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 java_assignment_4_ques1;

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

        public LoanAmortizationCalculator() {
                this(0.0, 0.0, 0);
        }
        public LoanAmortizationCalculator(double principal, double annualInterestRate) {
             this(principal, annualInterestRate, 0);
   }
        public LoanAmortizationCalculator(double principal, double annualInterestRate, int
loanTerm) {
                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;
   }

         public double calculateMonthlyPayment() {
            double monthlyInterestRate = annualInterestRate / 12 / 100;
            int numberOfMonths = loanTerm * 12;
            if (monthlyInterestRate == 0) {
               return principal / numberOfMonths;
            }
            return principal * (monthlyInterestRate * Math.pow(1 + monthlyInterestRate,
numberOfMonths)) /
                  (Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1);
         }

         // Method to calculate total amount paid
         public double calculateTotalAmountPaid() {
            return calculateMonthlyPayment() * loanTerm * 12;
         }

         // toString method to display loan details
         @Override
         public String toString() {
            return String.format("Principal Amount: ₹%.2f\nAnnual Interest Rate:
%.2f%%\nLoan Term: %d years",
                  principal, annualInterestRate, loanTerm);
         }


}



package java_assignment_4_ques1;

import java.util.Scanner;
```

```
public class LoanAmortizationCalculatorUtil {

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

		 System.out.print("Enter the principal amount (₹): ");
		double principal = sc.nextDouble();

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

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

		return new LoanAmortizationCalculator(principal, annualInterestRate,
loanTerm);

	}

	public static void printRecord(LoanAmortizationCalculator calculator) {
		System.out.println("\nLoan Details:");
	System.out.println(calculator);
	double monthlyPayment = calculator.calculateMonthlyPayment();
	double totalAmountPaid = calculator.calculateTotalAmountPaid();
	System.out.printf("Monthly Payment: ₹%.2f\n", monthlyPayment);
	System.out.printf("Total Amount Paid Over the Loan Term: ₹%.2f\n",
totalAmountPaid);

	}

}



package java_assignment_4_ques1;

import java.util.Scanner;

public class Program {

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		LoanAmortizationCalculatorUtil util = new
LoanAmortizationCalculatorUtil();
		LoanAmortizationCalculator calculator = null;

		boolean running = true;
		while(running) {
			System.out.println("1. Enter Loan Details");
```

```
                System.out.println("2. Exit");
                System.out.print("Choose an option: ");
                int option = sc.nextInt();

                switch (option) {
                        case 1: {
                                calculator =
LoanAmortizationCalculatorUtil.acceptRecord();

        LoanAmortizationCalculatorUtil.printRecord(calculator);
                                break;
                        }

                        case 2:
                                running = false;
                                System.out.println("Exiting the program...");
                                break;
                        default:
                                System.out.println("Invalid option choose again.");
                        }
                }
        }

}
```

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.

ASSIGNMENT NO.4

Ans.

```
package java_assignment_4_ques2;

public class CompoundInterestCalculator {
        double principal;
        double annualInterestRate;
        int numberOfCompounds;
        int years;


         public CompoundInterestCalculator(double principal, double annualInterestRate, int
numberOfCompounds, int years) {
            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 int getNumberOfCompounds() {
            return numberOfCompounds;
          }

          public void setNumberOfCompounds(int numberOfCompounds) {
            this.numberOfCompounds = numberOfCompounds;
          }

          public int getYears() {
            return years;
          }

          public void setYears(int years) {
            this.years = years;
          }
```

```
        // Method to calculate future value
        public double calculateFutureValue() {
            double ratePerPeriod = annualInterestRate / numberOfCompounds;
            double totalPeriods = numberOfCompounds * years;
            return principal * Math.pow(1 + ratePerPeriod, totalPeriods);
        }

        // Method to calculate total interest
        public double calculateTotalInterest() {
            return calculateFutureValue() - principal;
        }

        // toString method
        @Override
        public String toString() {
            return String.format("Principal: ₹%.2f, Annual Interest Rate: %.2f%%, Number
of Compounds per Year: %d, Duration: %d years",
                                 principal, annualInterestRate * 100, numberOfCompounds, years);
        }
}
```

```
package java_assignment_4_ques2;

import java.util.Scanner;

public class CompoundInterestCalculatorUtil {



    public static void menuList() {
                    System.out.println("Menu:");
        System.out.println("1. Calculate Compound Interest");
        System.out.println("2. Exit");

    }

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

            System.out.println("Enter the initial investment amount : " );
            double principal = sc.nextDouble();
            System.out.print("Enter the annual interest rate (in decimal form, e.g., 0.05 for
5%): ");
    double annualInterestRate = sc.nextDouble();

