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 7_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Maria, an online store owner, is looking to implement a pricing system that calculates the final price of products after applying discounts. She needs a program that takes the original price of a product and the discount percentage as input and computes the final discounted price. The discount is applied as a percentage of the original price. Maria wants to ensure that the final price is formatted to display exactly two decimal places.

Implement this functionality using the PriceCalculator interface and the DiscountCalculator class.

Input Format

The first line of input consists of the original price (a double value).

The second line of input consists of a discount percentage (a double value).

Output Format

The output displays the final price after the discount, adhering to the following format: "Final Price after discount: $[final_price]".

Here, [final_price] should be replaced with the calculated final price, formatted as a currency value with two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 100.0
10.0

Output: Final Price after discount: $90.00

Answer

```
import java.util.Scanner;

interface PriceCalculator {
   double calculatePrice(double originalPrice, double discount);
}

class DiscountCalculator implements PriceCalculator {
   public double calculatePrice(double originalPrice, double discount) {
      double discountedPrice = originalPrice - (originalPrice * discount / 100);
      return discountedPrice;
   }
}

class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      double originalPrice = scanner.nextDouble();
      double discount = scanner.nextDouble();
      PriceCalculator calculator = new DiscountCalculator();
      double finalPrice = calculator.calculatePrice(originalPrice, discount);
      System.out.printf("Final Price after discount: $%.2f%n", finalPrice); // Formats output to 2 decimal places
   }
```

}

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

2. Problem Statement

A developer aims to create a budget management system using two interfaces, ExpenseRecorder for recording expenses and BudgetCalculator for calculating remaining budgets.

The ExpenseTracker class implements these interfaces, allowing users to input an initial budget and record expenses iteratively until entering 0.0 as a sentinel value.

The program then computes and displays the remaining budget or notifies of budget exceedance.

Example

Input

100.0

20.0 30.0 10.0 0.0

Output

Remaining budget: Rs. 40.00

Explanation

The initial budget is 100.0. Expenses of 20.0, 30.0, and 10.0 are recorded.

Remaining budget is calculated (100.0 - 20.0 - 30.0 - 10.0 = 40.0).

Input Format

The first line of input is the initial budget as a double-point number (double type). The budget is a positive number.

The second line of input consists of individual expenses as double-point numbers. Each expense is separated by space.

To end the input, an expense of 0.0 is used.

Output Format

The output displays the remaining budget, formatted to two decimal places, in the following format:

If the remaining budget (double type) is non-negative, it prints "Remaining budget: Rs. [remainingBudget]".

If the remaining budget is negative, it prints "No remaining budget, You've exceeded your budget!".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 100.0
20.0 30.0 10.0 0.0

Output: Remaining budget: Rs. 40.00

Answer

```
import java.util.Scanner;

interface ExpenseRecorder {
   void recordExpense(double expense);
}

interface BudgetCalculator {
   double calculateRemainingBudget();
}
class ExpenseTracker implements ExpenseRecorder, BudgetCalculator {
   private double budget;
   private double totalExpenses;

   public ExpenseTracker(double budget) {
      this.budget = budget;
```

```
        this.totalExpenses = 0.0;
    }

    public void recordExpense(double expense) {
        totalExpenses += expense;
    }

    public double calculateRemainingBudget() {
        return budget - totalExpenses;
    }
}

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


        ExpenseTracker tracker = new ExpenseTracker(budget);

        double expense;
        do {
            expense = scanner.nextDouble();
            tracker.recordExpense(expense);
        } while (expense != 0.0);

        double remainingBudget = tracker.calculateRemainingBudget();
        if (remainingBudget >= 0) {
            System.out.printf("Remaining budget: Rs. %.2f", remainingBudget);
        } else {
            System.out.println("No remaining budget, You've exceeded your
budget!");
        }
    }
}
```

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

3. 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

Tax = 0.1 * (78,000 - 50,000) + 0.05 * 50,000 = 5,300

For Salary Rs. 1,10,000

Tax = 0.2 * (110000 - 100000)+ 0.1 * 50,000 + 0.05 * 50,000 = 9,500

For Salary Rs. 23,000

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) {
      int tax = 0;

      if (salary <= 50000) {
         tax = (int) (0.05 * salary);
      } else if (salary <= 100000) {
         tax = (int) (0.1 * (salary - 50000) + 0.05 * 50000);
```

```
        } else {
            tax = (int) (0.2 * (salary - 100000) + 0.1 * 50000 + 0.05 * 50000);
        }

        return 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***

4. Problem Statement

Alex and Bob are designing a control system for household appliances, and one of the appliances is a washing machine. You want to create a program to help them that models the washing machine as a motor and calculates its electricity consumption based on its capacity.

Define an interface named Motor with the following methods:

void run() double consume(double capacity)

Create a class called WashingMachine that implements the Motor interface.

In the WashingMachine class:

Implement the run() method to print "Washing machine is running."Implement a consume() method to print "Washing machine is consuming electricity."Implement the consume(double capacity) method to calculate the electricity consumption (in kWh) of the washing machine based on its capacity. The formula for electricity consumption is (capacity * 0.05).

Input Format

The input consists of a double value representing the capacity of the washing machine in kW.

Output Format

The first line of output prints "Washing machine is running."

The second line prints "Washing machine is consuming electricity."

The third line prints "Electricity consumption: X kWh" where X is a double value, rounded off to two decimal places, representing the electricity consumption.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2.5
Output: Washing machine is running.
Washing machine is consuming electricity.
Electricity consumption: 0.13 kWh

Answer

```
import java.util.Scanner;

interface Motor {
   void run();
   double consume(double capacity);
}

class WashingMachine implements Motor {
   public void run() {
     System.out.println("Washing machine is running.");
   }
   public void consume() {
     System.out.println("Washing machine is consuming electricity.");
   }


   public double consume(double capacity) {
     double consumption = capacity * 0.05;
     return consumption;
   }
}
public class Main {
   public static void main(String[] args) {
     Scanner scanner = new Scanner(System.in);

     WashingMachine washingMachine = new WashingMachine();

     double capacity = scanner.nextDouble();

     washingMachine.run();
     washingMachine.consume();


     double consumption = washingMachine.consume(capacity);
     System.out.printf("Electricity consumption: %.2f kWh" ,consumption);

     scanner.close();
   }
}
```

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