ASSIGNMENT NO.4

Note:

- The assignment is designed to practice constructor, getter/setter and toString method.
- Create a separate project for each question and create separate file for each class.
- Try to test the functionality by using menu-driven program.

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.

Ans:**package** Loan_Calculator_get_set_er;

```
import java.util.Scanner;
public class get_set_question1 {
  private double principal;
  private double annual;
  private float loanyear;
  private double monthlyPayment;
  private double totalPayment;
  static Scanner sc = new Scanner(System.in);
  public get_set_question1() {
     this(0, 0, 0);
  }
  public get_set_question1(double principal, double annual, float loanyear) {
     this.principal = principal;
     this.annual = annual;
     this.loanyear = loanyear;
```

```
    }
    public double getPrincipal() {
        return this.principal;
    }
    public double getAnnual() {
        return this.annual;
    }
    public float getLoanyear() {
        return this.loanyear;
    }
    public void setPrincipal(double principal) {
        this.principal = principal;
    }
    public void setAnnual(double annual) {
        this.annual = annual;
    }
    public void setLoanyear(float loanyear) {
        this.loanyear = loanyear;
    }
    public void calculate() {
        float numberOfMonths = this.loanyear * 12;
        double monthlyInterestRate = this.annual / 12 / 100;
        this.monthlyPayment = principal * (monthlyInterestRate * Math.pow(1 +
monthlyInterestRate, numberOfMonths)) /
                (Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1);
        this.totalPayment = this.monthlyPayment * numberOfMonths;
    }

    static class Loan {
        private get_set_question1 ln = new get_set_question1();
        public get_set_question1 getLn() {
            return ln;
        }
        public void accept() {
            System.out.println("Enter the Principal:");
            ln.setPrincipal(sc.nextDouble());
            System.out.println("Enter the annual interest rate:");
            ln.setAnnual(sc.nextDouble());
            System.out.println("Enter the loan years:");
            ln.setLoanyear(sc.nextFloat());
        }
        public void display() {
            ln.calculate(); // Call calculation before displaying
            System.out.println("Principal: " + ln.getPrincipal());
            System.out.println("Annual Interest Rate: " + ln.getAnnual());
            System.out.println("Loan Year: " + ln.getLoanyear());
            System.out.println("Monthly Payment: " + ln.monthlyPayment);
            System.out.println("Total Payment: " + ln.totalPayment);
        }
        public static int menuList() {
            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 static void main(String[] args) {
        int choice;
        Loan util = new Loan();
        while ((choice = Loan.menuList()) != 0) {
            switch (choice) {
                case 1:
                    util.accept();
                    break;
                case 2:
                    util.display();
                    break;
            }
        }
    }
}
```

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 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.

Ans:package Compound;

import java.util.Scanner;

ASSIGNMENT NO.4

```
class Compound {
        private double investment;
        private double annual;
        private double compounded;
        private double years;
        private double futureValue;
        private double totalInterest;


        public Compound() {
  this(0.0, 0.0, 0.0,0.0);
        }
        public Compound(double investment, double annual, double compounded,
double years) {
                this.investment = investment;
                this.annual = annual;
                this.compounded = compounded;
                this.years = years;
        }
        public void setinvestment(double investment) {
                this.investment = investment;
        }
        public void setannual(double annual) {
                this.annual = annual;
        }
        public void setcompounded(double compounded) {
                this.compounded = compounded;
        }
        public void setyears(double years) {
                this.years = years;
        }
        public double getannual() {
                return this.annual;
        }
        public double getinvestment() {
                return this.investment;
        }
        public double getyears() {
                return this.years;
        }

        public double getfutureValue() {
                return this.futureValue;
        }
        public double gettotalInterest() {
                return this.totalInterest;
        }
```

```
                    public void calculate() {
                            double rate = this.annual / 100;
                            this.futureValue = this.investment * Math.pow((1 + rate /
this.compounded), this.compounded * this.years);
                            this.totalInterest = futureValue - investment;
                    }

}

            class helper {
                    private  Compound c = new Compound();
                    public  Compound getc() {
                            return c;
                    }
                    private static Scanner sc = new Scanner(System.in);
                    public void accept() {
                            // Scanner sc=new Scanner(System.in);
                            System.out.println("Enter the investment");
                            c.setinvestment(sc.nextDouble());
//                          this.investment = sc.nextDouble();
                            System.out.println("Enter the annual");
                            c.setannual(sc.nextDouble());
//                          this.annual = sc.nextDouble();
                            System.out.println("Enter the compounded");
                            c.setcompounded(sc.nextDouble());
//                          this.compounded = sc.nextDouble();
                            System.out.println("Enter the years");
                            c.setyears(sc.nextDouble());
//                          this.years = sc.nextDouble();
                            c.calculate();
                    }
                    public void Print() {
                            System.out.println("The Montly paymenr is " +
c.getfutureValue());
                            System.out.println("The total payment is " + c.gettotalInterest());

//                          System.out.println("The Future Value is " + c.futureValue);
//                          System.out.println("The total Interest is " + c.totalInterest);
                    }

                    public static int menulist() {

                             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 class Question2c {
    public static void main(String[] args) {
        int choice;
        helper cc=new helper();
        while((choice=helper.menulist())!=0) {
            switch (choice) {
            case 1:
                cc.accept();

                break;
            case 2:
            cc.Print();
                break;
            }

        }
    }
}
```

