Rajalakshmi Engineering College

Name: SIVAGURU D
Email: 240701517@rajalakshmi.edu.in
Roll no: 240701517
Phone: 9345616842
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 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:

Loyalty points = amount spent / 10

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

Loyalty points = 2 * (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 aS;
    public int calculateLoyaltyPoints(int aS){
        return aS/10;
    }
}

class PremiumCustomer extends Customer {
    public int calculateLoyaltyPoints(int aS){
        return 2*(aS/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

A bank provides two types of deposit schemes: Fixed Deposits (FD) and Recurring Deposits (RD). Customers want to calculate the interest they can earn based on their selected scheme.

Develop a Java program using inheritance to compute the interest for FD and RD. The program should include:

A base class Account with attributes accountHolder and principalAmount, along with a method for interest calculation.A subclass FixedDeposit that calculates interest for FD.A subclass RecurringDeposit that calculates interest for RD.

Formulas Used:

Interest for FD: (principal amount * duration in years * rate of interest) / 100

Interest for RD: (maturity amount * duration in months * rate of interest) / (12 * 100), where maturity amount = monthly deposit * duration in months.

Input Format

The first line of input consists of the choice (1 for FD, 2 for RD).

If the choice is 1, the following lines consist of account holder (string), principal amount (double), duration in years (int), and rate of interest (double).

If the choice is 2, the following lines consist of account holder (string), monthly deposit (int), duration in months (int), and rate of interest (double).

Output Format

The output prints the calculated interest with one decimal place in the following format.

For choice 1: "Interest for FD: <calculated interest >"

For choice 2: "Interest for FD: <calculated interest >"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
Alice
50000.56
5
6.5
Output: Interest for FD: 16250.2

Answer

```
import java.util.Scanner;

class Account{
   public String aH;
   public double pA;
}
class FixedDeposit extends Account{
```

```
    private int dur;
    private double rate;
    public FixedDeposit(String aH,double pA,int dur,double rate)
    {
        this.aH=aH;
        this.pA=pA;
        this.dur=dur;
        this.rate=rate;
    }
    public float calculateInterest(){
        return (float)(pA*dur*rate)/100;
    }
}
class RecurringDeposit extends Account{
    private int dur;
    private double rate;
    private int dep;
    public RecurringDeposit(String aH,int dep,int dur,double rate)
    {
        this.aH=aH;
        this.dep=dep;
        this.dur=dur;
        this.rate=rate;
    }
    public float calculateInterest(){
        return (float)((dep*dur*dur*rate)/(12*100));
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int choice = sc.nextInt();

        switch (choice) {
            case 1:
                sc.nextLine();
                String fdName = sc.nextLine();
                double fdPrincipal = sc.nextDouble();
                int fdDuration = sc.nextInt();
                double fdRate = sc.nextDouble();
```

```
            FixedDeposit fd = new FixedDeposit(fdName, fdPrincipal, fdDuration,
fdRate);
            System.out.printf("Interest for FD: %.1f", fd.calculateInterest());
            break;

        case 2:
            sc.nextLine();
            String rdName = sc.nextLine();
            int rdDeposit = sc.nextInt();
            int rdDuration = sc.nextInt();
            double rdRate = sc.nextDouble();

            RecurringDeposit rd = new RecurringDeposit(rdName, rdDeposit,
rdDuration, rdRate);
            System.out.printf("Interest for RD: %.1f", rd.calculateInterest());
            break;

        default:
            System.out.println("Invalid Choice");
    }
  }
}
```

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

3. 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 × 10 = 400.00The extra 5 hours are paid at 1.5 times the hourly rate: 5 × (10×1.5) = 5 × 15 = 75.00Total 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{
    public String name;
    public double hr;
}
class HourlyEmployee extends Employee{
    public int hw;
    public HourlyEmployee(String name,double hr,int hw){
        this.name=name;
        this.hr=hr;
        this.hw=hw;
    }
    public double calculateWeeklySalary(){
        if(hw<=40) return hw*hr;
        else return (40*hr)+((hw-40)*(hr*1.5));
    }
}

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***

4. 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:

Margin = ((Revenue − Cost) / Revenue) × 100

Seasonal Margin = 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{
    public double rev;
    public double cost;
    public double calculateMargin(double rev,double cost){
        return ((rev-cost)/rev)*100;
    }
}
class SeasonalBusinessUtility extends BusinessUtility{
    public double calculateMargin(double rev,double cost){
        return (((rev-cost)/rev)*100)+10;
    }
}
class ProfitabilityChecker extends BusinessUtility{
    public int checkProfitability(double rm){
        if(rm<10) System.out.println("Business is not profitable.");
        else System.out.println("Business is profitable.");
        return 0;
    }
}

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***