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

Display the monthly payment and the total amount paid over the life of the loan, in Indian Rupees (₹).

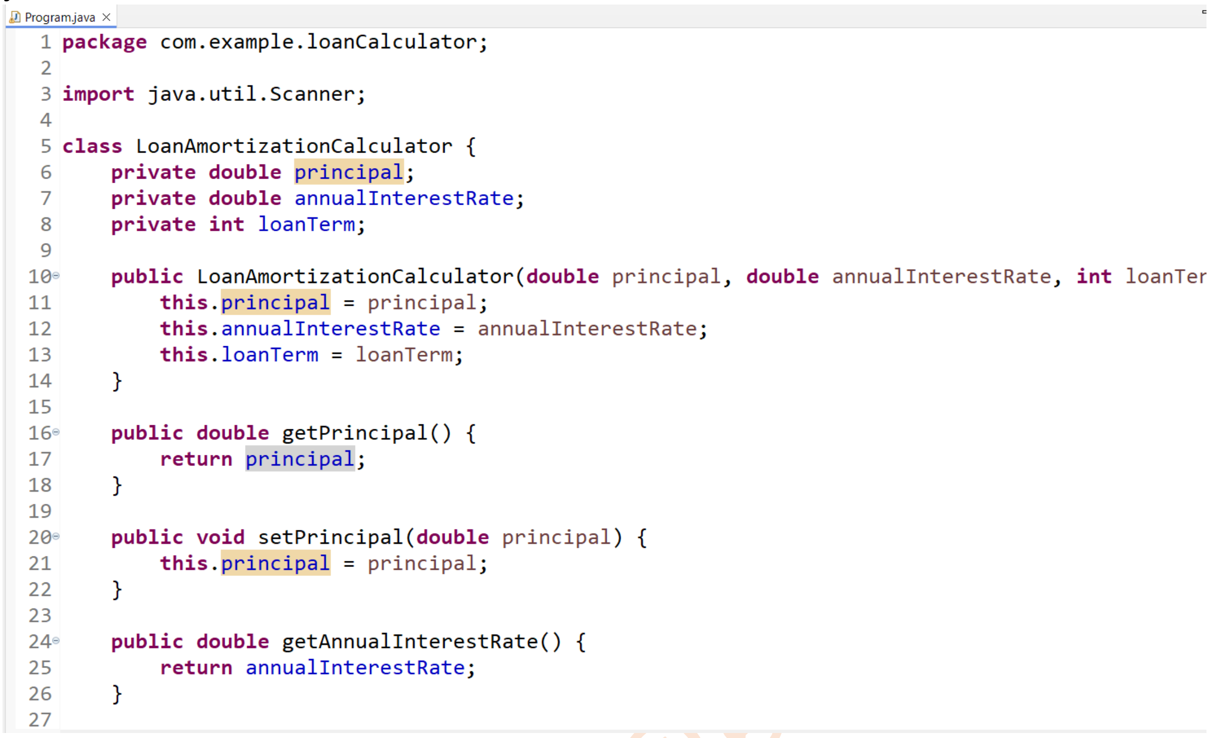
Define the class `LoanAmortizationCalculator` with fields, an appropriate constructor, getter and setter methods, a `toString` method and business logic methods. Define the class `LoanAmortizationCalculatorUtil` with methods `acceptRecord`, `printRecord`, and `menuList`. Define the class `Program` with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

Ans:

```
package com.example.loanCalculator;
import java.util.Scanner;
class LoanAmortizationCalculator {
private double principal;
 private double annualInterestRate;
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;
  }
```

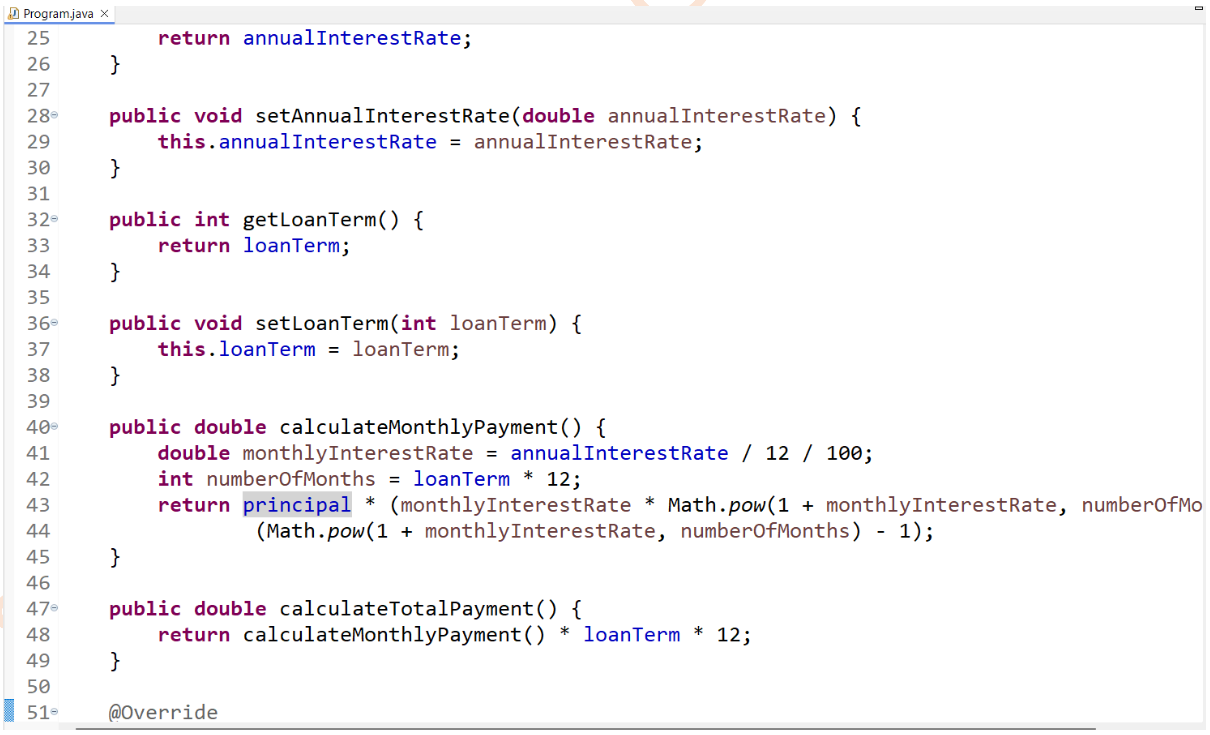
```
public double getAnnualInterestRate() {
return annualInterestRate;
 }
 public void setAnnualInterestRate(double annualInterestRate) {
this.annualInterestRate = annualInterestRate;
return loanTerm;
}
public void setLoanTerm(int loanTerm) {
this.loanTerm = loanTerm
 }
public double calculateMonthlyPayment() {
 double monthlyInterestRate = annualInterestRate / 12 / 100;
 int numberOfMonths = loanTerm * 12;
return principal * (monthlyInterestRate * Math.pow(1 + monthlyInterestRate,
numberOfMonths)) /
 (Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1);
}
public double calculateTotalPayment() {
 return calculateMonthlyPayment() * loanTerm * 12;
  }
@Override
public String toString() {
return "LoanAmortizationCalculator{" +"principal=" + principal +

               ", annualInterestRate=" + annualInterestRate +

               ", loanTerm=" + loanTerm +

               '}';
 }
```

}



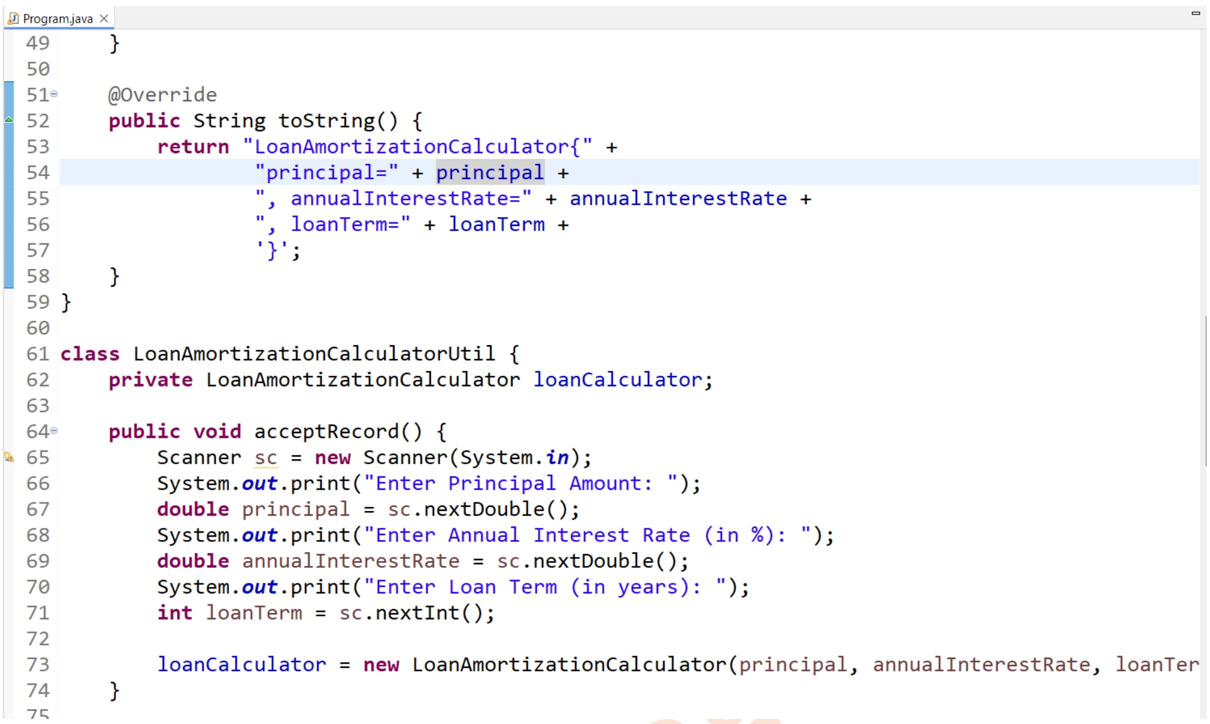
Program.java

```java
package com.example.loanCalculator;

import java.util.Scanner;

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

    public LoanAmortizationCalculator(double principal, double annualInterestRate, int loanTer
        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;
    }

```



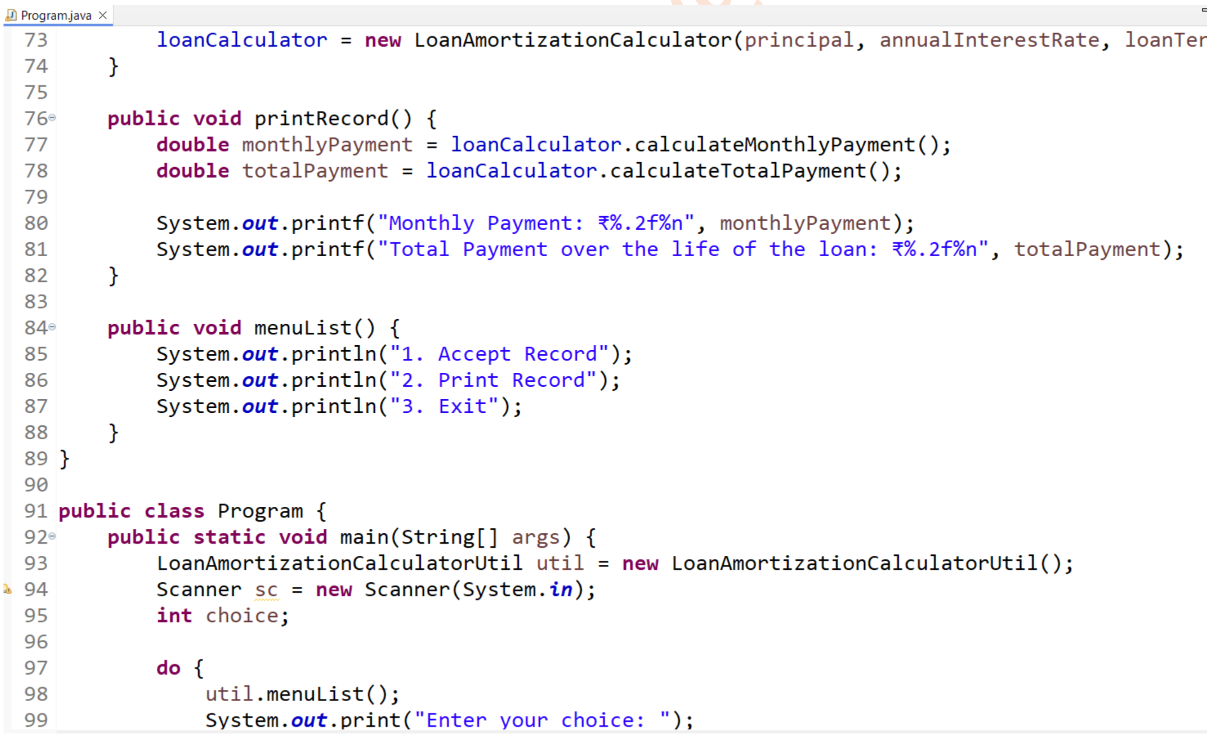
Program.java

```java
        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;
        return principal * (monthlyInterestRate * Math.pow(1 + monthlyInterestRate, numberOfMo
                (Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1);
    }

    public double calculateTotalPayment() {
        return calculateMonthlyPayment() * loanTerm * 12;
    }

    @Override
```



