ASSIGNMENT NO.3

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.

```
package ooj;
import java.util.*;
 class Loancal{
        double principal;
        double annualIntrestRate;
        int loanTerm;

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

                System.out.println("Enter the loan amount (Principal): Rupees ");
                principal = sc.nextDouble();

                System.out.println("Enter the annual intrest rate (in %): ");
                annualIntrestRate = sc.nextDouble();

                System.out.println("Enter the loan term (in years):  ");
                loanTerm = sc.nextInt();

                sc.close();
        }
    double calculateMonthlypay() {
```

```
double monthlyIntrestRate = annualIntrestRate / 12 / 100;
int numberOfMonths = loanTerm * 12;
return (double) (principal * (monthlyIntrestRate * Math.pow(1 + monthlyIntrestRate,
numberOfMonths)) / (Math.pow(1 + monthlyIntrestRate, numberOfMonths) - 1));

        }
    void printRecord() {
    double monthlyPayment = calculateMonthlypay();
    double totalpayment = monthlyPayment * loanTerm * 12;

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

public class Intrest {
        public static void main(String[] args) {
                Loancal cal= new Loancal();
                cal.acceptRecord();
                cal.printRecord();

        }


}
```

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.

```
package ooj;
import java.util.*;

public class Calculator {
        private double principal;
        private double annualIntrestRate;
        private int numberOfCompounds;
        private int years;
        private double futureValue;
        private double totalInterest;

        public void acceptRecord() {
                Scanner sc = new Scanner(System.in);
                System.out.println("Inital Interest Amount (in Rs):  ");
            principal = sc.nextDouble();
                System.out.println("Annual Interest Rate(in %):  ");
                 annualIntrestRate = sc.nextDouble() / 100;
                System.out.println("Number of times the intrest is compounded per year:  ");
                numberOfCompounds = sc.nextInt();
                System.out.println("Investment duration(in yrs):  ");
                 years = sc.nextInt();
                 sc.close();
        }

        public void calculateFutureValue() {
           futureValue = principal * Math.pow((1 + annualIntrestRate /
numberOfCompounds), numberOfCompounds * years);
           totalInterest = futureValue - principal;

        }
```

```
   public void printRecord() {
   System.out.printf("\nFuture Value of Investment: ₹ %.2f\n", futureValue);
   System.out.printf("Total Interest Earned: ₹ %.2f\n", totalInterest);
    }

   public static class CompoundInterestCalculator {
         public static void main(String[] args) {
         Calculator ci = new Calculator();
        ci.acceptRecord();
        ci.calculateFutureValue();
        ci.printRecord();
         }
         }
}
```

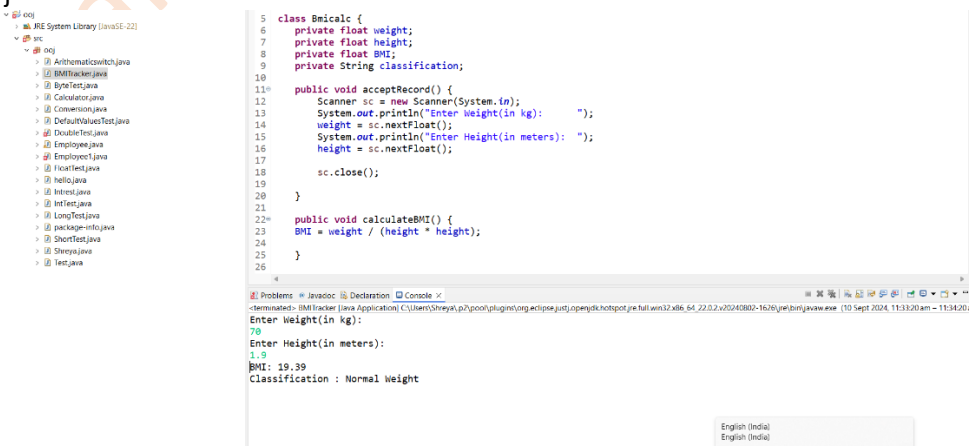
3.Create a system to calculate and classify Body Mass Index (BMI). The system should:

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.

```
package ooj;

import java.util.Scanner;

 class Bmicalc {
        private float weight;
        private float height;
        private float BMI;
        private String classification;

        public void acceptRecord() {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter Weight(in kg):     ");
                weight = sc.nextFloat();
                System.out.println("Enter Height(in meters):  ");
                height = sc.nextFloat();
```

```
            sc.close();
        }

        public void calculateBMI() {
        BMI = weight / (height * height);
        }

        public void classifyBMI() {
            if  (BMI < 18.5f) {
           classification = "Underweight";
          }else if (BMI >= 18.5 && BMI <24.9) {
           classification = "Normal Weight";
          }else if (BMI >= 25 && BMI < 29.9 ) {
           classification = "Overweight";
          }else if (BMI >= 30) {
           classification = "Obese";
          }

        }

        public void printRecord() {
            System.out.printf("BMI: %.2f%n", BMI );
          System.out.println("Classification : "  + classification);

        }
}
public class BMITracker {
        public static void main(String[] args) {
            Bmicalc calculator = new Bmicalc();
            calculator.acceptRecord();
            calculator.calculateBMI();
            calculator.classifyBMI();
            calculator.printRecord();
        }

}
```



4. Discount Calculation for Retail Sales

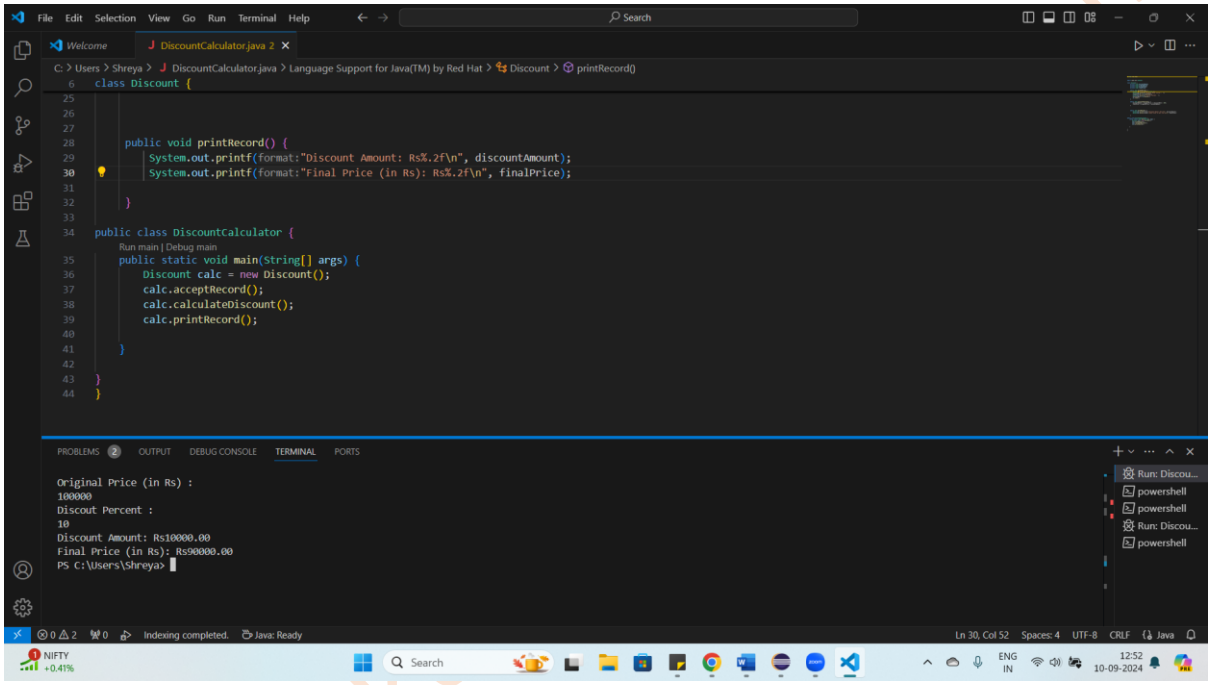
Design a system to calculate the final price of an item after applying a discount. The system should:

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.

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

class Discount {
	private float discountAmount;
	private float originalPrice;
	private float discountRate;
	private float finalPrice;

	public void acceptRecord() {
		Scanner sc = new Scanner(System.in);
		System.out.println("Original Price (in ₹) :  ");
		originalPrice = sc.nextFloat();
		System.out.println("Discout Percent : ");
		discountRate = sc.nextInt();
		sc.close();
	}

	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 (in ₹): ₹%.2f\n", finalPrice);

	}

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

```
        Discount calc = new Discount();
        calc.acceptRecord();
        calc.calculateDiscount();
        calc.printRecord();

    }

}
}
```



