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.

CODE:

```
package org.assignment3;

import java.util.Scanner;

class LoanAmortizationCalculator{
	double principal;
	double annualInterestRate;
	int loanTerm;
	double monthlyPayment;
	double totalAmountPaid;

	Scanner sc = new Scanner(System.in);

	public void acceptRecord() {
		System.out.println("Enter Principle Amount:₹ ");
		this.principal = sc.nextDouble();
		System.out.println("Enter annual interest rate: ");
		this.annualInterestRate = sc.nextDouble();
		System.out.println("Enter loan term: ");
```

```
            this.loanTerm = sc.nextInt();
        }

        public void calculateMonthlyPayment() {
            double monthlyInterestRate = (annualInterestRate / 12) / 100;

            int numberOfMonths = loanTerm * 12;

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

            totalAmountPaid = monthlyPayment * loanTerm * 12;

        }

        public void printRecord() {

            System.out.println("Monthly Payment: ₹" + monthlyPayment);
            System.out.println("Total Payment: ₹ "+totalAmountPaid );

        }
}
public class Program1 {

        public static void main(String[] args) {
      LoanAmortizationCalculator  loan = new LoanAmortizationCalculator ();

            loan.acceptRecord();
            loan.calculateMonthlyPayment();
            loan.printRecord();
        }
}
```

OUTPUT:

```
<terminated> Program1 [Java Application] C:\eclipse\eclipse\plugins\org.eclip
Enter Principle Amount:₹
200000
Enter annual interest rate:
4.5
Enter loan term:
12
Monthly Payment: ₹1800.016321183822
Total Payment: ₹ 259202.35025047036
```

2. Compound Interest Calculator for Investment

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.

CODE:

```
package org.assignment3;

import java.util.Scanner;

class CompoundInterestCalculator {
	int principle;
	double annualInterestRate;
	int numberOfCompounds;
	int years;
	double futureValue;
	double totalInterest;

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

		System.out.println("Principle :");
		this.principle = sc.nextInt();
		System.out.println("Annual Interest Rate :");
		this.annualInterestRate = sc.nextDouble() / 100;
		System.out.println("Number of Compounds :");
		this.numberOfCompounds = sc.nextInt();
		System.out.println("Years :");
		this.years = sc.nextInt();

		sc.close();
	}

	public void calculateFutureValue() {

		futureValue = principle * Math.pow(1 + annualInterestRate /
numberOfCompounds, numberOfCompounds * years);
	totalInterest = futureValue - principle;
```

```
        }

        public void printRecord() {

                System.out.println("Future Value : "+futureValue);
                System.out.println("Total Interest Earned : "+totalInterest);


        }

}
public class Project2 {


        public static void main(String[] args) {
      CompoundInterestCalculator calculator = new CompoundInterestCalculator();

                calculator.acceptRecord();
                calculator.calculateFutureValue();
                calculator.printRecord();
        }
}
```

OUTPUT:

```
Principle :
60000
Annual Interest Rate :
8
Number of Compounds :
5
Years :
10
Future Value : 132689.5275565388
Total Interest Earned : 72689.52755653879
```


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.

CODE:

```
package org.assignment3;

import java.util.Scanner;

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

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

		System.out.println("Enter Weight in kg: ");
		this.weight = sc.nextFloat();
		System.out.println("Enter Height in meters: ");
		this.height = sc.nextFloat();
	}

	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.println("BMI Value :	"+BMI);
```

```
            System.out.println("BMI Classification :      "+ classifyBMI());
        }
}

public class Program4 {

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

            bmi.acceptRecord();
            bmi.calculateBMI();
            bmi.classifyBMI();
            bmi.printRecord();
        }
}
```

OUTPUT:

```
Enter Weight in kg:
50
Enter Height in meters:
1.56
BMI Value :     20.545696
BMI Classification :    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:

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.

CODE:

```
package org.assignment3;

import java.util.Scanner;
```

```
class DiscountCalculator {
	float originalPrice;
	float discountRate;
	float finalPrice;
	float discountAmount;

	public void acceptRecord() {
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter Original Price: ");
		this.originalPrice = sc.nextFloat();
		System.out.println("Enter Discount Rate: ");
		this.discountRate = sc.nextFloat();
		sc.close();
	}

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

	public void printRecord() {

		System.out.println("Discount Amount: "+ discountAmount);
		System.out.println("Final Amount: "+ finalPrice);
	}
}

public class Program3 {

	public static void main(String[] args) {

      DiscountCalculator cal = new DiscountCalculator();

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

OUTPUT:

```
<terminated> Program3 [Java Application] C:\eclipse
Enter Original Price:
4000
Enter Discount Rate:
15
Discount Amount: 600.0
Final Amount: 3400.0
```

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.

CODE:

```
package org.assignment3;

import java.util.Scanner;

class TollBoothRevenueManager{
        private double carTollRate;
    private double truckTollRate;
    private double motorcycleTollRate;

    private int carCount;
    private int truckCount;
    private int motorcycleCount;

    private double totalRevenue;

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

```
        System.out.println("Enter toll rate for Car:");
        carTollRate = sc.nextDouble();

        System.out.println("Enter the toll rate for Truck:");
        truckTollRate = sc.nextDouble();

        System.out.println("Enter the toll rate for Motorcycle:");
        motorcycleTollRate = sc.nextDouble();
    }

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

        System.out.println("Enter the number of Cars:");
        carCount = sc.nextInt();

        System.out.println("Enter the number of Trucks:");
        truckCount = sc.nextInt();

        System.out.println("Enter the number of Motorcycles:");
        motorcycleCount = sc.nextInt();
    }

    public void calculateRevenue() {
        totalRevenue = (carCount * carTollRate) + (truckCount * truckTollRate) +
(motorcycleCount * motorcycleTollRate);
    }


    public void printRecord() {
        int totalVehicles = carCount + truckCount + motorcycleCount;

        System.out.println("Total number of vehicles: " + totalVehicles);
        System.out.println("Total revenue collected: Rs" + totalRevenue);
    }
}
public class Program5 {

        public static void main(String[] args) {
                TollBoothRevenueManager toll = new TollBoothRevenueManager();

                toll.acceptRecord();
                toll.setTollRates();
                toll.calculateRevenue();
                toll.printRecord();

        }
}
```

OUTPUT

```
<terminated> Program3 [Java Application] C:\Eclipse\Eclipse\plugins\org.ec
Enter the number of Motorcycles:
5
Enter toll rate for Car:
80
Enter the toll rate for Truck:
99.8
Enter the toll rate for Motorcycle:
68
Total number of vehicles: 11
Total revenue collected: Rs899.2
```