Program.java

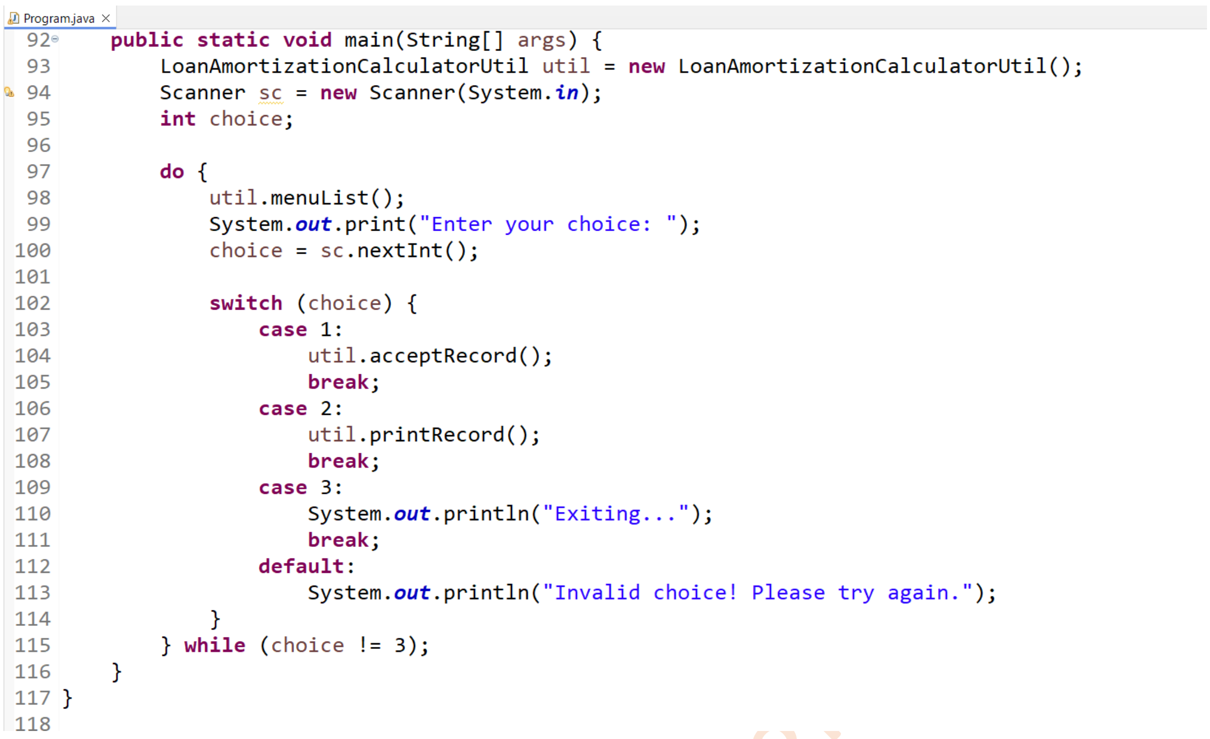
```java
    }

    @Override
    public String toString() {
        return "LoanAmortizationCalculator{" +
                "principal=" + principal +
                ", annualInterestRate=" + annualInterestRate +
                ", loanTerm=" + loanTerm +
                '}';
    }
}

class LoanAmortizationCalculatorUtil {
    private LoanAmortizationCalculator loanCalculator;

    public void acceptRecord() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Principal Amount: ");
        double principal = sc.nextDouble();
        System.out.print("Enter Annual Interest Rate (in %): ");
        double annualInterestRate = sc.nextDouble();
        System.out.print("Enter Loan Term (in years): ");
        int loanTerm = sc.nextInt();

        loanCalculator = new LoanAmortizationCalculator(principal, annualInterestRate, loanTer
    }
```



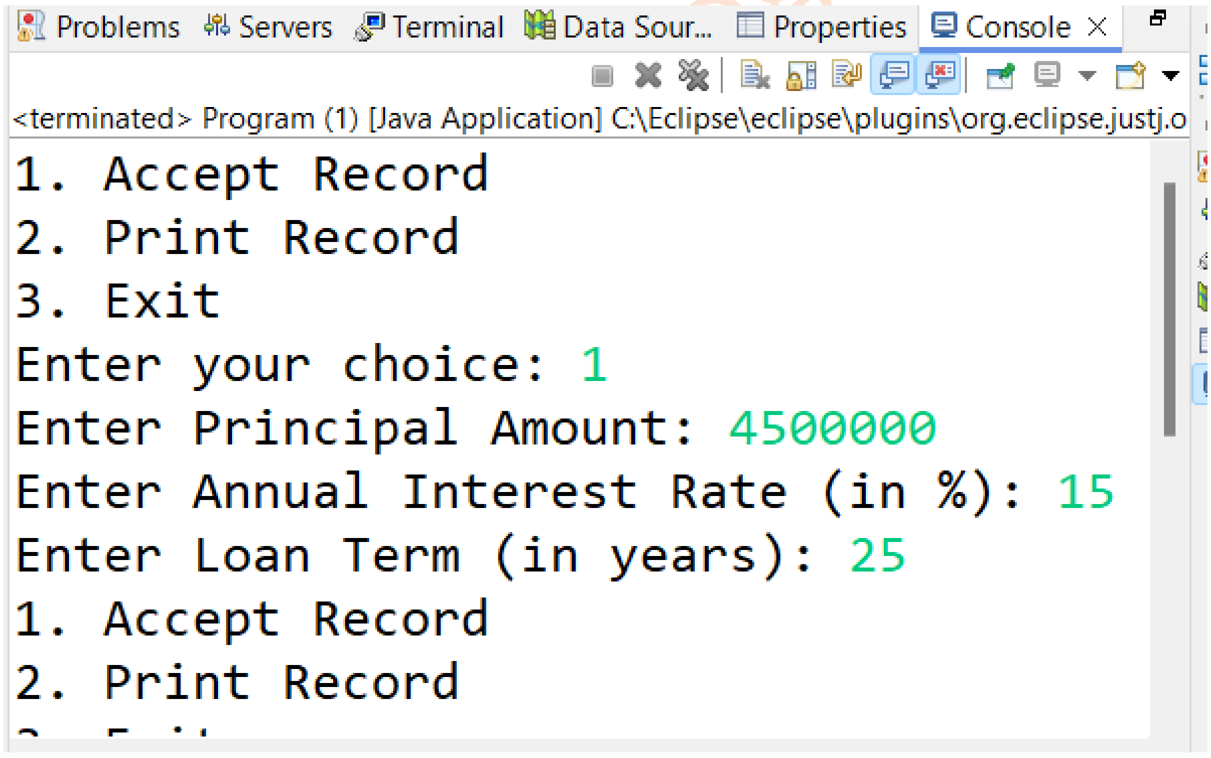
Program.java

```java
        loanCalculator = new LoanAmortizationCalculator(principal, annualInterestRate, loanTer
    }

    public void printRecord() {
        double monthlyPayment = loanCalculator.calculateMonthlyPayment();
        double totalPayment = loanCalculator.calculateTotalPayment();

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

    public void menuList() {
        System.out.println("1. Accept Record");
        System.out.println("2. Print Record");
        System.out.println("3. Exit");
    }
}

public class Program {
    public static void main(String[] args) {
        LoanAmortizationCalculatorUtil util = new LoanAmortizationCalculatorUtil();
        Scanner sc = new Scanner(System.in);
        int choice;

        do {
            util.menuList();
            System.out.print("Enter your choice: ");
```

