

Rajalakshmi Engineering College

Name: Anbarasu V
Email: 241501018@rajalakshmi.edu.in
Roll no: 241501018
Phone: 9488440199
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

REC_2028_OOPS using Java_Week 6_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Teena's retail store has implemented a Loyalty Points System to reward customers based on their spending. The program calculates and displays the loyalty points based on whether the customer is a regular or a premium customer.

For regular customers (class Customer), the loyalty points are calculated as:

$\text{Loyalty points} = \text{amount spent} / 10$

For premium customers (class PremiumCustomer, which inherits from Customer), the loyalty points are calculated as:

$\text{Loyalty points} = 2 * (\text{amount spent} / 10)$

The program should use method overriding for premium customers to

calculate their loyalty points. The method that needs to be overridden is calculateLoyaltyPoints in the Customer class.

Input Format

The first line of input consists of an integer representing the amount spent by the customer.

The second line consists of a string representing the premium customer status:

- "yes" if the customer is a premium customer.
- "no" if the customer is not a premium customer.

Output Format

The output should display the loyalty points earned based on the amount spent and the customer type.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50

yes

Output: 10

Answer

```
import java.util.Scanner;
```

```
class Customer {  
    public int calculateLoyaltyPoints(int amount) {  
        return amount / 10;  
    }  
}
```

```
class PremiumCustomer extends Customer {  
    @Override  
    public int calculateLoyaltyPoints(int amount) {  
        return 2 * (amount / 10);  
    }  
}
```

```

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

        int amountSpent = scanner.nextInt();

        String isPremium = scanner.next().toLowerCase();

        Customer customer;

        if (isPremium.equals("yes")) {
            customer = new PremiumCustomer();
        } else {
            customer = new Customer();
        }

        int loyaltyPoints = customer.calculateLoyaltyPoints(amountSpent);

        System.out.println(loyaltyPoints);
    }
}

```

Status : Correct

Marks : 10/10

2. Problem Statement

Arun wants to calculate the age gap between the grandfather and the son and determine the father's age after 5 years.

Your task is to assist him in developing a program using three classes: GrandFather, Father, and Son, where the GrandFather stores the grandfather's age, the Father extends GrandFather to include the father's age and calculates his age after 5 years, and Son extends Father to include the son's age and calculate the age difference between the grandfather and the son.

Input Format

The input consists of three integers representing the ages of the grandfather,

father, and son, one per line.

Output Format

The first line of output prints "Grandfather and son's age gap:" followed by an integer representing the age gap between the grandfather and the son, ending with "years".

The second line prints "Father's Age:" followed by an integer representing the father's age after 5 years, ending with "years".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50

30

3

Output: Grandfather and son's age gap: 47 years

Father's Age: 35 years

Answer

```
import java.util.Scanner;
```

```
class GrandFather {  
    int grandfatherAge;
```

```
    GrandFather() {}
```

```
    void setGrandfatherAge(int age) {  
        this.grandfatherAge = age;  
    }  
}
```

```
class Father extends GrandFather {  
    int fatherAge;
```

```
    Father() {}
```

```
    void setFatherAge(int age) {  
        this.fatherAge = age;
```

```

    }
    int calculateFatherAgeAfter5Years() {
        return fatherAge + 5;
    }
}

class Son extends Father {
    int sonAge;

    Son() {}

    void setSonAge(int age) {
        this.sonAge = age;
    }

    int calculateGrandfatherSonAgeDifference() {
        return grandfatherAge - sonAge;
    }
}

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

        int grandfatherAge = scanner.nextInt();
        son.setGrandfatherAge(grandfatherAge);

        int fatherAge = scanner.nextInt();
        son.setFatherAge(fatherAge);

        int sonAge = scanner.nextInt();
        son.setSonAge(sonAge);

        System.out.println("Grandfather and son's age gap: "+
            son.calculateGrandfatherSonAgeDifference() + " years");

        int fatherAgeAfter5Years = son.calculateFatherAgeAfter5Years();
        System.out.println("Father's Age: " + fatherAgeAfter5Years + " years");
    }
}

```

Status : Correct

Marks : 10/10

3. Problem Statement

Mary is managing a business and wants to analyze its profitability. She operates both a regular business model and a seasonal business model. To assess profitability, she uses a program that calculates and compares the profit margins for both models based on revenue and cost.

The program defines:

BusinessUtility class with a method calculateMargin(double revenue, double cost). SeasonalBusinessUtility (inherits from BusinessUtility) and overrides calculateMargin(double revenue, double cost), adding a seasonal adjustment of 10% to the base margin. ProfitabilityChecker class with a method checkProfitability(double regularMargin), which prints "Business is profitable." if the regular margin is 10% or more, otherwise prints "Business is not profitable."

Mary inputs revenue and cost, and the program compute and display the regular and seasonal margins using:

$$\text{Margin} = ((\text{Revenue} - \text{Cost}) / \text{Revenue}) \times 100$$

$$\text{Seasonal Margin} = \text{Margin} + 10$$

Input Format

The first line of input consists of a double value r , representing the revenue.

