ASSIGNMENT 4 Sandeep Sir

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 the class LoanAmortizationCalculator with fields, an appropriate constructor, getter and setter methods, a toString method and business logic methods. Define the class LoanAmortizationCalculatorUtil with methods acceptRecord, printRecord, and menuList. Define the class Program with a main method and test the functionality of the utility class.

```
//Program.java File

package org.example;
```

```
import java.util.Scanner;



public class Program {

    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        LoanAmortizationCalculatorUtil loan=new LoanAmortizatio
        loan.menuList();
    }

}
```

```
//LoanAmortizationCalculator.java File

package org.example;

class LoanAmortizationCalculator{
     private double principal;
     private double annualInterestRate;
     private int loanTerm;

     //constructor of LoanAmortizationCalculator
     public LoanAmortizationCalculator(double principal,double a
         this.principal=principal;
         this.annualInterestRate=annualInterestRate;
         this.loanTerm=loanTerm;
     }

    public double getPrincipal() {
        return principal;
    }
    public void setPrincipal(double principal) {
        this.principal = principal;
    }

    public double getAnnualInterestRate() {
        return annualInterestRate;
    }

    public void setAnnualInterestRate(double annualInterestRate
        this.annualInterestRate = annualInterestRate;
    }

    public int getLoanTerm() {
        return loanTerm;
    }

    public void setLoanTerm(int loanTerm) {
        this.loanTerm = loanTerm;
    }

    //calculate monthly payment
    public double calculateMonthlyPayment() {
        double monthlyInterestRate = (annualInterestRate / 100)
        int numberOfMonths = loanTerm * 12;
        double monthlyPayment=(principal * monthlyInterestRate 
                / (Math.pow(1 + monthlyInterestRate, numberOfMon
        return monthlyPayment;
    }

    //calculate total ammount paid
    public double calculateTotalAmount() {
        double monthlyPayment=calculateMonthlyPayment();
        double totalAmount= monthlyPayment*loanTerm*12;
        return totalAmount;
    }
    // toString method call directly from source
    @Override
    public String toString() {
        return "LoanAmortizationCalculator [principal=" + princi
                + ", loanTerm=" + loanTerm + "]";
    }




}
```

```
//LoanAmortizationCalculatorUtil.java File

package org.example;

import java.util.Scanner;


public class LoanAmortizationCalculatorUtil {
    Scanner sc=new Scanner(System.in);
    //variable LoanAmortizationCalculator
    private LoanAmortizationCalculator loan1=new LoanAmortizatio
    public void acceptRecord() {

        System.out.print("Principal Ammount:₹");
        loan1.setPrincipalAmount(sc.nextDouble());
        System.out.print("Annual interest rate:");
        double annualInterestRate=sc.nextDouble();
        System.out.print("Loan Term:");
        int loanTerm=sc.nextInt();

        //instance of LoanAmortizationCalculator
        //loan1=new LoanAmortizationCalculator(principal,annualI

    }

    void printRecord() {
        //System.out.println(loanAmortizationCalculator);
        double monthlyPayment=loan1.calculateMonthlyPayment();
        double totalAmount=loan1.calculateTotalAmount();
        System.out.println("Monthly Payment:₹"+monthlyPayment);
        System.out.println("Total amount:₹"+totalAmount);

    }
    public void menuList() {
        while(true) {
            System.out.println("1.Accept Record");
            System.out.println("2.Print Record");
            System.out.println("3.Exit");
            System.out.print("Enter option:");
            int choice=sc.nextInt();
            switch(choice) {
            case 1:
                acceptRecord();
                break;
            case 2:
                printRecord();
                break;
            case 3:
                System.out.println("Exit While");
                return;
                default:
                    System.out.println("Invalid Option");
            }
        }
    }
}
```

1. **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 the class CompoundInterestCalculator with fields, an appropriate constructor, getter and setter methods, a toString method and business logic methods. Define the class CompoundInterestCalculatorUtil with methods acceptRecord, printRecord, and menuList. Define the class Program with a main method to test the functionality of the utility class.