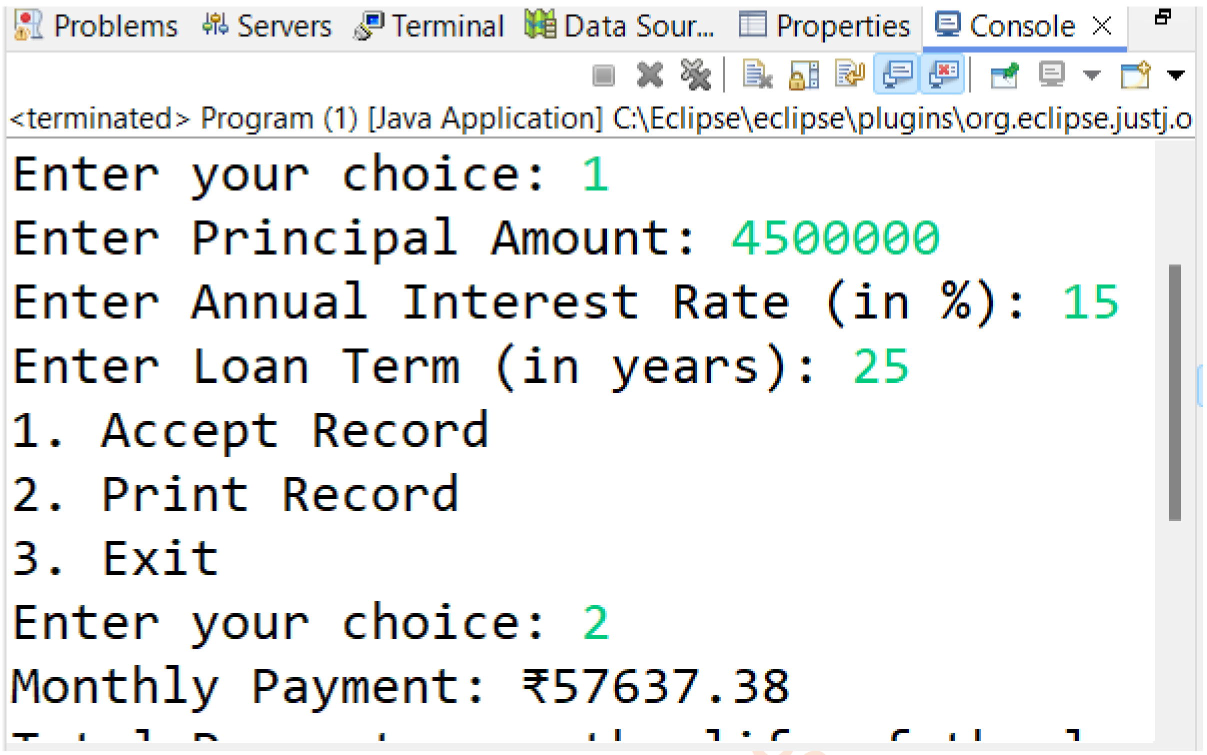
ASSIGNMENT NO.4



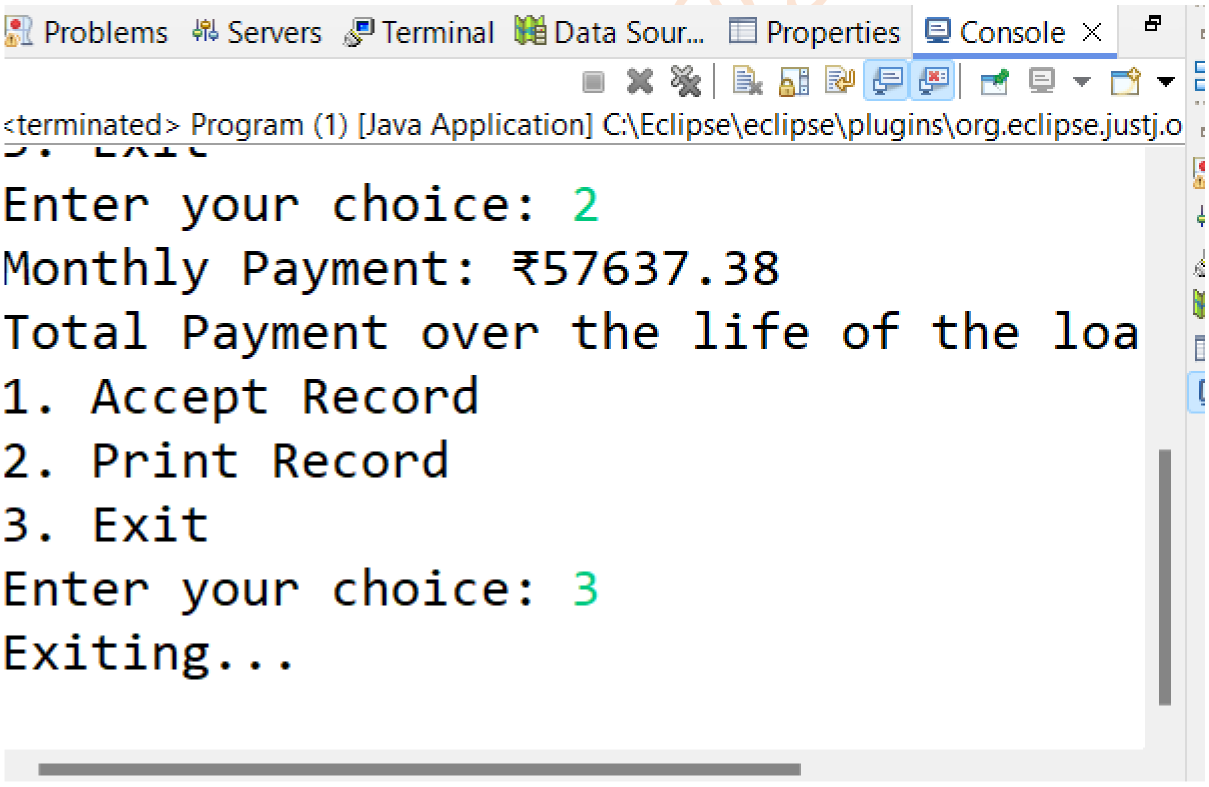
```java
    public static void main(String[] args) {
        LoanAmortizationCalculatorUtil util = new LoanAmortizationCalculatorUtil();
        Scanner sc = new Scanner(System.in);
        int choice;

        do {
            util.menuList();
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    util.acceptRecord();
                    break;
                case 2:
                    util.printRecord();
                    break;
                case 3:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice! Please try again.");
            }
        } while (choice != 3);
    }
}
```



```
<terminated> Program (1) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.o
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 1
Enter Principal Amount: 4500000
Enter Annual Interest Rate (in %): 15
Enter Loan Term (in years): 25
1. Accept Record
2. Print Record
```



```
<terminated> Program (1) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.o
Enter your choice: 1
Enter Principal Amount: 4500000
Enter Annual Interest Rate (in %): 15
Enter Loan Term (in years): 25
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 2
Monthly Payment: ₹57637.38
```



```
<terminated> Program (1) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.o
Enter your choice: 2
Monthly Payment: ₹57637.38
Total Payment over the life of the loa
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 3
Exiting...
```

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`

Display the future value and the total interest earned, in Indian Rupees (₹).

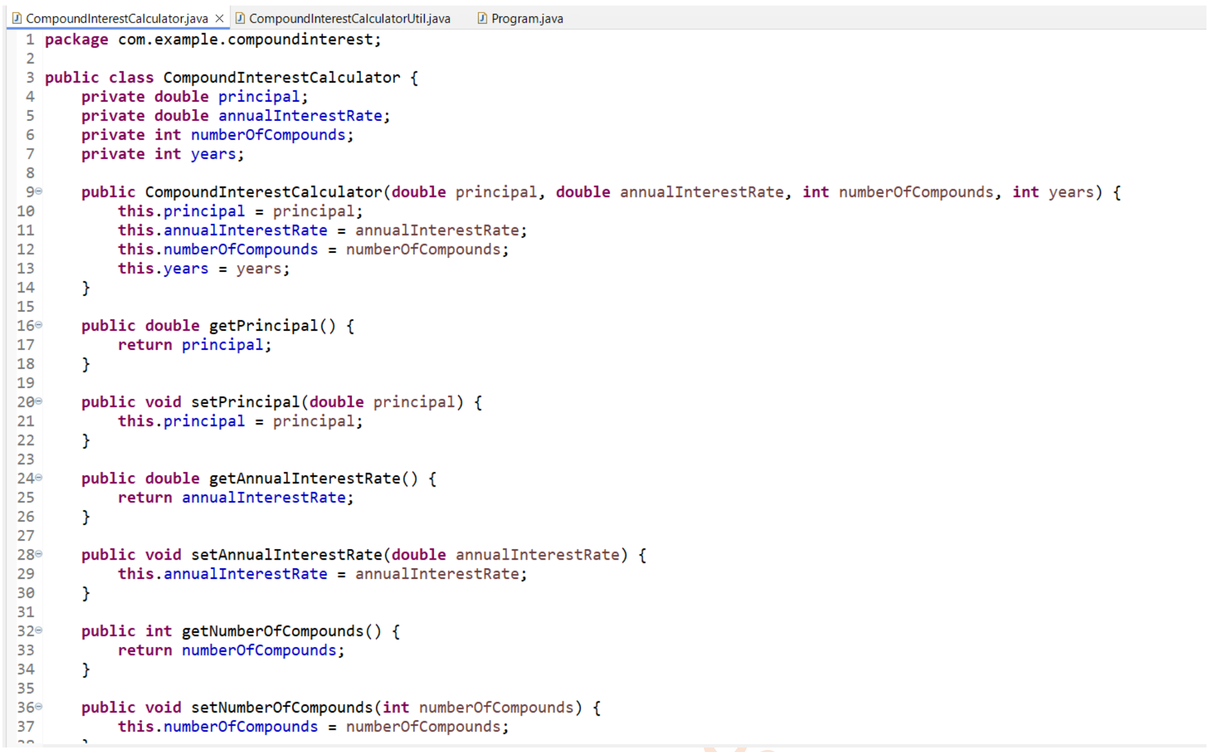
Define the class `CompoundInterestCalculator` with fields, an appropriate constructor, getter and setter methods, a `toString` method and business logic methods. Define the class `CompoundInterestCalculatorUtil` with methods `acceptRecord`, `printRecord`, and `menuList`. Define the class `Program` with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

Ans:

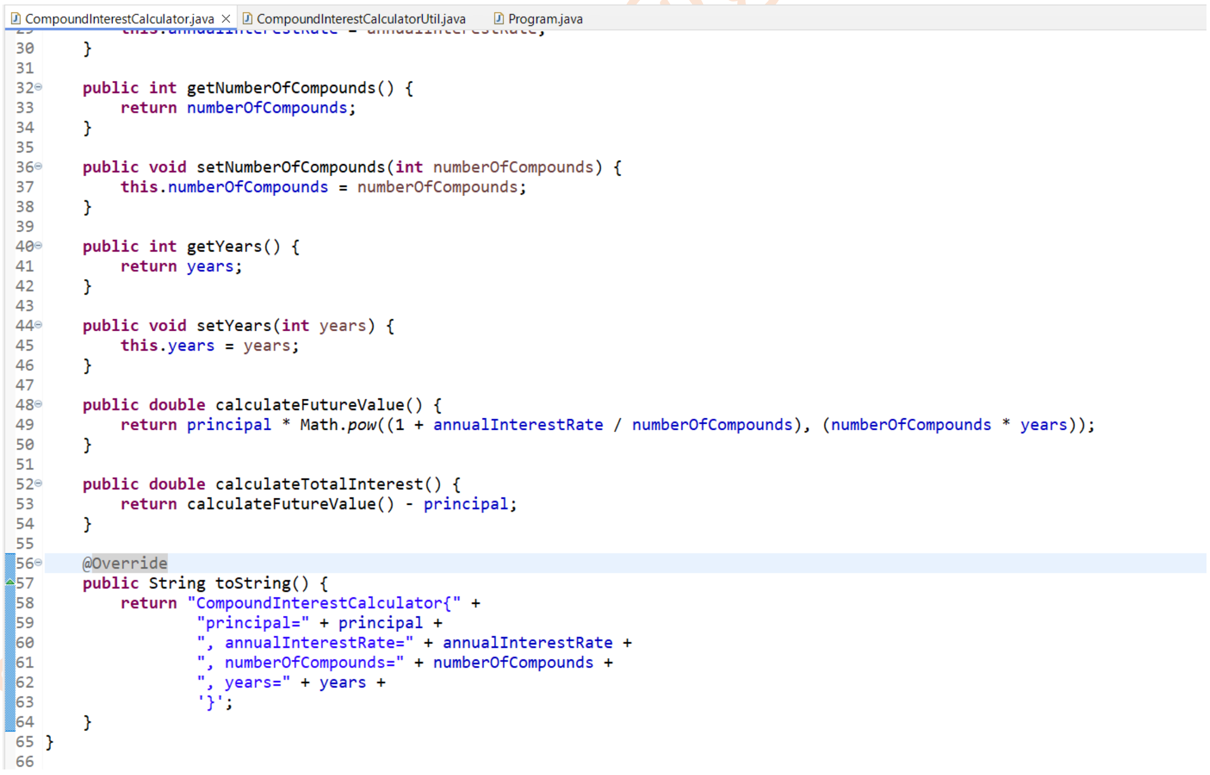
CompoundIntrestCalculator.java

```
package com.example.compoundinterest;

public class CompoundInterestCalculator {
    private double principal;
    private double annualInterestRate;
    private int numberOfCompounds;
    private 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;
    }

    public double calculateFutureValue() {
        return principal * Math.pow((1 + annualInterestRate / numberOfCompounds),
(numberOfCompounds * years));
    }

    public double calculateTotalInterest() {
        return calculateFutureValue() - principal;
    }

    @Override
    public String toString() {
        return "CompoundInterestCalculator{" +
                "principal=" + principal +
                ", annualInterestRate=" + annualInterestRate +
                ", numberOfCompounds=" + numberOfCompounds +
                ", years=" + years +
                '}';
    }
}
```

ASSIGNMENT NO.4


CompoundInterestCalculator.java
CompoundInterestCalculatorUtil.java
Program.java

```java
package com.example.compoundinterest;

public class CompoundInterestCalculator {
    private double principal;
    private double annualInterestRate;
    private int numberOfCompounds;
    private 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;
```


