed) {
        this.customerId = customerId;
        this.customerName = customerName;
        this.unitsConsumed = unitsConsumed;
    }

    public void setCustomerId(int customerId) {
        this.customerId = customerId;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public void setUnitsConsumed(double unitsConsumed) {
        this.unitsConsumed = unitsConsumed;
    }

    public int getCustomerId() {
        return customerId;
    }

    public String getCustomerName() {
        return customerName;
    }

    public double getUnitsConsumed() {
        return unitsConsumed;
    }

    public double calculateBill() {
        double bill = 0;
        double units = unitsConsumed;

        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 - (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().trim());
        for (int i = 0; i < N; i++) {
            int id = Integer.parseInt(sc.nextLine().trim());
            String name = sc.nextLine().trim();
            double units = Double.parseDouble(sc.nextLine().trim());
            Customer c = new Customer(id, name, units);
            System.out.printf("Customer ID: %d\nCustomer Name: %s\nFinal Bill: %.1f
\n",
                    c.getCustomerId(), c.getCustomerName(), c.calculateBill());
        }
        sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Noothan H C
Email: 241501137@rajalakshmi.edu.in
Roll no: 241501137
Phone: 8217612225
Branch: REC
Department: AI & ML - Section 4
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 accountNumber;
    private String customerName;
    private double balance;

    public Account(int accountNumber, String customerName, double balance) {
        this.accountNumber = accountNumber;
        this.customerName = customerName;
        this.balance = balance;
    }

    public void setAccountNumber(int accountNumber) {
        this.accountNumber = accountNumber;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

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

    public int getAccountNumber() {
        return accountNumber;
    }

    public String getCustomerName() {
        return customerName;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount >= 0) {
```

```
            balance += amount;
        }
    }

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = Integer.parseInt(sc.nextLine().trim());
        for (int i = 0; i < N; i++) {
            int accountNumber = Integer.parseInt(sc.nextLine().trim());
            String customerName = sc.nextLine().trim();
            double initialBalance = Double.parseDouble(sc.nextLine().trim());
            double depositAmount = Double.parseDouble(sc.nextLine().trim());
            double withdrawalAmount = Double.parseDouble(sc.nextLine().trim());

            Account acc = new Account(accountNumber, customerName,
initialBalance);
            acc.deposit(depositAmount);
            acc.withdraw(withdrawalAmount);

            System.out.printf("Account Number: %d\nCustomer Name: %s\nFinal
Balance: %.1f\n",
                    acc.getAccountNumber(), acc.getCustomerName(),
acc.getBalance());
        }
        sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: Noothan H C
Email: 241501137@rajalakshmi.edu.in
Roll no: 241501137
Phone: 8217612225
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 4_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a secure banking system, customers are required to create PIN codes for accessing their accounts. The bank wants to validate these PIN codes before accepting them.

A PIN code is considered valid if:

It consists of exactly 4 digits.All characters must be numeric (0–9).It cannot contain all identical digits (e.g., 1111 is invalid).

Your task is to determine whether each PIN code in the list is valid or not.

Input Format

The first line of input contains an integer T, representing the number of PIN codes to check.

The next T lines each contain a string S, representing a PIN code.

Output Format

For each PIN code S, the output print "YES" if it is valid.

Otherwise, the output print "NO".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
1234
Output: YES

Answer

```
// You are using Java
import java.util.*;
class hello{
   public static void main(String[] args){
      Scanner sc=new Scanner(System.in);
      int t=Integer.parseInt(sc.nextLine());
      for(int i=0;i<t;i++)
      {
         String pin=sc.nextLine().trim();
         if(isValid(pin)){
            System.out.println("YES");
         }else{
            System.out.println("NO");
         }
      }
   }
   public static boolean isValid(String pin){
      if(pin.length()!=4){
         return false;
      }
      for(char ch: pin.toCharArray()){
         if(!Character.isDigit(ch)){
            return false;
```

```
        }
    }
    char first=pin.charAt(0);
    boolean allSame=true;
    for(int i=1;i<4;i++){
        if(pin.charAt(i)!=first){
            allSame=false;
            break;
        }
    }
    if(allSame){
        return false;
    }
    return true;
}
}
```

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