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 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.

```
Ans:package BMI_get_set;
import java.util.Scanner;
class bmi{
    private double weight ;
    private double height;
private  double Bmi ;
public bmi() {
        this(0,0,0);

}
public bmi(double weight,double height,double BMI) {
        this.weight=weight;
        this.height=height;
        this.Bmi=BMI;

}
public void setweight(double weight) {
        this.weight=weight;
}
public void setheight(double height) {
        this.height=height;
}
public void setBMI(double BMI) {
        this.Bmi=BMI;
}
public double getweight() {
        return this.weight;
}
public double getheight() {
        return this.height;
}
public double getBMI() {
        return this.Bmi;
}
public double getBmi() {
        return this.Bmi;
        }
public void Calculation() {
         this. Bmi = weight / (height * height);
         if (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 if (Bmi >= 30) {
       System.out.println("Obese");
    } else {
       System.out.println("Invalid BMI value.");
    }
  }
}
class helper{
        private bmi bm=new bmi();
        static Scanner sc =new Scanner(System.in);
    public bmi getbmi() {
        return bm;
    }

    public void accept() {
            //  Scanner sc=new Scanner(System.in);
             System.out.println("Enter the weight in kg");
            bm. setweight(sc.nextDouble());
             System.out.println("Enter the height in meter");
                bm. setheight(sc.nextDouble());
                bm.Calculation();



            }

    public void Print() {
System.out.println("The BMI paymenr is " + bm.getBmi());
    }
    public static int menulist() {

        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 class Question3b {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int choice;
        helper cc=new helper();
        while((choice=helper.menulist())!=0) {
            switch (choice) {
            case 1:
                cc.accept();

                break;
            case 2:
            cc.Print();
                break;
            }

        }
        }
}
```

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 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.

```
Ans:package Discount;
import java.util.Scanner;
class discount{
        private double original ;
        private double discount;
private  double discountAmount ;
        private  double finalPrice ;


        public discount() {
                this(0.0,0.0);
        }

        public discount(double original ,double discount){
                this.original=original;
                this.discount=discount;

        }

        public  double getoriginal() {
                return original;
        }
        public  double getdiscount() {
                return discount;
        }
        public  double getdiscountAmount() {
                return discountAmount;
        }
        public  double getfinalPrice() {
                return finalPrice;
        }
        public void setoriginal(double original) {
                this.original=original;
        }
        public void setdiscount(double discount) {
                this.discount=discount;
        }

        public void Calculation() {
                 this. discountAmount= this.original * (this.discount / 100);
                 this.finalPrice = this.original - this.discountAmount;
}

}
class helper{
        static  Scanner sc = new Scanner(System.in);
        private discount ds= new discount();
        public discount getdis() {
                return ds;
        }

        public void accept() {
                System.out.println("Enter the Original Price");
                ds.setoriginal(sc.nextDouble());
                System.out.println("Enter the discount Price");
                ds.setdiscount(sc.nextDouble());
                ds.Calculation();

        }

        public void Print() {
                System.out.println("discount price : "+ ds.getdiscountAmount());
                System.out.println("final price : "+ ds.getfinalPrice());
        }
        public static int menulist() {

                 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 class question4d {
        public static void main(String[] args) {
                // TODO Auto-generated method stub

                int choice;
                helper cc=new helper();
                while((choice=helper.menulist())!=0) {
                    switch (choice) {
                    case 1:
                        cc.accept();

                        break;
                    case 2:
                    cc.Print();
                        break;
                    }

                }
        }
}
```