CompoundInterestCalculator.java
CompoundInterestCalculatorUtil.java
Program.java

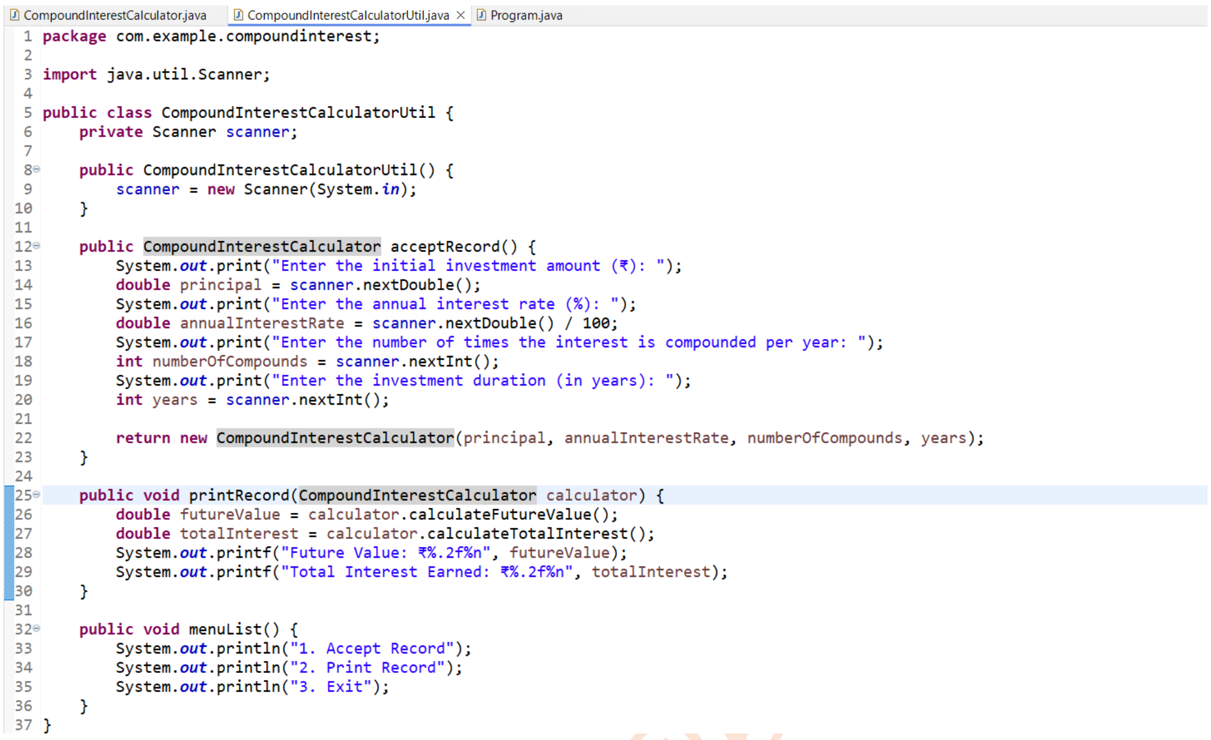
```java
    }

    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;
    }

    public double calculateFutureValue() {
        return principal * Math.pow((1 + annualInterestRate / numberOfCompounds), (numberOfCompounds * years));
    }

    public double calculateTotalInterest() {
        return calculateFutureValue() - principal;
    }

    @Override
    public String toString() {
        return "CompoundInterestCalculator{" +
                "principal=" + principal +
                ", annualInterestRate=" + annualInterestRate +
                ", numberOfCompounds=" + numberOfCompounds +
                ", years=" + years +
                '}';
    }
}
```

CompoundIntrestCalculatorUtil.java

package com.example.compoundinterest;

import java.util.Scanner;

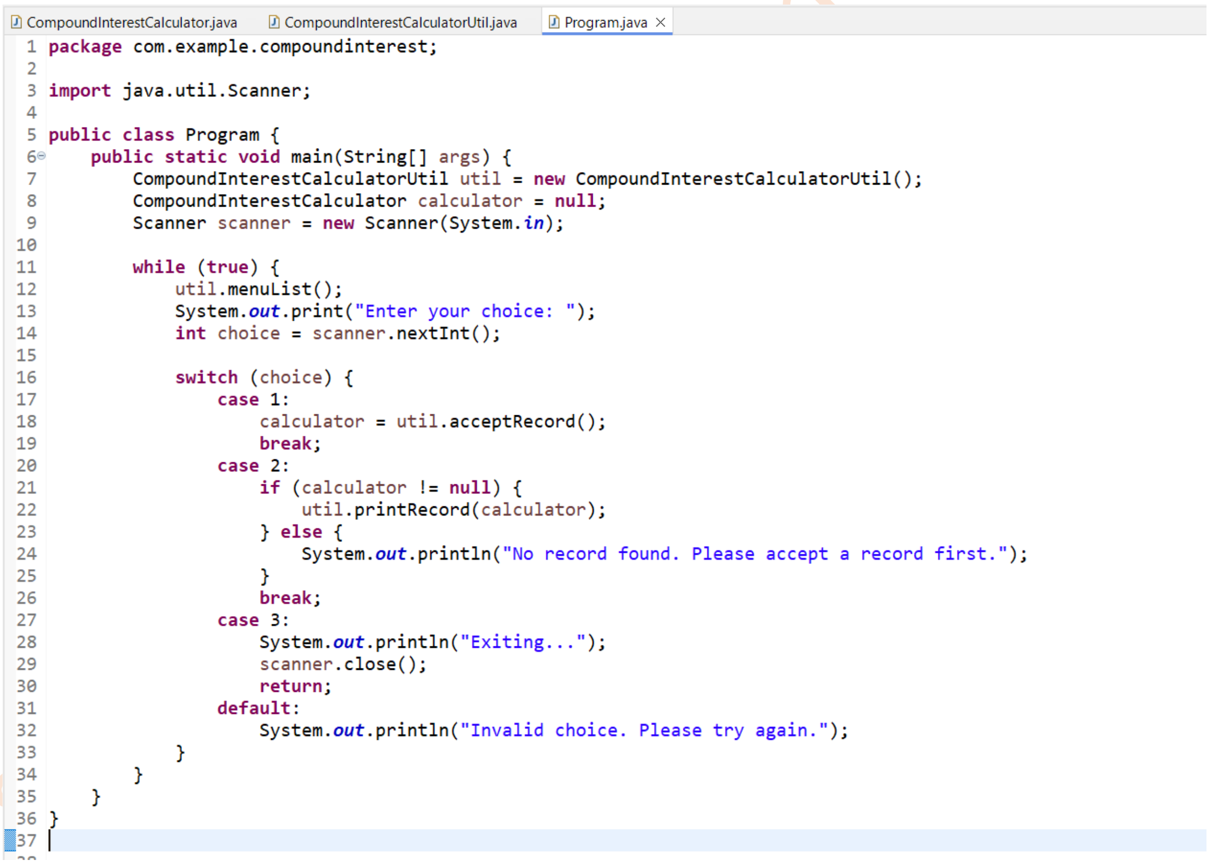
```
public class CompoundInterestCalculatorUtil {
    private Scanner scanner;

    public CompoundInterestCalculatorUtil() {
        scanner = new Scanner(System.in);
    }

    public CompoundInterestCalculator acceptRecord() {
        System.out.print("Enter the initial investment amount (₹): ");
        double principal = scanner.nextDouble();
        System.out.print("Enter the annual interest rate (%): ");
        double annualInterestRate = scanner.nextDouble() / 100;
        System.out.print("Enter the number of times the interest is compounded per
year: ");
        int numberOfCompounds = scanner.nextInt();
        System.out.print("Enter the investment duration (in years): ");
        int years = scanner.nextInt();

        return new CompoundInterestCalculator(principal, annualInterestRate,
numberOfCompounds, years);
    }

    public void printRecord(CompoundInterestCalculator calculator) {
        double futureValue = calculator.calculateFutureValue();
        double totalInterest = calculator.calculateTotalInterest();
        System.out.printf("Future Value: ₹%.2f%n", futureValue);
        System.out.printf("Total Interest Earned: ₹%.2f%n", totalInterest);
    }

    public void menuList() {
        System.out.println("1. Accept Record");
        System.out.println("2. Print Record");
        System.out.println("3. Exit");
    }
}
```


CompoundInterestCalculator.java
CompoundInterestCalculatorUtil.java
Program.java

```java
package com.example.compoundinterest;

import java.util.Scanner;

public class CompoundInterestCalculatorUtil {
    private Scanner scanner;

    public CompoundInterestCalculatorUtil() {
        scanner = new Scanner(System.in);
    }

    public CompoundInterestCalculator acceptRecord() {
        System.out.print("Enter the initial investment amount (₹): ");
        double principal = scanner.nextDouble();
        System.out.print("Enter the annual interest rate (%): ");
        double annualInterestRate = scanner.nextDouble() / 100;
        System.out.print("Enter the number of times the interest is compounded per year: ");
        int numberOfCompounds = scanner.nextInt();
        System.out.print("Enter the investment duration (in years): ");
        int years = scanner.nextInt();

        return new CompoundInterestCalculator(principal, annualInterestRate, numberOfCompounds, years);
    }

    public void printRecord(CompoundInterestCalculator calculator) {
        double futureValue = calculator.calculateFutureValue();
        double totalInterest = calculator.calculateTotalInterest();
        System.out.printf("Future Value: ₹%.2f%n", futureValue);
        System.out.printf("Total Interest Earned: ₹%.2f%n", totalInterest);
    }

    public void menuList() {
        System.out.println("1. Accept Record");
        System.out.println("2. Print Record");
        System.out.println("3. Exit");
    }
}
```

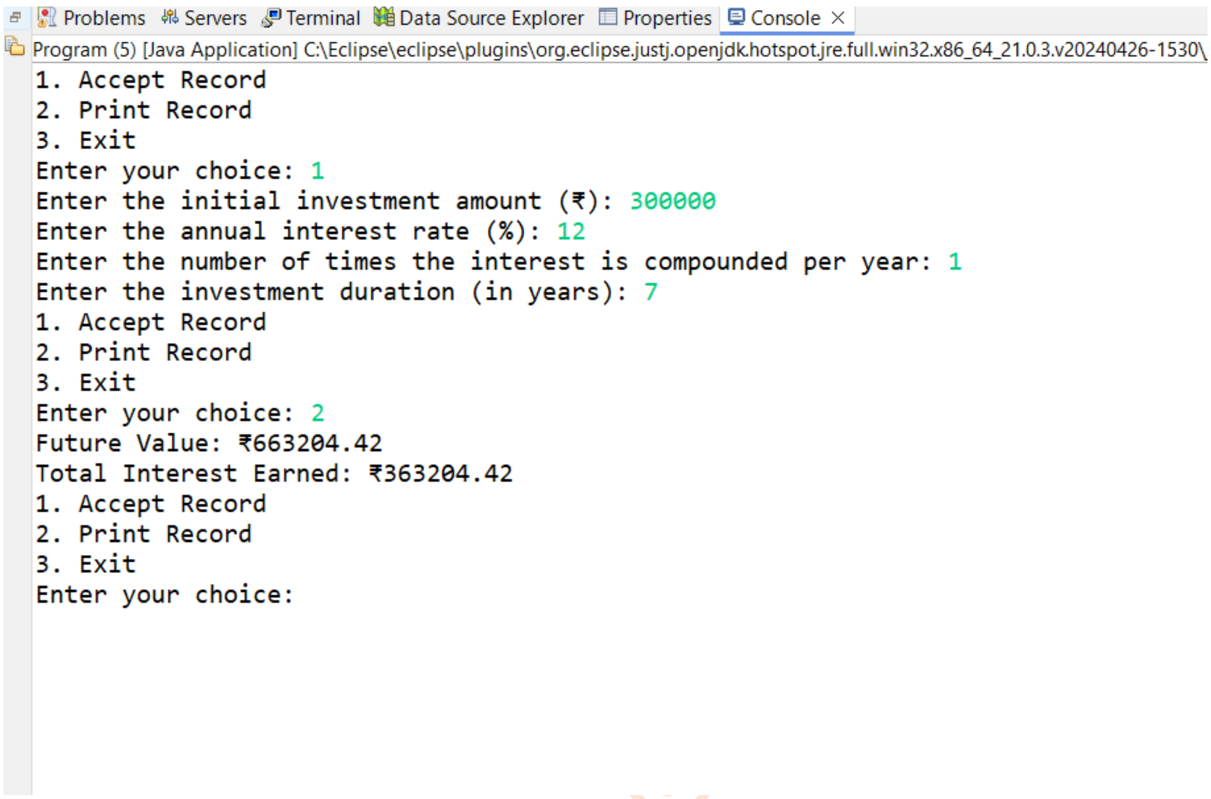
Program.java

```java
package com.example.compoundinterest;


import java.util.Scanner;

public class Program {
    public static void main(String[] args) {
        CompoundInterestCalculatorUtil util = new CompoundInterestCalculatorUtil();
        CompoundInterestCalculator calculator = null;
        Scanner scanner = new Scanner(System.in);

        while (true) {
            util.menuList();
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    calculator = util.acceptRecord();
                    break;
                case 2:
                    if (calculator != null) {
                        util.printRecord(calculator);
                } else {
                    System.out.println("No record found. Please accept a record first.");
                }
                break;
            case 3:
                System.out.println("Exiting...");
                scanner.close();
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
  }
}
```