```
package org.example;

import java.util.Scanner;

class CompoundInterestCalculator{


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

    public  CompoundInterestCalculator(double principal,double a
        this.principal=principal;
        this.annualInterestRate=annualInterestRate;
        this.numberOfCompounds=numberOfCompounds;
        this.years=years;
    }

    public double getPrincipal() {
        return principal;
    }

    public void setPrincipal(double principal) {
        this.principal = principal;
    }

    public double getAnnualInterestRate() {
        return annualInterestRate;
    }

    public void setAnnualInterestRate(double annualInterestRate
        this.annualInterestRate = annualInterestRate;
    }

    public double getNumberOfCompounds() {
        return numberOfCompounds;
    }

    public void setNumberOfCompounds(double numberOfCompounds)
        this.numberOfCompounds = numberOfCompounds;
    }

    public int getYears() {
        return years;
    }
    public void setYears(int years) {
        this.years = years;
    }
    public double calculateFutureValue() {
        double futureValue = principal * Math.pow(1 + annualInte
        return futureValue;
    }
    public double calculateTotalInterest() {
        double futureValue=calculateFutureValue();
        double totalInterest = futureValue - principal;
        return totalInterest;
    }

    @Override
    public String toString() {
        return "CompoundInterestCalculator [principal=" + princi
                + ", numberOfCompounds=" + numberOfCompounds + "
    }


}

class CompoundInterestCalculatorUtil{
    Scanner sc=new Scanner(System.in);
    private CompoundInterestCalculator cmp;

     void acceptRecord() {
        System.out.print("Initial investment amount:");
        double  principal=sc.nextDouble();
        System.out.print("Annual interest rate:");
        double annualInterestRate=sc.nextDouble();
        System.out.print("Number Of Compounds:");
        double numberOfCompounds=sc.nextDouble();
        System.out.print("Investment duration Year:");
        int years=sc.nextInt();
        cmp=new CompoundInterestCalculator(principal,annualInte
    }
     void printRecord() {
        double futureValue=cmp.calculateFutureValue();
        double totalInterest=cmp.calculateTotalInterest();
        System.out.println("Future value:"+futureValue);
        System.out.println("Total interest earned:"+totalIntere
    }
    public void menuList() {
        while(true) {
            System.out.println("MENU LIST");
            System.out.println("1.ACCEPT RECORDS");
            System.out.println("2.SHOW CALCULATION");
            System.out.println("3.EXIT---");
            System.out.print("ENTER OPTION:");

            int choice=sc.nextInt();

            switch(choice) {
            case 1:
                acceptRecord();
                break;
            case 2:
                printRecord();
                break;
            case 3:
                System.out.println("EXITT");
                return;
            default:
                System.out.println("INVALID OPTION");
            }
        }
    }
}
public class Program {
    public static void main(String[] args) {
        CompoundInterestCalculatorUtil cmp1=new CompoundInterest
        cmp1.menuList();
        System.out.println(cmp1.toString());

    }

}
```

1. **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 the class BMITracker with fields, an appropriate constructor, getter and setter methods, a toString method, and business logic methods. Define the class BMITrackerUtil with methods acceptRecord, printRecord, and menuList. Define the class Program with a main method to test the functionality of the utility class.

