Rajalakshmi Engineering College

Name: Sruthi Dinesh
Email: 240701536@rajalakshmi.edu.in
Roll no: 240701536
Phone: 7845725087
Branch: REC
Department: CSE - Section 7
Batch: 2028
Degree: B.E - CSE

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

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

```java
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.");
    }
```

```java
    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*

2.   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

```java
import java.util.Scanner;


interface SpeedConverter {
    double convertSpeed(double mph);
}

class MPHtoKPHConverter implements SpeedConverter {
    public double convertSpeed(double mph) {
        return mph * 1.60934;
    }
}

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*

3. 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);
}

```java
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!");
        }
    }
```

}

4. Problem Statement

Jeevan is developing a fitness-tracking application to monitor daily physical activity.

The application incorporates a FitnessTracker class that implements two interfaces: StepCounter for tracking the number of steps taken and CalorieCalculator for estimating total calories burned based on total steps.

Jeevan needs your help creating a program.

Note

The calorie calculation formula is: Total caloriesBurned = (total steps / 100.0) * 20.0.

Input Format

The first line of input is an integer n, representing the number of days Jeevan wants to input data.

The second line consists of space-separated integers, representing the number of steps Jeevan took on each day.

Output Format

The first line of output prints: "Total Steps: <totalSteps>", where '<totalSteps>' is the sum of steps (integer) taken over 'n' days.

The second line prints: "Calories Burned: <caloriesBurned>", where '<caloriesBurned>' is the estimated total calories (double-point number) burned based on the total steps taken rounded off to two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
340 234 987

Output: Total Steps: 1561
Calories Burned: 312.20

Answer

```java
import java.util.Scanner;

interface StepCounter {
    void countSteps(int steps);
}

interface CalorieCalculator {
    double calculateCaloriesBurned(int steps);
}

class FitnessTracker implements StepCounter, CalorieCalculator {
    private int totalSteps;

    public void countSteps(int steps) {
        totalSteps += steps;
    }

    public double calculateCaloriesBurned(int steps) {
        double caloriesBurned = (steps / 100.0) * 20.0;
        return caloriesBurned;
    }

    public int getTotalSteps() {
        return totalSteps;
    }
}
class Main
{

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

        FitnessTracker tracker = new FitnessTracker();
```

```java
        int n = scanner.nextInt();

        for (int i = 0; i < n; i++) {
            int steps = scanner.nextInt();
            tracker.countSteps(steps);
        }

        int totalSteps = tracker.getTotalSteps();
        System.out.println("Total Steps: " + totalSteps);

        double caloriesBurned = tracker.calculateCaloriesBurned(totalSteps);
        System.out.printf("Calories Burned: %.2f%n", caloriesBurned);

        scanner.close();
    }
}
```

Status : Correct   *Marks : 10/10*