CompoundInterestCalculator.java   CompoundInterestCalculatorUtil.java   Program.java

```
package com.example.compoundinterest;

import java.util.Scanner;

public class Program {
    public static void main(String[] args) {
        CompoundInterestCalculatorUtil util = new CompoundInterestCalculatorUtil();
        CompoundInterestCalculator calculator = null;
        Scanner scanner = new Scanner(System.in);

        while (true) {
            util.menuList();
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    calculator = util.acceptRecord();
                    break;
                case 2:
                    if (calculator != null) {
                        util.printRecord(calculator);
                    } else {
                        System.out.println("No record found. Please accept a record first.");
                    }
                    break;
                case 3:
                    System.out.println("Exiting...");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
    }
}
```



```
Program (5) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530\
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 1
Enter the initial investment amount (₹): 300000
Enter the annual interest rate (%): 12
Enter the number of times the interest is compounded per year: 1
Enter the investment duration (in years): 7
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 2
Future Value: ₹663204.42
Total Interest Earned: ₹363204.42
1. Accept Record
2. Print Record
3. Exit
Enter your choice:
```

3. BMI (Body Mass Index) Tracker

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

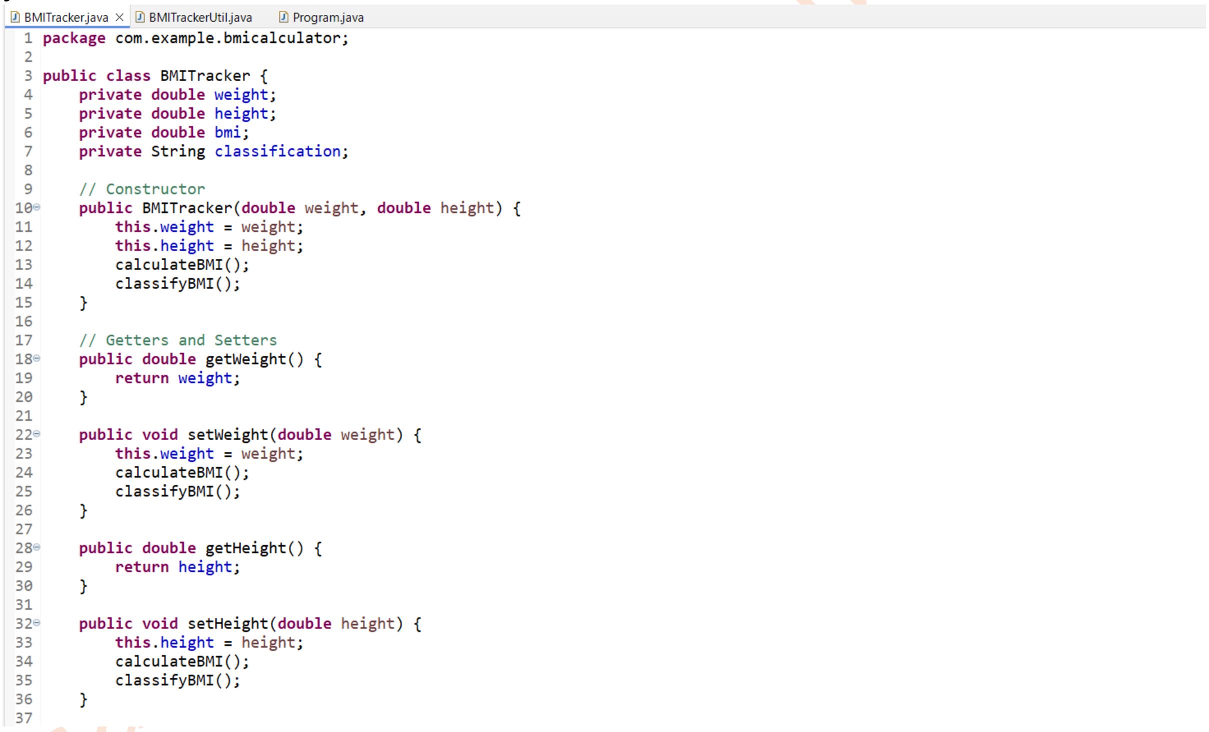
1. Accept weight (in kilograms) and height (in meters) from the user.
2. Calculate the BMI using the formula:
   - **BMI Calculation:** `BMI = weight / (height * height)`
3. Classify the BMI into one of the following categories:
   - Underweight: BMI < 18.5
   - Normal weight: 18.5 ≤ BMI < 24.9
   - 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:

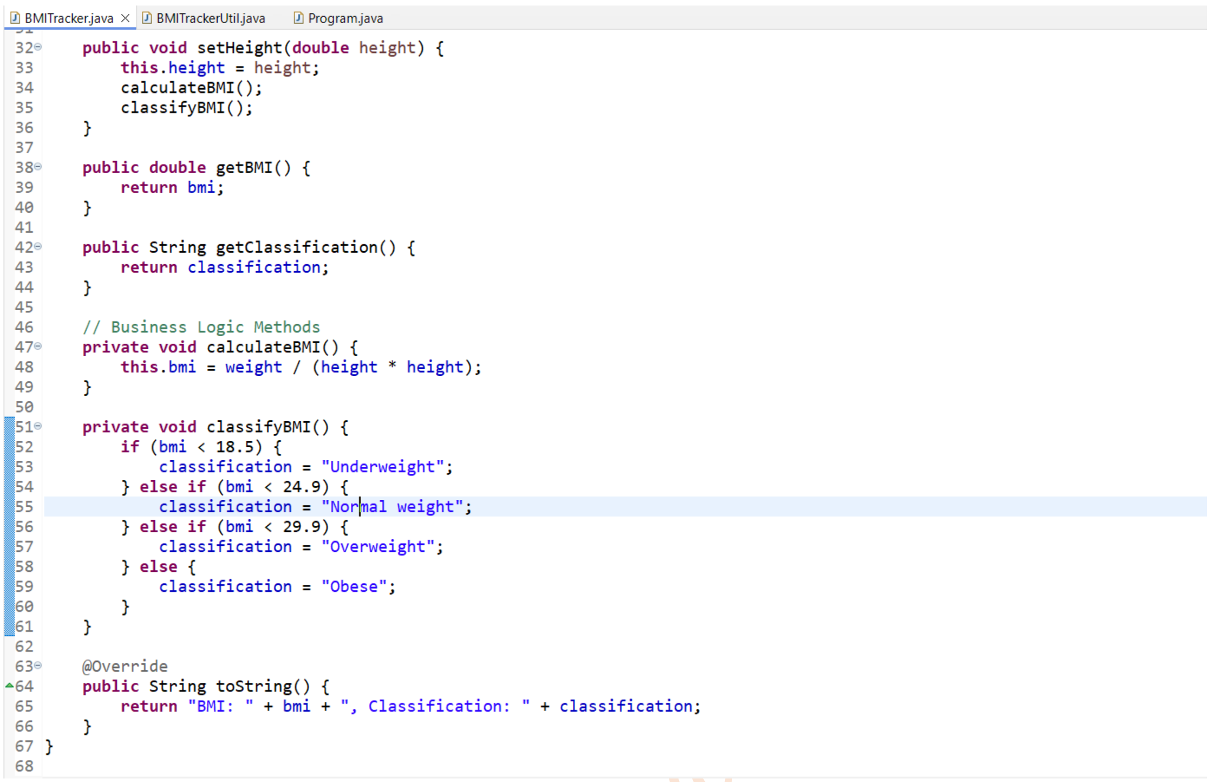
BMITracker.java

```
package com.example.bmicalculator;

public class BMITracker {
    private double weight;
private double height;
private double bmi;
private String classification;

// Constructor
public BMITracker(double weight, double height) {
   this.weight = weight;
   this.height = height;
   calculateBMI();
   classifyBMI();
}

// Getters and Setters
public double getWeight() {
   return weight;
}

public void setWeight(double weight) {
   this.weight = weight;
   calculateBMI();
   classifyBMI();
}

public double getHeight() {
   return height;
}

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

public double getBMI() {
   return bmi;
}

public String getClassification() {
   return classification;
}

// Business Logic Methods
private void calculateBMI() {
   this.bmi = weight / (height * height);
}

private void classifyBMI() {
        if (bmi < 18.5) {
            classification = "Underweight";
        } else if (bmi < 24.9) {
            classification = "Normal weight";
        } else if (bmi < 29.9) {
            classification = "Overweight";
        } else {
            classification = "Obese";
        }
    }

    @Override
    public String toString() {
        return "BMI: " + bmi + ", Classification: " + classification;
    }


}
```



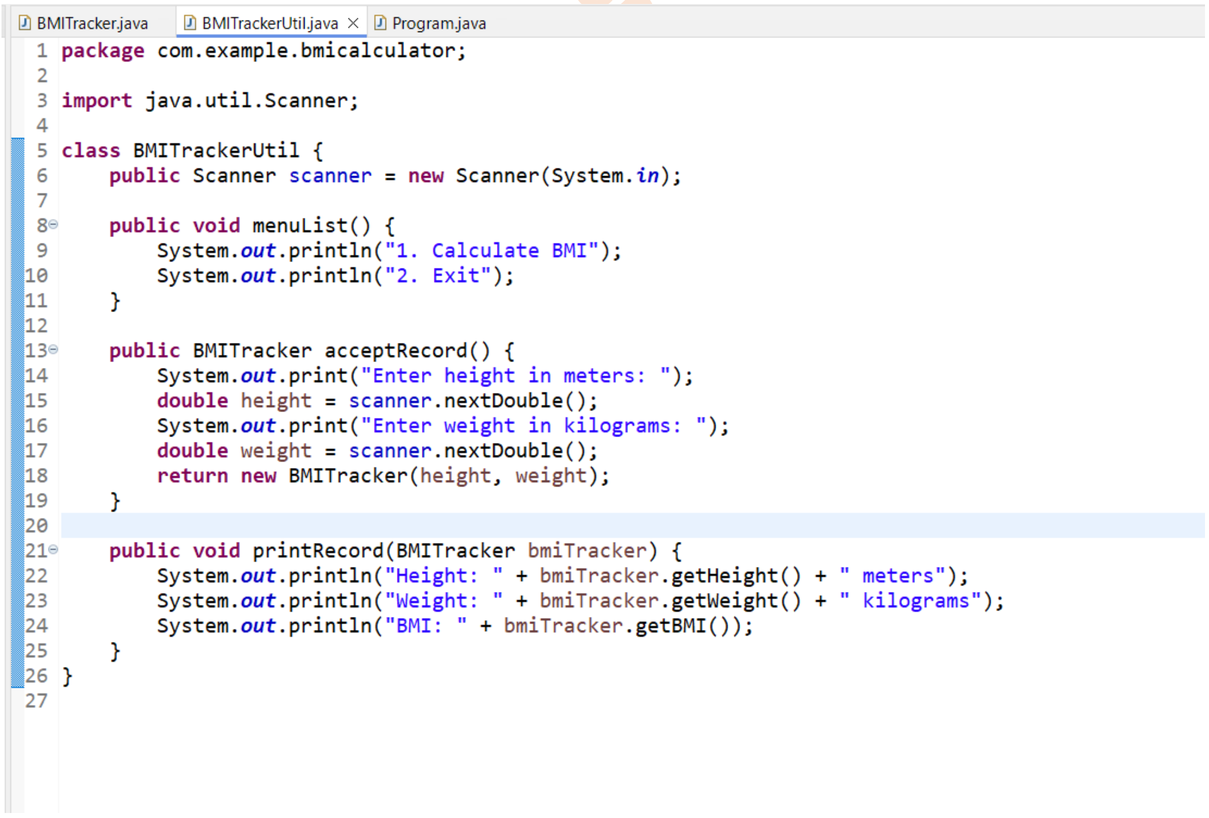
BMITracker.java × BMITrackerUtil.java Program.java

```
package com.example.bmicalculator;

public class BMITracker {
    private double weight;
    private double height;
    private double bmi;
    private String classification;

    // Constructor
    public BMITracker(double weight, double height) {
        this.weight = weight;
        this.height = height;
        calculateBMI();
        classifyBMI();
    }

    // Getters and Setters
    public double getWeight() {
        return weight;
    }

    public void setWeight(double weight) {
        this.weight = weight;
        calculateBMI();
        classifyBMI();
    }

    public double getHeight() {
        return height;
    }

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

