Assignment 3

09 September 2024 10:57

1. Loan Amortization Calculator

Implement a system to calculate and display the monthly payments for a mortgage loan. The system should:
Accept the principal amount (loan amount), annual interest rate, and loan term (in years) from the user.
Calculate the monthly payment using the standard mortgage formula:

Monthly Payment Calculation:
monthlyPayment = principal * (monthlyInterestRate * (1 + monthlyInterestRate)^(numberOfMonths)) / ((1 + monthlyInterestRate)^(numberOfMonths) - 1)
Where monthlyInterestRate = annualInterestRate / 12 / 100 and numberOfMonths = loanTerm * 12

Note: Here ^ means power and to find it you can use Math.pow( ) method
Display the monthly payment and the total amount paid over the life of the loan, in Indian Rupees (₹).
Define class LoanAmortizationCalculator with methods acceptRecord, calculateMonthlyPayment & printRecord and test the functionality in main method.

Code:

```
package org.example;

import java.util.Scanner;

class Account{
    int principal;
    int annualInterestRate;
    int loanTerm;
    int monthlyPayment;
    int monthlyInterestRate;
    int numberOfMonths;

    void accept_data() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter principle amount        :    ");
        principal = sc.nextInt();
        System.out.print("Enter anual interest rate  :    ");
        annualInterestRate = sc.nextInt();
        System.out.print("Enter loan term              :    ");
        loanTerm = sc.nextInt();
    }

    void calculate_payment() {
        monthlyInterestRate = annualInterestRate / 12 / 100;
        numberOfMonths = loanTerm * 12;
        monthlyPayment = principal * (monthlyInterestRate * (1 + monthlyInterestRate)^(numberOfMonths)) / ((1 +
        monthlyInterestRate)^(numberOfMonths) - 1);

    }

    void print_data() {
        System.out.print("Monthly payment              :    "+ this.monthlyPayment);
    }
}


public class Loan_Calculator {

    public static void main(String[] args) {
```

```
        Account acc = new Account();

        acc.accept_data();
        acc.calculate_payment();
        acc.print_data();
    }

}
```

Output:

```
Enter principle amount          :       1200
Enter anual interest rate       :       12
Enter loan term                 :       8
Monthly payment                 :       1225
```

2. Compound Interest Calculator for Investment

Develop a system to compute the future value of an investment with compound interest. The system should:
Accept the initial investment amount, annual interest rate, number of times the interest is compounded per year, and investment duration (in years) from the user.
Calculate the future value of the investment using the formula:

Future Value Calculation:
futureValue = principal * (1 + annualInterestRate / numberOfCompounds)^(numberOfCompounds * years)
Total Interest Earned: totalInterest = futureValue - principal

Display the future value and the total interest earned, in Indian Rupees (₹).
Define class CompoundInterestCalculator with methods acceptRecord , calculateFutureValue, printRecord and test the functionality in main method.

Code:

```
package org.example;

import java.util.Scanner;

class CompoundInterestCalculator{
    int principal;
    int annualInterestRate;
    int numberOfCompounds;
    int years;
    int futureValue;
    int totalInterest;

    void acceptRecord(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Principal       :       ");
        principal = sc.nextInt();
        System.out.print("number Of Compounds    :       ");
        numberOfCompounds = sc.nextInt();
        System.out.print("years   :       ");
```

```
        years = sc.nextInt();
        sc.nextLine();
        System.out.print("annual Interest Rate        :        ");
        annualInterestRate = sc.nextInt();
    }

    void calculateFutureValue(){

        futureValue = principal * (1 + annualInterestRate / numberOfCompounds)^(numberOfCompounds * years);

        totalInterest = futureValue - principal;

    }

    void printRecord(){
        System.out.println("future Value        :        "+ futureValue);
        System.out.print("total Interest:        "+ totalInterest);
    }

}

public class CI_Calculator {
    public static void main(String[] args) {
        CompoundInterestCalculator cic = new CompoundInterestCalculator();

        cic.acceptRecord();
        cic.calculateFutureValue();
        cic.printRecord();

    }
}
```

Output:

```
Principal       :       12000
number Of Compounds     :       20
years   :       8
annual Interest Rate    :       9
future Value    :       11840
total Interest  :       -160
```

3. BMI (Body Mass Index) Tracker

Create a system to calculate and classify Body Mass Index (BMI). The system should:
Accept weight (in kilograms) and height (in meters) from the user.

Calculate the BMI using the formula:
BMI Calculation: BMI = weight / (height * height)
Classify the BMI into one of the following categories:
Underweight: BMI < 18.5
Normal weight: 18.5 ≤ BMI < 24.9
Overweight: 25 ≤ BMI < 29.9
Obese: BMI ≥ 30

Display the BMI value and its classification.
Define class BMITracker with methods acceptRecord, calculateBMI, classifyBMI & printRecord and test the functionality in main method.