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

```
        System.out.print("Enter the investment duration (in years): ");
        int years = sc.nextInt();
        return new CompoundInterestCalculator(principal, annualInterestRate,
numberOfCompounds, years);
            }

            public static void printRecord(CompoundInterestCalculator calculator) {
                    double futureValue = calculator.calculateFutureValue();
        double totalInterest = calculator.calculateTotalInterest();

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

            }

}


package java_assignment_4_ques2;

import java.util.Scanner;

public class Program {

            public static void main(String[] args) {
                    Scanner sc = new Scanner(System.in);
                    while(true) {
                            CompoundInterestCalculatorUtil.menuList();
                            System.out.println("choose an option: ");
                            int choice = sc.nextInt();

                            switch(choice) {
                            case 1:
                CompoundInterestCalculator calculator =
CompoundInterestCalculatorUtil.acceptRecord();
                                    CompoundInterestCalculatorUtil.printRecord(calculator);
                                    break;
                            case 2:
                                    System.out.println("Exiting..");
                                    sc.close();
                                    return;
                            default:
                                    System.out.println("Invalid choice. Please try again.");
                            }
                    }
```

```
    }

}
```

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 java_assignment_4_ques3;

public class BMITracker {

        double weight;
        double height;


         public BMITracker(double weight, double height) {
                        this.weight = weight;
                        this.height = height;
                }


        public double getWeight() {
                return weight;
        }


        public void setWeight(double weight) {
                this.weight = weight;
```

```
	}


	public double getHeight() {
		return height;
	}


	public void setHeight(double height) {
		this.height = height;
	}



	public double getBMICalculation() {
		return weight/(height*height);

	}


	public String getBMICategory() {
		double bmi = getBMICalculation();
		if(bmi<18.5) {
			return "Underweight";
		}
		else if(bmi<24.9) {
			return "Normal Weight";
		}
		else if(bmi<29.9) {
			return "Overweight";
		}
		else {
			return "Obese";
		}
	}
	 @Override
	   public String toString() {
	      return String.format("Weight: %.2f kg, Height: %.2f m", weight, height);
	   }


}


package java_assignment_4_ques3;

import java.util.Scanner;

public class BMITrackerUtil {
	public static BMITracker acceptRecord() {
```

```
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter the weight : ");
		double weight = sc.nextDouble();
		System.out.println("Enter the height : ");
		double height = sc.nextDouble();
		return new BMITracker(weight, height) ;
	}

	public static void printRecord(BMITracker bmi) {
		double bmiValue = bmi.getBMICalculation();
		String bmiClassification = bmi.getBMICategory();
		System.out.println(bmiValue);
		System.out.println(bmiClassification);
	}



	public static void menuList() {
		System.out.println("Menu : ");
		System.out.println("1.Calculate bmi : ");
		System.out.println("2.Exit : ");
	}

}
package java_assignment_4_ques3;

import java.util.Scanner;

public class Program {

	public static void main(String[] args) {

		Scanner sc = new Scanner(System.in);
		while(true) {
			BMITrackerUtil.menuList();
			System.out.println("Enter the choice : ");
			int choice = sc.nextInt();

			switch(choice) {

			case 1:
				BMITracker bmi =  BMITrackerUtil.acceptRecord();
				BMITrackerUtil.printRecord(bmi);
				break;
			case 2:
				System.out.println("Exiting..");
				sc.close();
				return;
			default:
				System.out.println("Invalid choice. Please try again.");
```

```
                }
            }

        }

}
```

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 java_assignment_4_ques4;

public class DiscountCalculator {
        int originalPrice;
        double discountPercentage;

        public DiscountCalculator(int price , double discount) {
                this.originalPrice = price;
                this.discountPercentage = discount;
        }
        public int getOriginalPrice() {
                return originalPrice;
        }
        public void setOriginalPrice(int originalPrice) {
                this.originalPrice = originalPrice;
        }
        public double getDiscountPercentage() {
                return discountPercentage;
        }
        public void setDiscountPercentage(double discountPercentage) {
```

```
		this.discountPercentage = discountPercentage;
	}

	public double getDiscountedPrice() {
		return originalPrice*(discountPercentage/100);
	}
	public double getFinalPrice() {
		double discountedPrice = getDiscountedPrice();
		double finalPrice = originalPrice - discountedPrice;
		return finalPrice;
	}

	@Override
   public String toString() {
      return String.format("original Price: %.2f ₹, discount: %.2f %", originalPrice,
discountPercentage);
   }


}

package java_assignment_4_ques4;

import java.util.Scanner;

public class DiscountCalculatorUtil {

	public static void menuList() {
		System.out.println("Menu : ");
		System.out.println("1. Discount Amount Calculation and final price
calculation : ");
		System.out.println("2. Exit : ");

	}

	public static DiscountCalculator acceptRecord() {
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter the price of item :  " );
		int price = sc.nextInt();
		System.out.println("Enter the discount percentage :  " );
		double discount = sc.nextDouble();
		return new DiscountCalculator(price , discount);
	}

	public static void printRecord(DiscountCalculator disc) {
		double discountPrice = disc.getDiscountedPrice();
		double finalPrice = disc.getFinalPrice();
		System.out.println("The discounted amount =  ₹ " + discountPrice);
		System.out.println("The final price = ₹ " + finalPrice);
```

```
	}

}
```

```
package java_assignment_4_ques4;

import java.util.Scanner;


public class Program {

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		while(true) {
			DiscountCalculatorUtil.menuList();
			System.out.println("Enter the choice : ");
			int choice = sc.nextInt();

			switch(choice) {

			case 1:
				DiscountCalculator disc =
DiscountCalculatorUtil.acceptRecord();
				DiscountCalculatorUtil.printRecord(disc);
				break;
			case 2:
				System.out.println("Exiting..");
				sc.close();
				return;
			default:
				System.out.println("Invalid choice. Please try again.");

			}
		}


	}
}
```