```
//program

package com.assignment;
```

```java
public class Program {

    public static void main(String[] args) {
        BMITrackerUtil util=new BMITrackerUtil();
        int choice;

        while((choice = BMITrackerUtil.menuList())!=0) {
            switch(choice) {
            case 1:
                util.acceptRecord();
            case 2:
                util.printRecord();


            }

        }



    }

}
```

```java
package com.assignment;

public class BMITracker {

    private float weight;
    private float height;
    private float bmi;

    public BMITracker() {

    }
    public BMITracker(float weight, float height) {
        super();
        this.weight = weight;
        this.height = height;
    }
    public float getWeight() {
        return weight;
    }
    public void setWeight(float weight) {
        this.weight=weight;
    }
    public float getHeight() {
        return height;
    }
    public void setHeight(float height) {
        this.height=height;
    }
    public void calculateBMI() {
        this.bmi=this.weight/(this.height*this.height);
        if(this.bmi<18.5) {
            System.out.println("Underweight");
        }
        else if(bmi>=18.5 && bmi <24.9) {
            System.out.println("Normal Weight");

        }
        else if (bmi>=25 && bmi <29.9) {
            System.out.println("Overweight");
        }
        else {
            System.out.println("Obese");
        }
    }

    public String toString() {
        return "BMI:"+this.bmi;
    }



}
```

```
package com.assignment;

import java.util.Scanner;

public class BMITrackerUtil {
   private BMITracker bt=new BMITracker();

  private static Scanner sc=new Scanner(System.in);

   public void acceptRecord() {
       System.out.println("Weight in kg:");
       bt.setWeight(sc.nextFloat());
       System.out.println("Height in m:");
       bt.setHeight(sc.nextFloat());
      // bt.calculateBMI();
   }
   public void printRecord() {
       System.out.println(bt.toString());

   }
   public static int menuList() {
       System.out.println("0.Exit");
       System.out.println("1.Accept Record");
       System.out.println("2.Print Record");
       System.out.println("Enter Option:");
       int choice=sc.nextInt();
       return choice;
```

```
    }
}
```

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 the class DiscountCalculator with fields, an appropriate constructor, getter and setter methods, a toString method, and business logic methods. Define the class DiscountCalculatorUtil with methods acceptRecord, printRecord, and menuList. Define the class Program with a main method to test the functionality of the utility class.

```
package com.assignment4.question4;

import java.util.Scanner;

public class Question4 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        DiscountCalculatorUtil d1 = new DiscountCalculatorUtil(

        int choice;
        while ( (  choice = d1.menuList( sc ) ) != 0 ) {
            switch( choice ) {
            case 1:
                d1.acceptRecord(sc);
                break;
            case 2:
                d1.printRecord();
                break;
            }
        }
    }

}
```

```
package com.assignment4.question4;

public class DiscountCalculator {
    private double ogPrice;
    private double discountRate;
    private double discountAmount;
    private double finalPrice;

    public DiscountCalculator() {
        // TODO Auto-generated constructor stub
    }

    public double getOgPrice() {
        return ogPrice;
    }

    public void setOgPrice(double ogPrice) {
        this.ogPrice = ogPrice;
    }

    public double getDiscountRate() {
        return discountRate;
    }

    public void setDiscountRate(double discountRate) {
        this.discountRate = discountRate;
    }

    void calculateDiscount () {
        this.discountAmount = this.ogPrice * (this.discountRate
        this.finalPrice = this.ogPrice - this.discountAmount;
    }


    @Override
    public String toString() {
        calculateDiscount ();
        return "DiscountCalculator [discountAmount=" + discount
    }

}
```

```
package com.assignment4.question4;

import java.util.Scanner;

public class DiscountCalculatorUtil {
    private DiscountCalculator d = new DiscountCalculator();

    public static int menuList(Scanner sc) {
        System.out.println("\n");
        System.out.println("0.Exit.");
        System.out.println("1.Accept Record.");
        System.out.println("2.Print Record.");
        System.out.print("Enter choice  :   ");
        int choice = sc.nextInt( );
        return choice;
    }

    public void acceptRecord(Scanner sc) {
        System.out.println("ENTER Original Price: ");
        d.setOgPrice(sc.nextDouble());
        System.out.println("ENTER Discount Rate: ");
        d.setDiscountRate(sc.nextDouble());
    }

    public void printRecord() {
        System.out.println(d.toString());
    }
}
```

1. **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 the class TollBoothRevenueManager with fields, an appropriate constructor, getter and setter methods, a toString method, and business logic methods. Define the class TollBoothRevenueManagerUtil with methods acceptRecord, printRecord, and menuList. Define the class Program with a main method to test the functionality of the utility class.