```



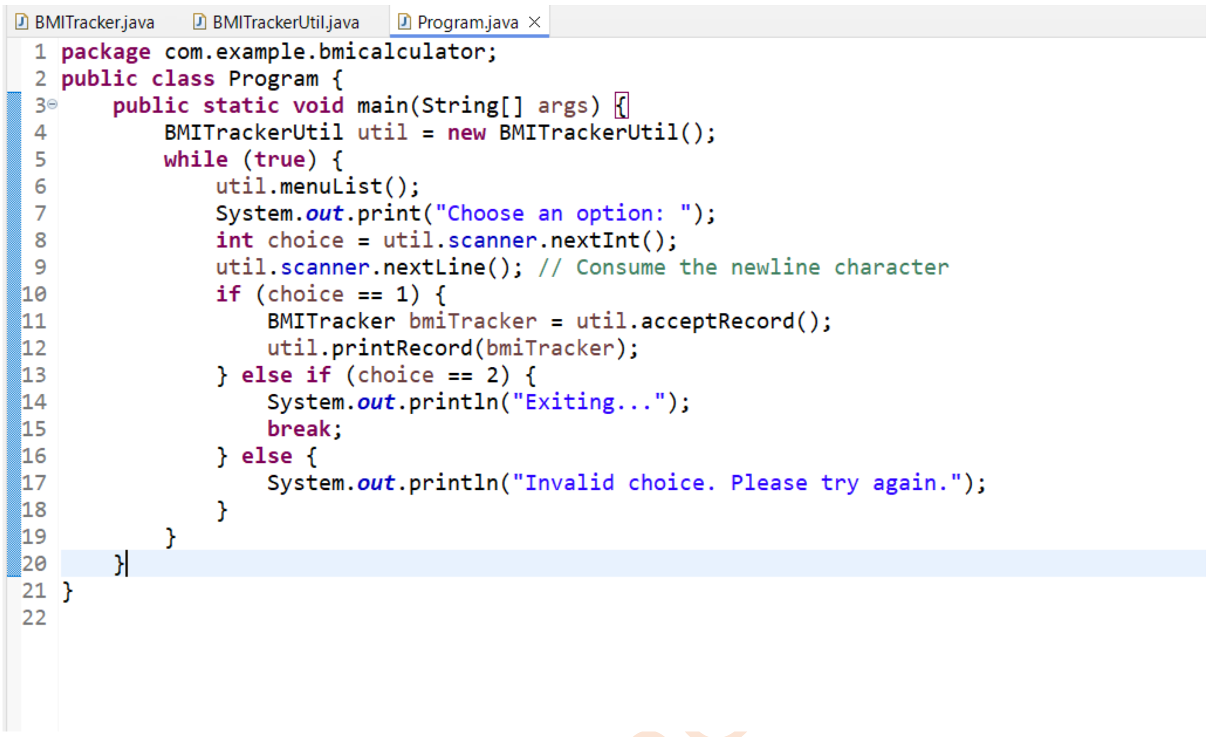
```java
    public void setHeight(double height) {
        this.height = height;
        calculateBMI();
        classifyBMI();
    }

    public double getBMI() {
        return bmi;
    }

    public String getClassification() {
        return classification;
    }

    // Business Logic Methods
    private void calculateBMI() {
        this.bmi = weight / (height * height);
    }

    private void classifyBMI() {
        if (bmi < 18.5) {
            classification = "Underweight";
        } else if (bmi < 24.9) {
            classification = "Normal weight";
        } else if (bmi < 29.9) {
            classification = "Overweight";
        } else {
            classification = "Obese";
        }
    }

    @Override
    public String toString() {
        return "BMI: " + bmi + ", Classification: " + classification;
    }
}
```

BMITrackerUtil.java

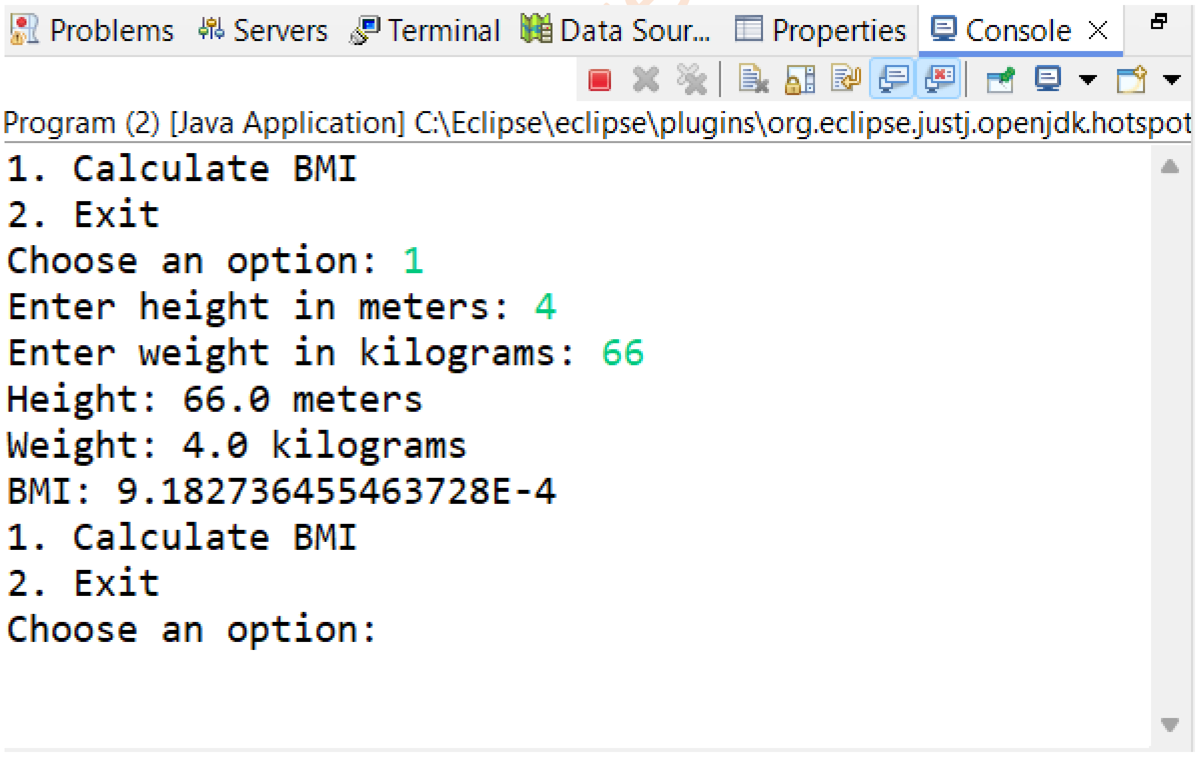


```java
package com.example.bmicalculator;

import java.util.Scanner;

class BMITrackerUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Calculate BMI");
        System.out.println("2. Exit");
    }

    public BMITracker acceptRecord() {
        System.out.print("Enter height in meters: ");
        double height = scanner.nextDouble();
        System.out.print("Enter weight in kilograms: ");
        double weight = scanner.nextDouble();
        return new BMITracker(height, weight);
    }

    public void printRecord(BMITracker bmiTracker) {
        System.out.println("Height: " + bmiTracker.getHeight() + " meters");
        System.out.println("Weight: " + bmiTracker.getWeight() + " kilograms");
        System.out.println("BMI: " + bmiTracker.getBMI());
    }
}
```

Program.java



```java
package com.example.bmicalculator;
public class Program {
    public static void main(String[] args) {
        BMITrackerUtil util = new BMITrackerUtil();
        while (true) {
            util.menuList();
            System.out.print("Choose an option: ");
            int choice = util.scanner.nextInt();
            util.scanner.nextLine(); // Consume the newline character
            if (choice == 1) {
                BMITracker bmiTracker = util.acceptRecord();
                util.printRecord(bmiTracker);
            } else if (choice == 2) {
                System.out.println("Exiting...");
                break;
            } else {
                System.out.println("Invalid choice. Please try again.");
            }
        }
    }
}
```



Program (2) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot

```
1. Calculate BMI
2. Exit
Choose an option: 1
Enter height in meters: 4
Enter weight in kilograms: 66
Height: 66.0 meters
Weight: 4.0 kilograms
BMI: 9.182736455463728E-4
1. Calculate BMI
2. Exit
Choose an option:
```

4. Discount Calculation for Retail Sales

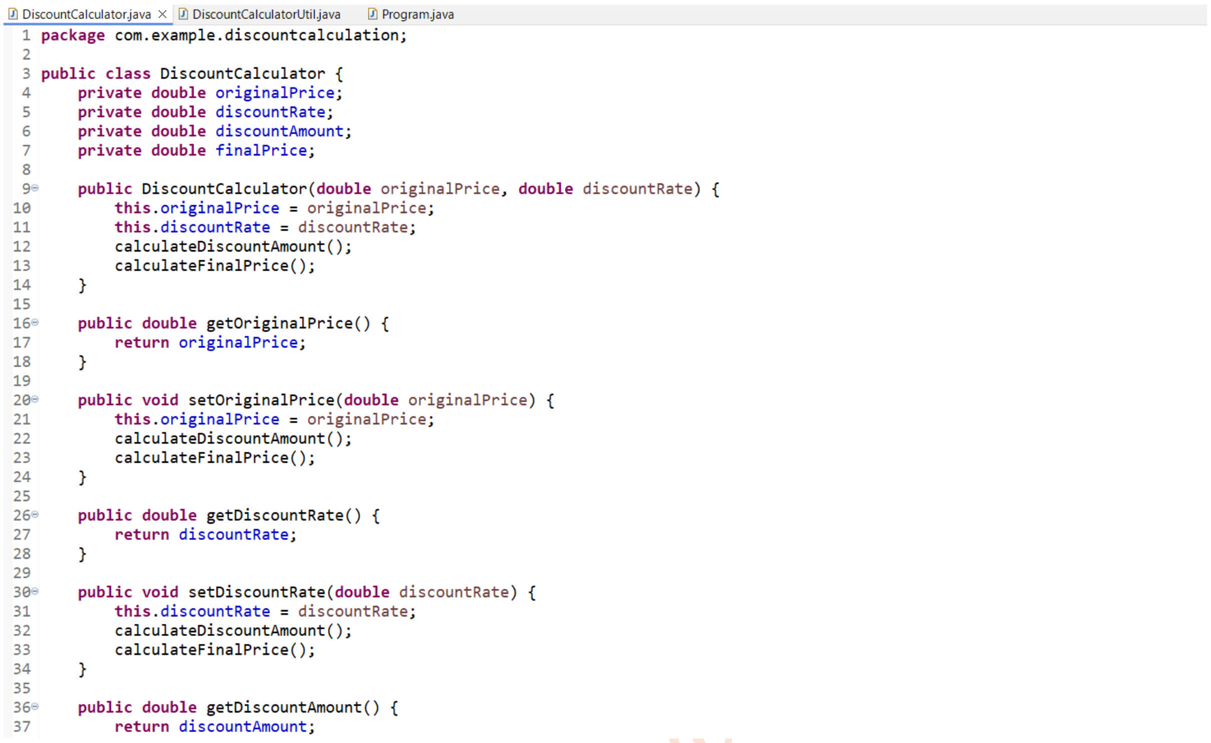
Design a system to calculate the final price of an item after applying a discount. The system should:

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:
DiscountCalculator.java

```
package com.example.discountcalculation;

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

  public DiscountCalculator(double originalPrice, double discountRate) {
    this.originalPrice = originalPrice;
    this.discountRate = discountRate;
    calculateDiscountAmount();
    calculateFinalPrice();
  }

  public double getOriginalPrice() {
    return originalPrice;
  }

  public void setOriginalPrice(double originalPrice) {
    this.originalPrice = originalPrice;
    calculateDiscountAmount();
    calculateFinalPrice();
  }

  public double getDiscountRate() {
    return discountRate;
}

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

public double getDiscountAmount() {
    return discountAmount;
}

public double getFinalPrice() {
    return finalPrice;
}

private void calculateDiscountAmount() {
    this.discountAmount = originalPrice * (discountRate / 100);
}

private void calculateFinalPrice() {
    this.finalPrice = originalPrice - discountAmount;
}

@Override
public String toString() {
    return "Original Price: ₹" + originalPrice + "\n" +
           "Discount Rate: " + discountRate + "%\n" +
           "Discount Amount: ₹" + discountAmount + "\n" +
           "Final Price: ₹" + finalPrice;
}
```

