

# Rajalakshmi Engineering College

Name: Nandhini Velmurugan  
Email: 241901063@rajalakshmi.edu.in  
Roll no: 241901063  
Phone: 9043367699  
Branch: REC  
Department: CSE (CS) - Section 1  
Batch: 2028  
Degree: B.E - CSE (CS)

Scan to verify results



## 2024\_28\_III\_OOPS Using Java Lab

### **REC\_2028\_OOPS using Java\_Week 7\_CY**

Attempt : 1  
Total Mark : 40  
Marks Obtained : 40

#### **Section 1 : Coding**

##### **1. Problem Statement**

John is developing a car loan calculator and has structured his program using two interfaces, Principal and InterestRate, defining methods for principal and interest rate retrieval.

The Loan class implements these interfaces, taking principal and annual interest rates as parameters. User input is solicited for these values, and the program ensures their validity before performing calculations. If input values are invalid (less than or equal to zero), an error message is displayed.

Note: Total interest = principal \* interest rate \* years

##### ***Input Format***

The first line of input consists of a double value P, representing the principal.

The second line consists of a double value R, representing the annual interest rate.

The third line consists of an integer value N, representing the loan duration in years.

### ***Output Format***

If the input values are valid, print "Total interest paid: Rs. " followed by a double value, representing the total interest paid, rounded off to two decimal places.

If the input values are invalid (negative or zero values for principal, annual interest rate, or loan duration), print "Invalid input values!".

Refer to the sample output for formatting specifications.

### ***Sample Test Case***

Input: 20000.00

0.05

5

Output: Total interest paid: Rs.5000.00

### ***Answer***

```
import java.util.Scanner;  
interface interestRate{  
    double getInterestRate();  
}  
  
interface Principal{  
    double getPrincipal();  
}  
  
class Loan implements Principal, interestRate{  
    private double principal;  
    private double interestRate;  
    public Loan(double principal, double interestRate){
```

```

        this.principal = principal;
        this.interestRate = interestRate;
    }

    public double getPrincipal(){
        return principal;
    }
    public double getInterestRate(){
        return interestRate;
    }
    public double calculateTotalInterest(int years) {
        return principal * interestRate * years;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double carPrice = scanner.nextDouble();

        double annualInterestRate = scanner.nextDouble();

        int loanDuration = scanner.nextInt();

        if (carPrice <= 0 || annualInterestRate <= 0 || loanDuration <= 0) {
            System.out.println("Invalid input values!");
            return;
        }

        Loan carLoan = new Loan(carPrice, annualInterestRate);
        double totalInterest = carLoan.calculateTotalInterest(loanDuration);

        System.out.printf("Total interest paid: Rs.%2f%n", totalInterest);
    }
}

```

**Status :** Correct

**Marks :** 10/10

## 2. Problem Statement:

Sam is developing a geometry application and needs a class for trapezoid calculations. Create a "Trapezoid" class implementing a "ShapeInput" interface with a method to input trapezoid dimensions.

Also, implement a "ShapeCalculator" interface with methods to compute area and perimeter. In the "Main" class, instantiate Trapezoid, gather user input, and display the calculated area and perimeter with two decimal places.

#### Note

Area of Trapezoid =  $(1/2) * (\text{base1} + \text{base2}) * \text{height}$

Perimeter of Trapezoid =  $\text{base1} + \text{base2} + \text{side1} + \text{side2}$

#### *Input Format*

The first line of input is a double-point value representing base1 of the trapezoid.

The second line of input is a double-point value representing base2 of the trapezoid.

The third line of input is a double-point value representing the height of the trapezoid.

The fourth line of input is a double-point value representing side1 of the trapezoid.

The fifth line of input is a double-point value representing side2 of the trapezoid.

#### *Output Format*

The output displays the two lines of the calculated area (double type) and perimeter (double type) of the trapezoid, each rounded to two decimal places in the following format:

"Area of the Trapezoid: <<calculated area>>".

Perimeter of the Trapezoid: <<calculated perimeter>>".

Refer to the sample output for the formatting specifications.

### **Sample Test Case**

Input: 1.0

2.0

1.0

3.0

1.0

Output: Area of the Trapezoid: 1.50

Perimeter of the Trapezoid: 7.00

### **Answer**

```
import java.util.Scanner;  
  
interface ShapeInput {  
    void getInput();  
}  
  
interface ShapeCalculator {  
    double calculateArea();  
    double calculatePerimeter();  
}  
  
class Trapezoid implements ShapeInput, ShapeCalculator {  
    double base1, base2, height, side1, side2;  
  
    public void getInput() {  
        Scanner scan = new Scanner(System.in);  
        base1 = scan.nextDouble();  
        base2 = scan.nextDouble();  
        height = scan.nextDouble();  
        side1 = scan.nextDouble();  
        side2 = scan.nextDouble();  
    }  
  
    public double calculateArea() {  
        return 0.5 * (base1 + base2) * height;  
    }  
}
```

```

        public double calculatePerimeter() {
            return base1 + base2 + side1 + side2;
        }
    }

    public class Main {
        public static void main(String[] args) {
            Trapezoid trapezoid = new Trapezoid();
            trapezoid.getInput();

            double area = trapezoid.calculateArea();
            double perimeter = trapezoid.calculatePerimeter();

            System.out.println("Area of the Trapezoid: " + String.format("%.2f", area));
            System.out.println("Perimeter of the Trapezoid: " + String.format("%.2f",
perimeter));
        }
    }
}

```

**Status :** Correct

**Marks :** 10/10

### 3. Problem Statement:

Rathish is planning a road trip and needs a program to convert speeds between miles per hour (MPH) and kilometers per hour (KPH).