Code:

```
package org.example;
import java.util.Scanner;

class BMITracker{
	private float height;
	private float weight;
	private float BMI;
	private float calculateBMI;

	void acceptRecord(){
		Scanner sc = new Scanner(System.in);
		System.out.print("height in meter	:	");
		height=sc.nextFloat();
		System.out.print("weight in kg	:	");
		weight=sc.nextFloat();

	}

	void calculateBMI(){
		BMI = weight/(height * height);
		System.out.println("bmi	:	" + BMI);
	}

	void classifyBMI(){
		if(BMI<18.5) {
			System.out.print("Underweight");
		}
		else if(BMI>18.5 && BMI<24.9){
			System.out.print("Normal weight");
		}
		else if(BMI>25 && BMI<29.9){
			System.out.print("Overweight");
		}
		else{
			System.out.print("Obese");
		}
	}

	void printRecord(){
		this.calculateBMI();
		this.classifyBMI();
	}
}

public class Program{
	public static void main(String[] args){
		BMITracker bmi = new BMITracker();

		bmi.acceptRecord();
		bmi.printRecord();

	}

}
```

Output:

```
height in meter :       2
weight in kg    :       90
bmi     :       22.5
Normal weight
```

4. Discount Calculation for Retail Sales

Design a system to calculate the final price of an item after applying a discount. The system should:
Accept the original price of an item and the discount percentage from the user.
Calculate the discount amount and the final price using the following formulas:
Discount Amount Calculation: discountAmount = originalPrice * (discountRate / 100)

Final Price Calculation: finalPrice = originalPrice - discountAmount

Display the discount amount and the final price of the item, in Indian Rupees (₹).
Define class DiscountCalculator with methods acceptRecord, calculateDiscount & printRecord and test the functionality in main method.

Code:

```
package org.example;
import java.util.Scanner;

class DiscountCalculator{
    private double discountAmount;
    private double originalPrice;
    private double discountRate;
    private double finalPrice;


    void acceptRecord(){
        Scanner sc = new Scanner(System.in);
        System.out.print("original Price : ");
        originalPrice = sc.nextDouble();
        System.out.print("discount Rate : ");
        discountRate = sc.nextDouble();
    }

    void calculateDiscount(){
        discountAmount = originalPrice*(discountRate / 100);
        finalPrice = originalPrice-discountAmount;
    }

    void printRecord() {
        System.out.println("discount Amount : " + this.discountAmount);
        System.out.println("final price : " + this.finalPrice);
    }

}


public class Program {

    public static void main(String[] args) {
        DiscountCalculator dc = new DiscountCalculator();
```

```
            dc.acceptRecord();
            dc.calculateDiscount();
            dc.printRecord();
        }

}
```

Output:

```
original Price : 15000
discount Rate : 12
discount Amount : 1800.0
final price : 13200.0
```

5. Toll Booth Revenue Management

Develop a system to simulate a toll booth for collecting revenue. The system should:
Allow the user to set toll rates for different vehicle types: Car, Truck, and Motorcycle.
Accept the number of vehicles of each type passing through the toll booth.
Calculate the total revenue based on the toll rates and number of vehicles.
Display the total number of vehicles and the total revenue collected, in Indian Rupees (₹).

Toll Rate Examples:
Car: ₹50.00
Truck: ₹100.00
Motorcycle: ₹30.00

Define class TollBoothRevenueManager with methods acceptRecord, setTollRates, calculateRevenue & printRecord and test the functionality in main method.

Code:

```
package org.example;
import java.util.Scanner;

class TollBoothRevenueManager {
   private int count;
   private float revenue;
   private float toll;
   private String vehicle;

   void acceptRecord() {
      Scanner sc = new Scanner(System.in);
      System.out.print("Vehicle passed is: ");
      vehicle = sc.nextLine();
   }

   void setTollRates() {
      switch(vehicle.toLowerCase()) {
         case "car":
            System.out.println("Car passed the toll");
            toll = 50;
            count += 1;
            break;
         case "truck":
```

```
            System.out.println("Truck passed the toll");
            toll = 100;
            count += 1;
            break;
        case "motorcycle":
            System.out.println("Motorcycle passed the toll");
            toll = 30;
            count += 1;
            break;
        default:
            System.out.println("Invalid vehicle type");
            toll = 0;
            break;
      }
   }

   void calculateRevenue() {
      revenue += toll;
   }

   void printRecord() {
      System.out.println("Revenue: " + this.revenue);
      System.out.println("Number of vehicles: " + this.count);
      System.out.println();
   }

   static void menulist() {
      System.out.println("Press 0 to Exit");
      System.out.println("Press 1 for Accept Record");
      System.out.println("Press 2 for Print Record");
   }
}

public class Program {
   public static void main(String[] args) {
      TollBoothRevenueManager tbrm = new TollBoothRevenueManager();
      Scanner sc = new Scanner(System.in);
      int choice = -1;

      while(choice!=0) {

         TollBoothRevenueManager.menulist();
         choice = sc.nextInt();
         switch(choice) {
                case 1:
                        tbrm.acceptRecord();
                        tbrm.setTollRates();
                        tbrm.calculateRevenue();
                        break;
                case 2:
                        tbrm.printRecord();
                        break;
                default:
                        System.out.println("Exit");
                        break;
         }
      }

   }
}
```

Output:

```
original Price : 15000
discount Rate : 12
discount Amount : 1800.0
final price : 13200.0
```