Note:

- The assignment is designed to practice class, fields, and methods only.
- Create a separate project for each question.
- Do not use getter/setter methods or constructors for these assignments.
- Define two classes: one class to implement the logic and another class to test it.

1. Loan Amortization Calculator

Implement a system to calculate and display the monthly payments for a mortgage loan. The system should:

1. Accept the principal amount (loan amount), annual interest rate, and loan term (in years) from the user.
2. 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
3. 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.

Solution:

```
package exp.java;

import java.util.Scanner;

class LoanAmortizationCalculator {

    private double principal;
    private double annualInterestRate;
    private int loanTerm;

    Scanner scanner = new Scanner(System.in);

    public void acceptRecord() {
        System.out.print("Enter the loan amount (₹): ");
        principal = scanner.nextDouble();

        System.out.print("Enter the annual interest rate (%): ");
        annualInterestRate = scanner.nextDouble();

        System.out.print("Enter the loan term (in years): ");
```

```
            loanTerm = scanner.nextInt();
        }

        public double calculateMonthlyPayment() {
            double monthlyInterestRate = annualInterestRate / 12 / 100;
            int numberOfMonths = loanTerm * 12;

            double monthlyPayment = principal * (monthlyInterestRate * Math.pow(1 +
monthlyInterestRate, numberOfMonths))
                    / (Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1);

            return monthlyPayment;
        }

        public void printRecord(double monthlyPayment) {
            int numberOfMonths = loanTerm * 12;
            double totalPayment = monthlyPayment * numberOfMonths;

            System.out.printf("Monthly Payment: ₹%.2f%n", monthlyPayment);
            System.out.printf("Total Payment over the life of the loan: ₹%.2f%n",
totalPayment);
        }
}

public class loan {
        public static void main(String[] args) {

            LoanAmortizationCalculator cal = new LoanAmortizationCalculator();

            cal.acceptRecord();

            double monthlyPayment = cal.calculateMonthlyPayment();

            cal.printRecord(monthlyPayment);
        }
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> loan (1) [Java Application] D:\ESCLIPSE\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x
Enter the loan amount (₹): 100000
Enter the annual interest rate (%): 5.5
Enter the loan term (in years): 2
Monthly Payment: ₹4409.57
Total Payment over the life of the loan: ₹105829.57
```

2. Compound Interest Calculator for Investment

ASSIGNMENT NO.3

Develop a system to compute the future value of an investment with compound interest. The system should:

1. 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.
2. 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`
3. 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.

Solution:

```
package exp.java;

import java.util.Scanner;

class CompoundInterest {

	private double principal;
	private double annualInterestRate;
	private int numberOfCompounds;
	private int years;

	Scanner Sc = new Scanner(System.in);

	public void acceptRecord() {

		System.out.print("Enter the initial investment amount (₹): ");
		principal = Sc.nextDouble();

		System.out.print("Enter the annual interest rate (%): ");
		annualInterestRate = Sc.nextDouble();

		System.out.print("Enter the number of times the interest is compounded per
year: ");
		numberOfCompounds = Sc.nextInt();

		System.out.print("Enter the investment duration (in years): ");
		years = Sc.nextInt();
	}

	// formulas
	public double calculateFutureValue() {
		double rate = annualInterestRate / 100; // Converting percentage to decimal
		double futureValue = principal * Math.pow((1 + rate /
numberOfCompounds), numberOfCompounds * years);
```

```
            return futureValue;
        }

        public void printRecord(double futureValue) {
            double totalInterest = futureValue - principal;

            System.out.printf("Future Value: ₹%.2f%n", futureValue);
            System.out.printf("Total Interest Earned: ₹%.2f%n", totalInterest);
        }
}

public class Calculator {
        public static void main(String[] args) {
            CompoundInterest cal = new CompoundInterest();

            cal.acceptRecord();

            double futureValue = cal.calculateFutureValue();

            cal.printRecord(futureValue);
        }
}
```

Output:

```
Problems  Javadoc  Declaration  Console ×  Coverage
<terminated> Calculator [Java Application] D:\ESCLIPSE\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530
Enter the initial investment amount (₹): 70000
Enter the annual interest rate (%): 7.5
Enter the number of times the interest is compounded per year: 2
Enter the investment duration (in years): 2
Future Value: ₹81105.53
Total Interest Earned: ₹11105.53
```

3. BMI (Body Mass Index) Tracker

Create a system to calculate and classify Body Mass Index (BMI). The system should:

1. Accept weight (in kilograms) and height (in meters) from the user.
2. Calculate the BMI using the formula:
   - **BMI Calculation:** `BMI = weight / (height * height)`
3. 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
4. Display the BMI value and its classification.

Define class BMITracker with methods acceptRecord, calculateBMI, classifyBMI & printRecord and test the functionality in main method.