Create an interface, SpeedConverter, with a method convertSpeed(double mph). Implement the interface with MPHtoKPHConverter class, allowing Rathish to input MPH and receive the converted speed in KPH, rounded to two decimal points.

Formula: Speed in KPH = 1.60934 \* Speed in MPH.

#### ***Input Format***

The input consists of a single double-point number representing the speed in miles per hour (MPH).

#### ***Output Format***

The output displays the converted speed (double-point number) in kilometers per hour (KPH) rounded off to two decimal points in the following format:

"Speed in KPH: <<converted speed>>".

Refer to the sample output for the formatting specifications.

### ***Sample Test Case***

Input: 1.0

Output: Speed in KPH: 1.61

### ***Answer***

```
import java.util.Scanner;

interface SpeedConverter{
    double convertSpeed(double mph);
}

class MPHtoKPHConverter implements SpeedConverter{
    public double convertSpeed(double mph){
        double kph = 1.60934 * mph;
        return Math.round(kph*100.0)/100.0;
    }
}

class SpeedConversionApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        double speedInMPH = scanner.nextDouble();

        SpeedConverter converter = new MPHtoKPHConverter();

        double speedInKPH = converter.convertSpeed(speedInMPH);

        System.out.printf("Speed in KPH: %.2f\n", speedInKPH);

        scanner.close();
    }
}
```

**Status : Correct**

**Marks : 10/10**

#### 4. Problem Statement:

Ray is developing a tax calculation program in Java. The program includes an interface named TaxCalculator with a method to calculate tax based on salary. The SimpleTaxCalculator class implements this interface and determines the tax to be paid based on the salary amount using progressive tax slabs.

Your task is to implement this system. The program first takes an integer T representing the number of test cases, followed by T salary values. For each salary, calculate the total tax to be paid based on the following progressive tax rules:

For the first 50,000 of salary, the tax rate is 5%. For the next 50,000 (i.e., from 50,001 to 1,00,000), the tax rate is 10%. For any amount above 1,00,000, the tax rate is 20%. (That is, only the amount above 1,00,000 is taxed at 20%).

#### Example

##### Input

3

78000

110000

23000

##### Output

5300

9500

1150

#### Explanation

For Salary Rs. 78,000

$$\text{Tax} = 0.1 * (78,000 - 50,000) + 0.05 * 50,000 = 5,300$$

For Salary Rs. 1,10,000

$$\text{Tax} = 0.2 * (110000 - 100000) + 0.1 * 50,000 + 0.05 * 50,000 = 9,500$$

For Salary Rs. 23,000

$$\text{Tax} = 0.05 * 23,000 = 1,150$$

### ***Input Format***

The first line of the input consists of an integer, T, representing the number of test cases.

The next T lines of the input consist of a single integer, representing the annual salary of an individual, separated by a line.

### ***Output Format***

The output displays the calculated tax as an integer for each test case, separated by a line.

Refer to the sample output for the formatting specifications.

### ***Sample Test Case***

Input: 2

100

300

Output: 5

15

### ***Answer***

```
import java.util.Scanner;  
  
interface TaxCalculator {  
    int calculateTax(int salary);  
}  
  
class SimpleTaxCalculator implements TaxCalculator {  
    public int calculateTax(int salary) {  
        double tax = 0;  
        if (salary <= 50000) {  
            tax = 0.05 * salary;  
        } else if (salary <= 100000) {  
            tax = 0.1 * salary;  
        } else if (salary <= 200000) {  
            tax = 0.2 * salary;  
        } else {  
            tax = 0.2 * (100000 - 50000) + 0.1 * (50000 - 20000) + 0.05 * (salary - 200000);  
        }  
        return (int) tax;  
    }  
}
```

```
        } else if (salary <= 100000) {
            tax = 0.05 * 50000 + 0.10 * (salary - 50000);
        } else {
            tax = 0.05 * 50000 + 0.10 * 50000 + 0.20 * (salary - 100000);
        }
        return (int) tax;
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int T = scanner.nextInt();

        TaxCalculator taxCalculator = new SimpleTaxCalculator();

        for (int i = 0; i < T; i++) {
            int salary = scanner.nextInt();
            int tax = taxCalculator.calculateTax(salary);
            System.out.println(tax);
        }

        scanner.close();
    }
}
```

**Status :** Correct

**Marks :** 10/10