5. Toll Booth Revenue Management

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

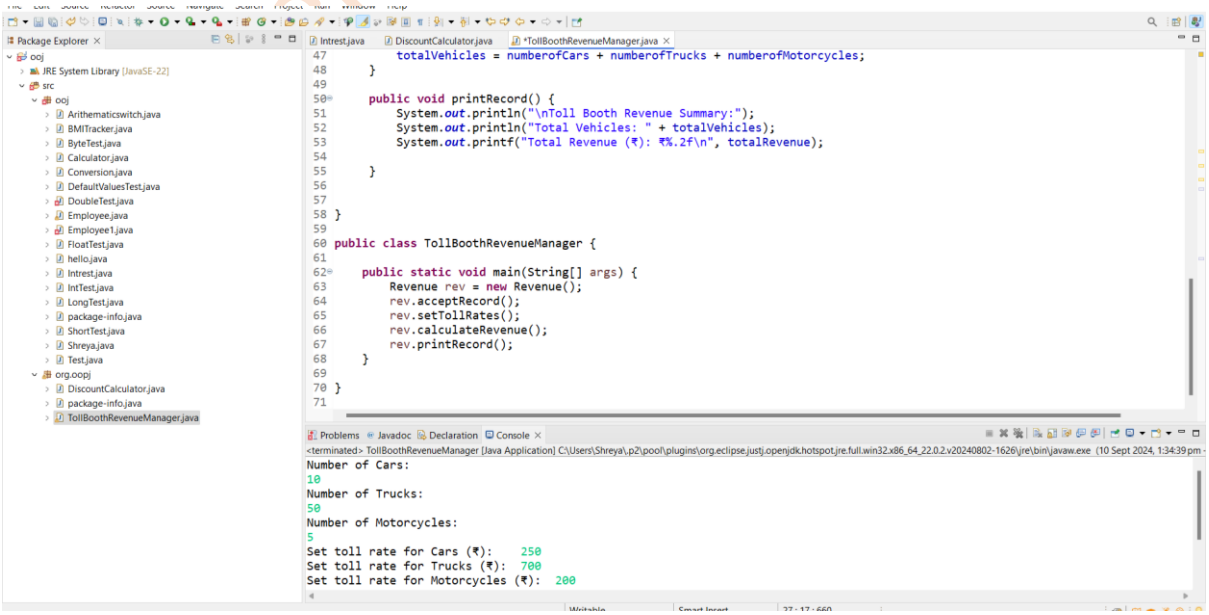
1. **Allow the user to set toll rates for different vehicle types: Car, Truck, and Motorcycle.**
2. **Accept the number of vehicles of each type passing through the toll booth.**
3. **Calculate the total revenue based on the toll rates and number of vehicles.**
4. **Display the total number of vehicles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

```
package org.oopj;

import java.util.Scanner;

class Revenue {
	private double carTollRate;
	private double TruckTollRate;
	private double MotorcycleTollRate;
	private int numberofCars;
	private int numberofTrucks;
	private int numberofMotorcycles;

	private double totalRevenue;
	private int totalVehicles;

	public void acceptRecord() {
	Scanner sc = new Scanner(System.in);
	System.out.println("Number of Cars: ");
	int cars = sc.nextInt();

	System.out.println("Number of Trucks: ");
	int trucks = sc.nextInt();

	System.out.println("Number of Motorcycles: ");
	int motorcycle = sc.nextInt();

	sc.close();

	}

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

	   System.out.print("Set toll rate for Cars (₹):    ");
	   carTollRate = sc.nextDouble();

	   System.out.print("Set toll rate for Trucks (₹):  ");
	   TruckTollRate = sc.nextDouble();

	   System.out.print("Set toll rate for Motorcycles (₹):  ");
	   MotorcycleTollRate = sc.nextDouble();
	   sc.close();
	}
```

```
	public void calculateRevenue() {
		totalRevenue = (numberofCars * carTollRate) +(numberofTrucks * 
TruckTollRate) +(numberofMotorcycles * MotorcycleTollRate);
	totalVehicles = numberofCars + numberofTrucks + numberofMotorcycles;
	}

	public void printRecord() {
		System.out.println("\nToll Booth Revenue Summary:");
	   System.out.println("Total Vehicles: " + totalVehicles);
	   System.out.printf("Total Revenue (₹): ₹%.2f\n", totalRevenue);

	}


}

public class TollBoothRevenueManager {

	public static void main(String[] args) {
		Revenue rev = new Revenue();
		rev.acceptRecord();
		rev.setTollRates();
		rev.calculateRevenue();
		rev.printRecord();
	}

}
```



ASSIGNMENT NO.3