Solution:

```
    package exp.java.in;

import java.util.Scanner;

 class BMITracker {
   private double Weight; // in kilograms
   private double Height; // in meters
   private double bmi;

   public void acceptRecord() {
          Scanner Sc = new Scanner(System.in);
           System.out.print("Enter weight : ");
       Weight = Sc.nextDouble();
       System.out.print("Enter height : ");
       Height = Sc.nextDouble();
       Sc.close();
   }
   public void calculateBMI() {

      bmi = Weight / (Height * Height);
   }

   public String classifyBMI() {
      if (bmi < 18.5) {
         return "Underweight";
      } else if (bmi >= 18.5 && bmi < 24.9) {
         return "Normal weight";
      } else if (bmi >= 25 && bmi < 29.9) {
         return "Overweight";
      } else {
         return "Obese";
      }
   }
   public void printRecord() {
      System.out.printf("Your BMI is: %.2f\n", bmi);
      System.out.println("BMI Classification: " + classifyBMI());
   }
    public static void main(String[] args) {
          BMITracker tracker = new BMITracker();
                tracker.acceptRecord();
                tracker.calculateBMI();
                tracker.printRecord();

          }

}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> BMITracker [Java Application] D:\ESCLIPSE\eclipse\plugins\
Enter weight : 65
Enter height : 5.6
Your BMI is: 2.07
BMI Classification: Underweight
```

4. Discount Calculation for Retail Sales

Design a system to calculate the final price of an item after applying a discount. The system should:

1. Accept the original price of an item and the discount percentage from the user.
2. 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`
3. 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.

Solution:

```
package exp.java.in;

import java.util.Scanner;
class DiscountCalculator {
        private double originalPrice;
        private double discountRate;
        private double discountAmount;
        private double finalPrice;

        Scanner sc = new Scanner(System.in);
        public void acceptRecord() {

                System.out.print("Enter original price  : ");
                originalPrice = sc.nextDouble();
                System.out.print("Enter discount Rate   :  ");
                discountRate = sc.nextDouble();
        }
        public void calculateDiscount() {
                discountAmount = originalPrice * (discountRate / 100 );
                finalPrice = originalPrice - discountAmount;
        }
         public void printRecord() {
```

```
        System.out.printf("Discount Amount: ₹%.2f\n", discountAmount);
        System.out.printf("Final Price after Discount: ₹%.2f\n", finalPrice);
    }


    public static void main(String[] args) {
        DiscountCalculator calculator = new DiscountCalculator();
        calculator.acceptRecord();
        calculator.calculateDiscount();
        calculator.printRecord();
    }

}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> DiscountCalculator [Java Application] D:\ESCLIPSE\eclipse\plugins\or
Enter original price  : 5510
Enter discount Rate   :  10
Discount Amount: ₹551.00
Final Price after Discount: ₹4959.00
```

5. Toll Booth Revenue Management

Develop a system to simulate a toll booth for collecting revenue. The system should:

1. Allow the user to set toll rates for different vehicle types: Car, Truck, and Motorcycle.
2. Accept the number of vehicles of each type passing through the toll booth.
3. Calculate the total revenue based on the toll rates and number of vehicles.
4. 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.

Solution:

```
package exp.java;
import java.util.Scanner;

class TollBoothRevenueManager {
  private double carRate;
```

```
    private double truckRate;
    private double motorcycleRate;
    private int numCars;
    private int numTrucks;
    private int numMotorcycles;
    private double totalRevenue;

    Scanner sc = new Scanner(System.in);

    public void setTollRates() {

        System.out.print("Enter toll rate for Cars : ");
        carRate = sc.nextDouble();
        System.out.print("Enter toll rate for Trucks : ");
        truckRate = sc.nextDouble();
        System.out.print("Enter toll rate for Motorcycles : ");
        motorcycleRate = sc.nextDouble();
    }


    public void acceptRecord() {
        //Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of Cars     : ");
        numCars = sc.nextInt();
        System.out.print("Enter the number of Trucks: ");
        numTrucks = sc.nextInt();
        System.out.print("Enter the number of Motorcycles: ");
        numMotorcycles = sc.nextInt();
    }


    public void calculateRevenue() {
        totalRevenue = (numCars * carRate) + (numTrucks * truckRate) + (numMotorcycles *
motorcycleRate);
    }
    public void printRecord() {
        int totalVehicles = numCars + numTrucks + numMotorcycles;
        System.out.println("Total number of vehicles: " + totalVehicles);
        System.out.printf("Total revenue collected: ₹%.2f\n", totalRevenue);
    }
}
public class RevenueManagement{
    public static void main(String[] args) {
        TollBoothRevenueManager manager = new TollBoothRevenueManager();
        manager.setTollRates();
        manager.acceptRecord();
        manager.calculateRevenue();
```

```
        manager.printRecord();
    }
}
```

Output:

```
Problems  Javadoc  Declaration  Console ×
<terminated> RevenueManagement [Java Application] D:\ESCLIPSE\eclipse\plugins\org.
Enter toll rate for Cars : 50
Enter toll rate for Trucks : 100
Enter toll rate for Motorcycles : 30
Enter the number of Cars        : 7
Enter the number of Trucks: 5
Enter the number of Motorcycles: 2
Total number of vehicles: 14
Total revenue collected: ₹910.00
```