The second line consists of a double value c , representing the cost.

Output Format

The first line prints a double value, representing the regular profit margin, rounded to two decimal places, in the format: "Regular Margin: X. XX%", where X.XX denotes the calculated regular margin.

The second line prints a double value, representing the seasonal profit margin, rounded to two decimal places, in the format: "Seasonal Margin: X. XX%", where X.XX denotes the calculated seasonal margin.

The third line prints a string, indicating whether the business is profitable or not profitable, based on the regular margin.

If the regular margin is less than 10, print "Business is not profitable.". If it is 10 or greater, print "Business is profitable."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1000.0

800.0

Output: Regular Margin: 20.00%

Seasonal Margin: 30.00%

Business is profitable.

Answer

```
import java.util.Scanner;
```

```
class BusinessUtility {  
    double calculateMargin(double revenue, double cost) {  
        return ((revenue - cost) / revenue) * 100;  
    }  
}
```

```
class SeasonalBusinessUtility extends BusinessUtility {  
    @Override  
    double calculateMargin(double revenue, double cost) {  
        return super.calculateMargin(revenue, cost) + 10;  
    }  
}
```

```
class ProfitabilityChecker {  
    void checkProfitability(double regularMargin) {  
        if (regularMargin >= 10) {  
            System.out.println("Business is profitable.");  
        } else {  
            System.out.println("Business is not profitable.");  
        }  
    }  
}
```

```

    }
}
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double revenue = scanner.nextDouble();
        double cost = scanner.nextDouble();
        BusinessUtility business = new BusinessUtility();
        SeasonalBusinessUtility seasonalBusiness = new
SeasonalBusinessUtility();
        double regularMargin = business.calculateMargin(revenue, cost);
        double seasonalMargin = seasonalBusiness.calculateMargin(revenue,
cost);

        System.out.printf("Regular Margin: %.2f%%\n", regularMargin);
        System.out.printf("Seasonal Margin: %.2f%%\n", seasonalMargin);

        ProfitabilityChecker checker = new ProfitabilityChecker();
        checker.checkProfitability(regularMargin);
        scanner.close();
    }
}

```

Status : Correct

Marks : 10/10

4. Problem Statement

Adams has a reputation company with a great number of employees. He must calculate the salary weekly according to the hourly rate and working hours. Create a program to define a class Employee with attributes name and hourly rate. Create a subclass HourlyEmployee that calculates the weekly salary based on the number of hours worked.

(The first 40 hours are based on the regular hour rate. If the work hours are greater than 40 then the work wage is 1.5 times the hourly rate)

Note: Use Math(Math.max, Math.min) functions .

Example

Input:

Chris

10

45

Output:

Weekly Salary: Rs.475.00

Explanation:

Calculation:

The first 40 hours are paid normally: $40 \times 10 = 400.00$ The extra 5 hours are paid at 1.5 times the hourly rate: $5 \times (10 \times 1.5) = 5 \times 15 = 75.00$ Total salary: $400.00 + 75.00 = 475.00$

Input Format

The first line of input consists of a string that represents the name of the employee.

The second line consists of a double value that represents the rate for an hour.

The last line consists of an integer that represents the total hours worked.

Output Format

The output displays the total salary of the employee, where salary is rounded to two decimal places in the format: "Weekly Salary: Rs.<double value>".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Dave

10.0

40

Output: Weekly Salary: Rs.400.00

Answer

```
import java.util.Scanner;
import java.text.DecimalFormat;
```

```
class Employee {
    String name;
    double hourlyRate;

    Employee(String name, double hourlyRate) {
        this.name = name;
        this.hourlyRate = hourlyRate;
    }
}
```

```
class HourlyEmployee extends Employee {
    int hoursWorked;
```

```
    HourlyEmployee(String name, double hourlyRate, int hoursWorked) {
        super(name, hourlyRate);
        this.hoursWorked = hoursWorked;
    }
```

```
    double calculateWeeklySalary() {
        int regularHours = (int)Math.min(40, hoursWorked);
        int overtimeHours = (int)Math.max(0, hoursWorked - 40);
        double regularPay = regularHours * hourlyRate;
        double overtimePay = overtimeHours * (hourlyRate * 1.5);
        return regularPay + overtimePay;
    }
}
```

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

        String name = scanner.nextLine();
        double hourlyRate = scanner.nextDouble();
        int hoursWorked = scanner.nextInt();

        HourlyEmployee employee = new HourlyEmployee(name, hourlyRate,
hoursWorked);
```

```
double weeklySalary = employee.calculateWeeklySalary();
DecimalFormat df = new DecimalFormat("#.00");
String formattedSalary = df.format(weeklySalary);
System.out.println("Weekly Salary: Rs." + formattedSalary);
scanner.close();
    }
}
```

Status : Correct

Marks : 10/10