}



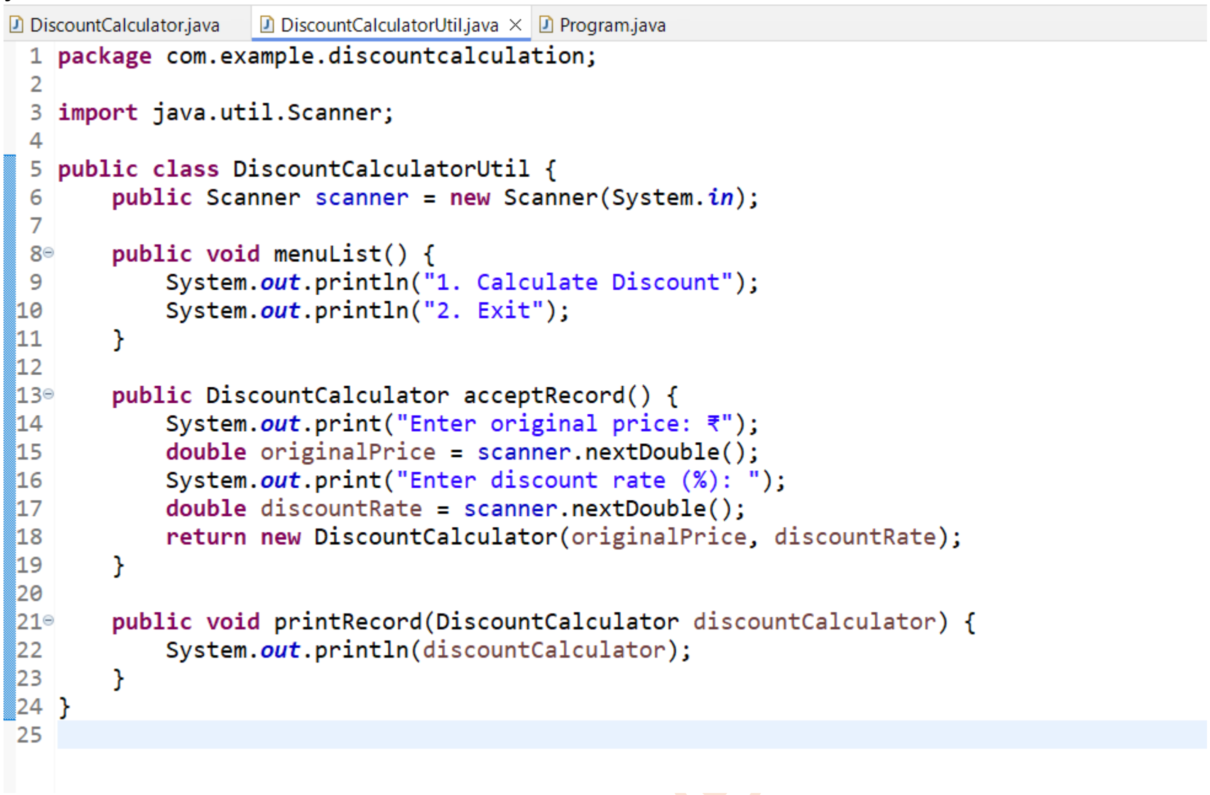
DiscountCalculator.java × DiscountCalculatorUtil.java Program.java

```java
package com.example.discountcalculation;

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

    public DiscountCalculator(double originalPrice, double discountRate) {
        this.originalPrice = originalPrice;
        this.discountRate = discountRate;
        calculateDiscountAmount();
        calculateFinalPrice();
    }

    public double getOriginalPrice() {
        return originalPrice;
    }

    public void setOriginalPrice(double originalPrice) {
        this.originalPrice = originalPrice;
        calculateDiscountAmount();
        calculateFinalPrice();
    }

    public double getDiscountRate() {
        return discountRate;
    }

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

    public double getDiscountAmount() {
        return discountAmount;
```

DiscountCalculatorUtil.java

```java
package com.example.discountcalculation;

import java.util.Scanner;

public class DiscountCalculatorUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Calculate Discount");
        System.out.println("2. Exit");
    }

    public DiscountCalculator acceptRecord() {
        System.out.print("Enter original price: ₹");
        double originalPrice = scanner.nextDouble();
        System.out.print("Enter discount rate (%): ");
        double discountRate = scanner.nextDouble();
        return new DiscountCalculator(originalPrice, discountRate);
    }

    public void printRecord(DiscountCalculator discountCalculator) {
        System.out.println(discountCalculator);
    }
```

}



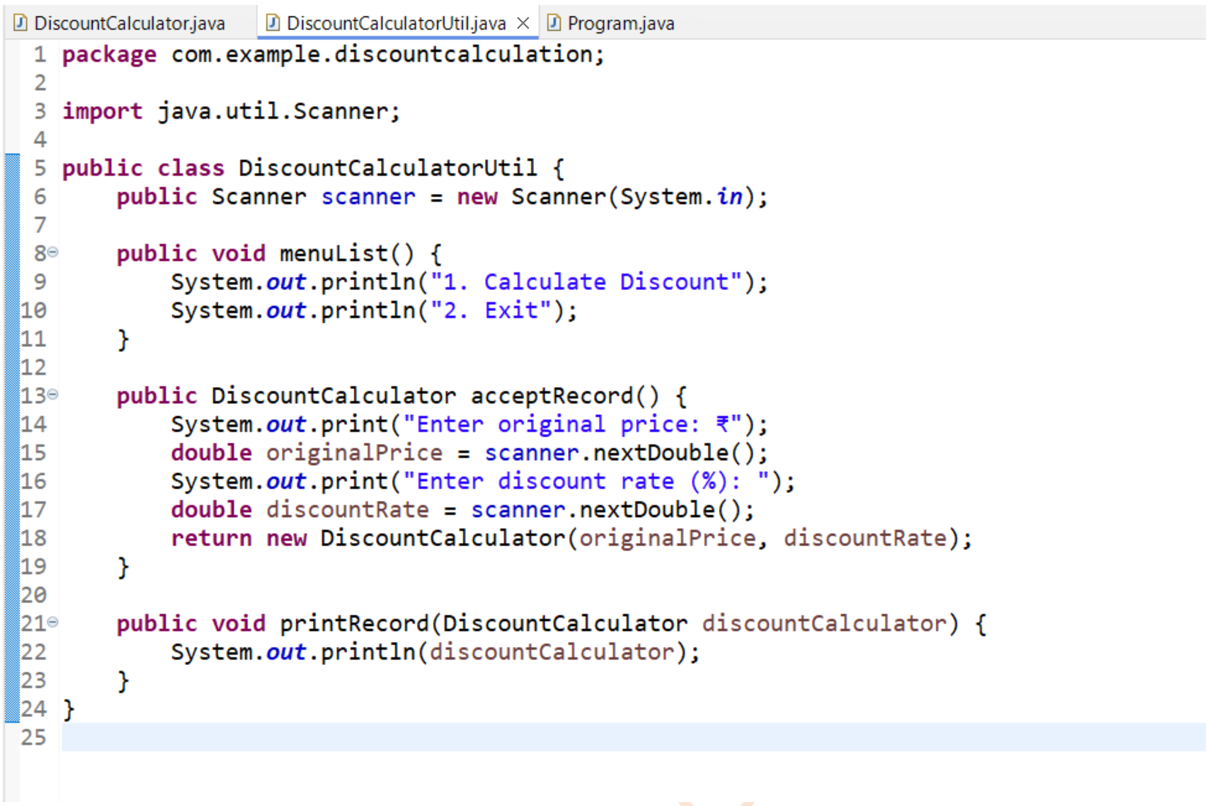
DiscountCalculator.java | DiscountCalculatorUtil.java | Program.java

```
package com.example.discountcalculation;

import java.util.Scanner;

public class DiscountCalculatorUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Calculate Discount");
        System.out.println("2. Exit");
    }

    public DiscountCalculator acceptRecord() {
        System.out.print("Enter original price: ₹");
        double originalPrice = scanner.nextDouble();
        System.out.print("Enter discount rate (%): ");
        double discountRate = scanner.nextDouble();
        return new DiscountCalculator(originalPrice, discountRate);
    }

    public void printRecord(DiscountCalculator discountCalculator) {
        System.out.println(discountCalculator);
    }
}
```

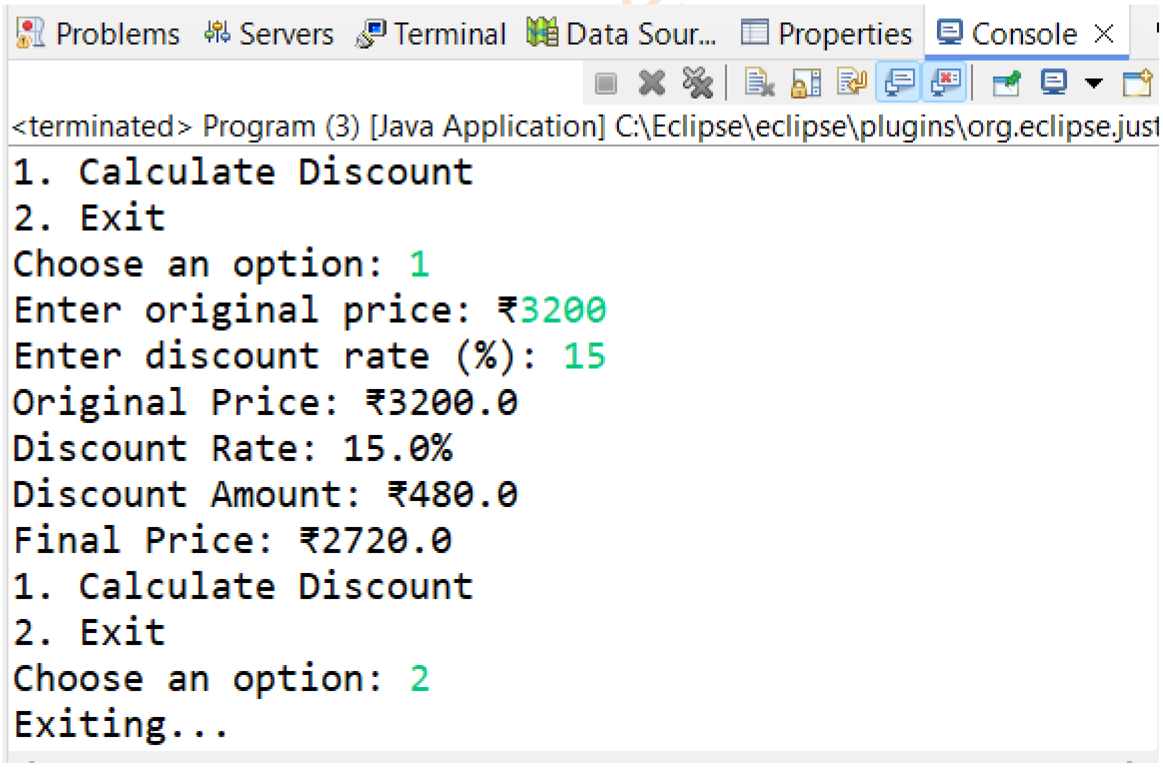
Program.java

```
package com.example.discountcalculation;

import java.util.Scanner;

public class DiscountCalculatorUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Calculate Discount");
        System.out.println("2. Exit");
    }

    public DiscountCalculator acceptRecord() {
        System.out.print("Enter original price: ₹");
        double originalPrice = scanner.nextDouble();
        System.out.print("Enter discount rate (%): ");
        double discountRate = scanner.nextDouble();
        return new DiscountCalculator(originalPrice, discountRate);
    }

    public void printRecord(DiscountCalculator discountCalculator) {
        System.out.println(discountCalculator);
    }
}
```

ASSIGNMENT NO.4



```java
package com.example.discountcalculation;

import java.util.Scanner;

public class DiscountCalculatorUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Calculate Discount");
        System.out.println("2. Exit");
    }

    public DiscountCalculator acceptRecord() {
        System.out.print("Enter original price: ₹");
        double originalPrice = scanner.nextDouble();
        System.out.print("Enter discount rate (%): ");
        double discountRate = scanner.nextDouble();
        return new DiscountCalculator(originalPrice, discountRate);
    }

    public void printRecord(DiscountCalculator discountCalculator) {
        System.out.println(discountCalculator);
    }
}
```



```
<terminated> Program (3) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.just
1. Calculate Discount
2. Exit
Choose an option: 1
Enter original price: ₹3200
Enter discount rate (%): 15
Original Price: ₹3200.0
Discount Rate: 15.0%
Discount Amount: ₹480.0
Final Price: ₹2720.0
1. Calculate Discount
2. Exit
Choose an option: 2
Exiting...
```