```
package com.assignment4.question5;

import java.util.Scanner;

public class Question5 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        TollBoothRevenueManagerUtil d1 = new TollBoothRevenueMar

        int choice;
        while ( (  choice = d1.menuList( sc ) ) != 0 ) {
            switch( choice ) {
            case 1:
                d1.setTollRates(sc);
                break;
            case 2:
                d1.acceptRecord(sc);
                break;
            case 3:
                d1.printRecord();
                break;
            }
        }


    }

}
```

```
package com.assignment4.question5;

import java.util.Scanner;

public class TollBoothRevenueManager {
    private double carRate;
    private int carNumber;
    private double truckRate;
    private int truckNumber;
    private double motorcycleRate;
    private int motorcycleNumber;
    private double carRevenue;
    private double truckRevenue;
    private double motorcycleRevenue;

    public TollBoothRevenueManager() {
        // TODO Auto-generated constructor stub
    }

    void calculateRevenue() {
        this.carRevenue = this.carRate * this.carNumber;
        this.truckRevenue = this.truckRate * this.truckNumber;
        this.motorcycleRevenue = this.motorcycleRate * this.moto
    }


    public double getCarRate() {
        return carRate;
    }

    public void setCarRate(double carRate) {
        this.carRate = carRate;
    }

    public int getCarNumber() {
        return carNumber;
    }

    public void setCarNumber(int carNumber) {
        this.carNumber = carNumber;
    }

    public double getTruckRate() {
        return truckRate;
    }

    public void setTruckRate(double truckRate) {
        this.truckRate = truckRate;
    }

    public int getTruckNumber() {
        return truckNumber;
    }

    public void setTruckNumber(int truckNumber) {
        this.truckNumber = truckNumber;
    }

    public double getMotorcycleRate() {
        return motorcycleRate;
    }

    public void setMotorcycleRate(double motorcycleRate) {
        this.motorcycleRate = motorcycleRate;
    }

    public int getMotorcycleNumber() {
        return motorcycleNumber;
    }

    public void setMotorcycleNumber(int motorcycleNumber) {
        this.motorcycleNumber = motorcycleNumber;
    }

    @Override
    public String toString() {
        calculateRevenue();
        return "TollBoothRevenueManager [carRevenue=" + carReve
                + ", motorcycleRevenue=" + motorcycleRevenue + '
    }
}
```

```
package com.assignment4.question5;

import java.util.Scanner;

public class TollBoothRevenueManagerUtil {
    private TollBoothRevenueManager d = new TollBoothRevenueMana

    public static int menuList(Scanner sc) {
        System.out.println("\n");
        System.out.println("0.Exit.");
        System.out.println("1.Enter Rates.");
        System.out.println("2.Accept Record.");
        System.out.println("3.Print Record.");
        System.out.print("Enter choice  :   ");
        int choice = sc.nextInt( );
        return choice;
    }

    void setTollRates(Scanner sc){
        System.out.println("Enter Rate per Car: ");
        d.setCarRate(sc.nextDouble());
        System.out.println("Enter Rate per Truck: ");
        d.setTruckRate(sc.nextDouble());
        System.out.println("Enter Rate per Motorcycle: ");
        d.setMotorcycleRate(sc.nextDouble());
    }

    public void acceptRecord(Scanner sc) {
        System.out.println("Enter Number of Cars: ");
        d.setCarNumber(sc.nextInt());
        System.out.println("Enter Number of Truck: ");
        d.setTruckNumber(sc.nextInt());
        System.out.println("Enter Number of Motorcycle: ");
        d.setMotorcycleNumber(sc.nextInt());
    }

    public void printRecord() {
        System.out.println(d.toString());
    }


}
```