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 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.

Ans.

```
package java_assignment_4_ques5;

public class TollBoothRevenueManager {
		int carCount;
		int truckCount;
		int motorcycleCount;
		double carTollRate;
		double truckTollRate;
		double motorcycleTollRate;


		public TollBoothRevenueManager(int carCount, int truckCount, int
motorcycleCount) {
			this.carCount = carCount;
			this.truckCount = truckCount;
			this.motorcycleCount = motorcycleCount;
			this.carTollRate = 50;
			this.truckTollRate = 100;
			this.motorcycleTollRate = 30;
		}
		public int getCarCount() {
			return carCount;
		}
		public void setCarCount(int carCount) {
			this.carCount = carCount;
		}
		public int getTruckCount() {
			return truckCount;
		}
		public void setTruckCount(int truckCount) {
			this.truckCount = truckCount;
```

```
	}
	public int getMotorcycleCount() {
		return motorcycleCount;
	}
	public void setMotorcycleCount(int motorcycleCount) {
		this.motorcycleCount = motorcycleCount;
	}
	public double getCarTollRate() {
		return carTollRate;
	}
	public void setCarTollRate(double carTollRate) {
		this.carTollRate = carTollRate;
	}
	public double getTruckTollRate() {
		return truckTollRate;
	}
	public void setTruckTollRate(double truckTollRate) {
		this.truckTollRate = truckTollRate;
	}
	public double getMotorcycleTollRate() {
		return motorcycleTollRate;
	}
	public void setMotorcycleTollRate(double motorcycleTollRate) {
		this.motorcycleTollRate = motorcycleTollRate;
	}
	public int getTotalVehicleCount() {
		return carCount+truckCount+motorcycleCount;
	}
	public double getRevenueCollected() {
		int cars = getCarCount();
		double carsToll = getCarTollRate();
		int trucks = getTruckCount();
		double trucksToll = getTruckTollRate();
		int motorcycle = getMotorcycleCount();
		double motorcycleToll = getMotorcycleTollRate();

		double revenue  =
(cars*carsToll)+(trucks*trucksToll)+(motorcycle*motorcycleToll);
		return revenue;

	}
}
```

```
package java_assignment_4_ques5;

import java.util.Scanner;
```

```
public class TollBoothRevenueManagerUtil {

	public static void menuList() {
		System.out.println("Menu : ");
		System.out.println("1. total number of vehicles and total revenue collected :
");
		System.out.println("2. Exit : ");

	}

	public static TollBoothRevenueManager acceptRecord() {
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter the number of cars :");
		int carCount = sc.nextInt();
		System.out.println("Enter the number of Trucks :");
		int truckCount = sc.nextInt();
		System.out.println("Enter the number of Motorcycles :");
		int motorcycleCount = sc.nextInt();
		return new TollBoothRevenueManager( carCount, truckCount,
motorcycleCount);
	}

	public static void printRecord(TollBoothRevenueManager disc) {
		int vehicleCount = disc.getTotalVehicleCount();
		double revenueCollected = disc.getRevenueCollected();
		System.out.println("The total number of vehicles = " + vehicleCount);
		System.out.println("Total revenue Collected = " + revenueCollected);

	}



}
```

```
package java_assignment_4_ques5;

import java.util.Scanner;


public class Program {

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
```

```
                while(true) {
                        TollBoothRevenueManagerUtil.menuList();
                        System.out.println("Enter the choice : ");
                        int choice = sc.nextInt();

                        switch(choice) {

                        case 1:
                                TollBoothRevenueManager disc =
TollBoothRevenueManagerUtil.acceptRecord();
                                TollBoothRevenueManagerUtil.printRecord(disc);
                                break;
                        case 2:
                                System.out.println("Exiting..");
                                sc.close();
                                return;
                        default:
                                System.out.println("Invalid choice. Please try again.");

                        }
                }

        }

}
```