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

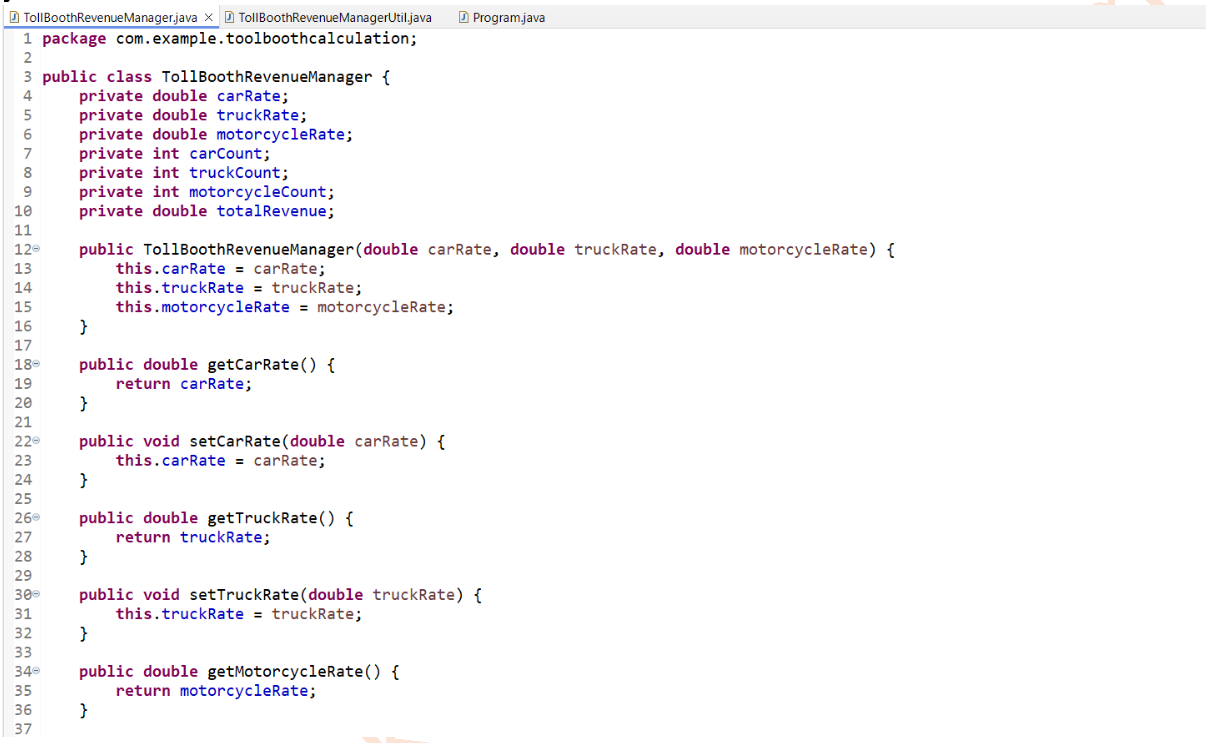
Define the class `TollBoothRevenueManager` with fields, an appropriate constructor, getter and setter methods, a `toString` method, and business logic methods. Define the class `TollBoothRevenueManagerUtil` with methods `acceptRecord`, `printRecord`, and `menuList`. Define the class `Program` with a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

Ans:

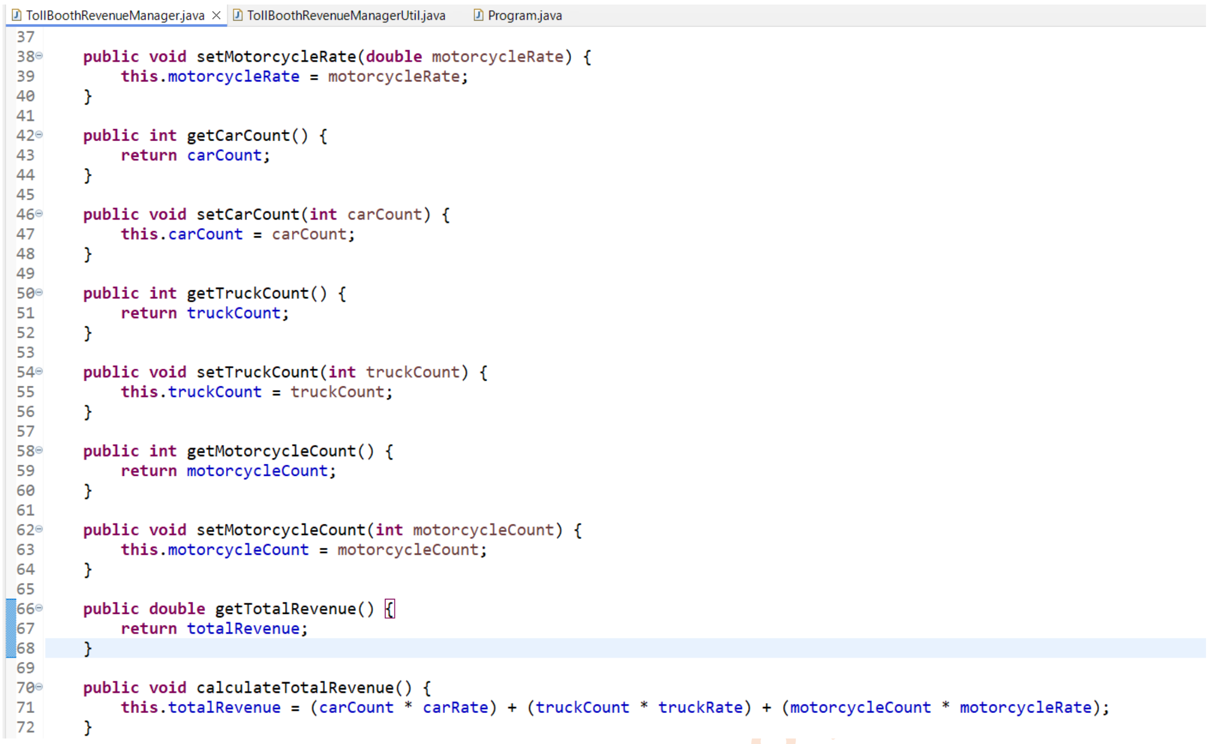
TollBoothRevenueManager.java

```
package com.example.toolboothcalculation;

public class TollBoothRevenueManager {
   private double carRate;
   private double truckRate;
   private double motorcycleRate;
   private int carCount;
   private int truckCount;
   private int motorcycleCount;
   private double totalRevenue;

   public TollBoothRevenueManager(double carRate, double truckRate, double
motorcycleRate) {
      this.carRate = carRate;
      this.truckRate = truckRate;
      this.motorcycleRate = motorcycleRate;
   }

   public double getCarRate() {
      return carRate;
   }

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

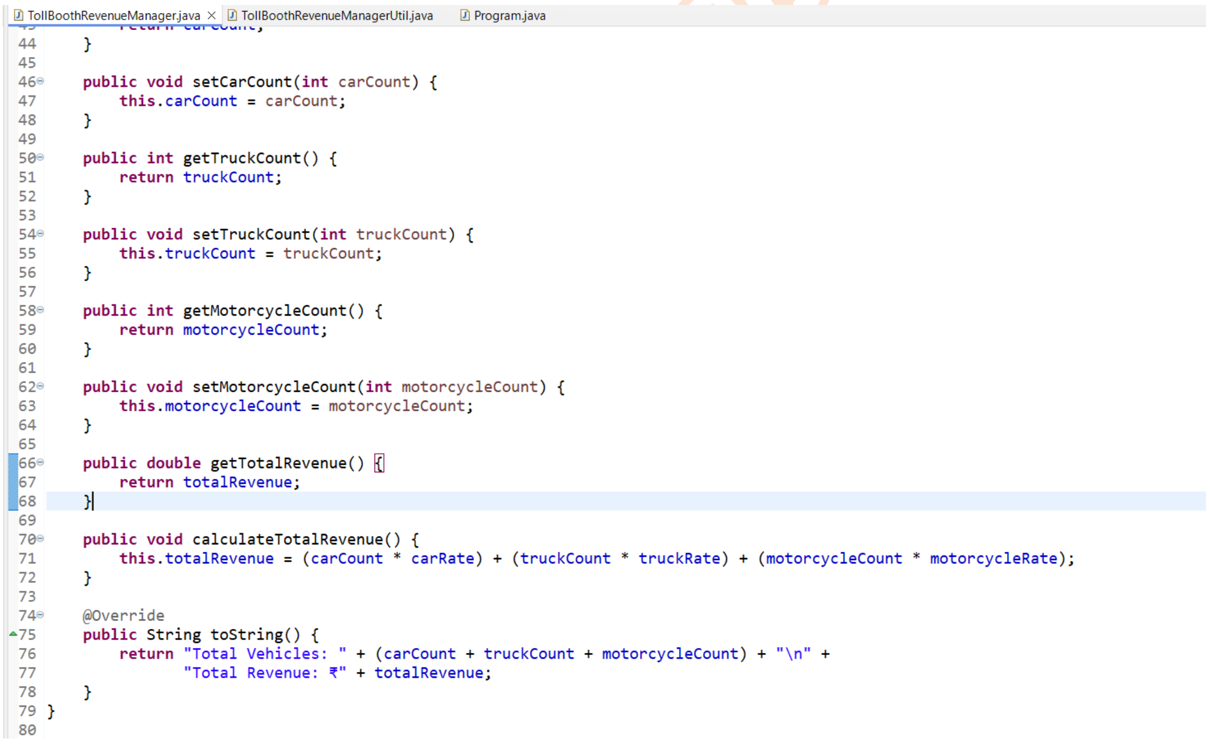
```
}

public double getTruckRate() {
    return truckRate;
}

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

public double getMotorcycleRate() {
    return motorcycleRate;
}

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

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 getTotalRevenue() {
    return totalRevenue;
}

public void calculateTotalRevenue() {
        this.totalRevenue = (carCount * carRate) + (truckCount * truckRate) +
(motorcycleCount * motorcycleRate);
    }

    @Override
    public String toString() {
        return "Total Vehicles: " + (carCount + truckCount + motorcycleCount) + "\n" +
               "Total Revenue: ₹" + totalRevenue;
    }
}
```


TollBoothRevenueManager.java   TollBoothRevenueManagerUtil.java   Program.java

```
package com.example.toolboothcalculation;

public class TollBoothRevenueManager {
    private double carRate;
    private double truckRate;
    private double motorcycleRate;
    private int carCount;
    private int truckCount;
    private int motorcycleCount;
    private double totalRevenue;

    public TollBoothRevenueManager(double carRate, double truckRate, double motorcycleRate) {
        this.carRate = carRate;
        this.truckRate = truckRate;
        this.motorcycleRate = motorcycleRate;
    }

    public double getCarRate() {
        return carRate;
    }

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

    public double getTruckRate() {
        return truckRate;
    }

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

    public double getMotorcycleRate() {
        return motorcycleRate;
    }

```


TollBoothRevenueManager.java
TollBoothRevenueManagerUtil.java
Program.java

```java

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

    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 getTotalRevenue() {
        return totalRevenue;
    }

    public void calculateTotalRevenue() {
        this.totalRevenue = (carCount * carRate) + (truckCount * truckRate) + (motorcycleCount * motorcycleRate);
    }
```


TollBoothRevenueManager.java
TollBoothRevenueManagerUtil.java
Program.java

```java
    }

    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 getTotalRevenue() {
        return totalRevenue;
    }

    public void calculateTotalRevenue() {
        this.totalRevenue = (carCount * carRate) + (truckCount * truckRate) + (motorcycleCount * motorcycleRate);
    }

    @Override
    public String toString() {
        return "Total Vehicles: " + (carCount + truckCount + motorcycleCount) + "\n" +
               "Total Revenue: ₹" + totalRevenue;
    }
}

```

TollBoothRevenueManager.java

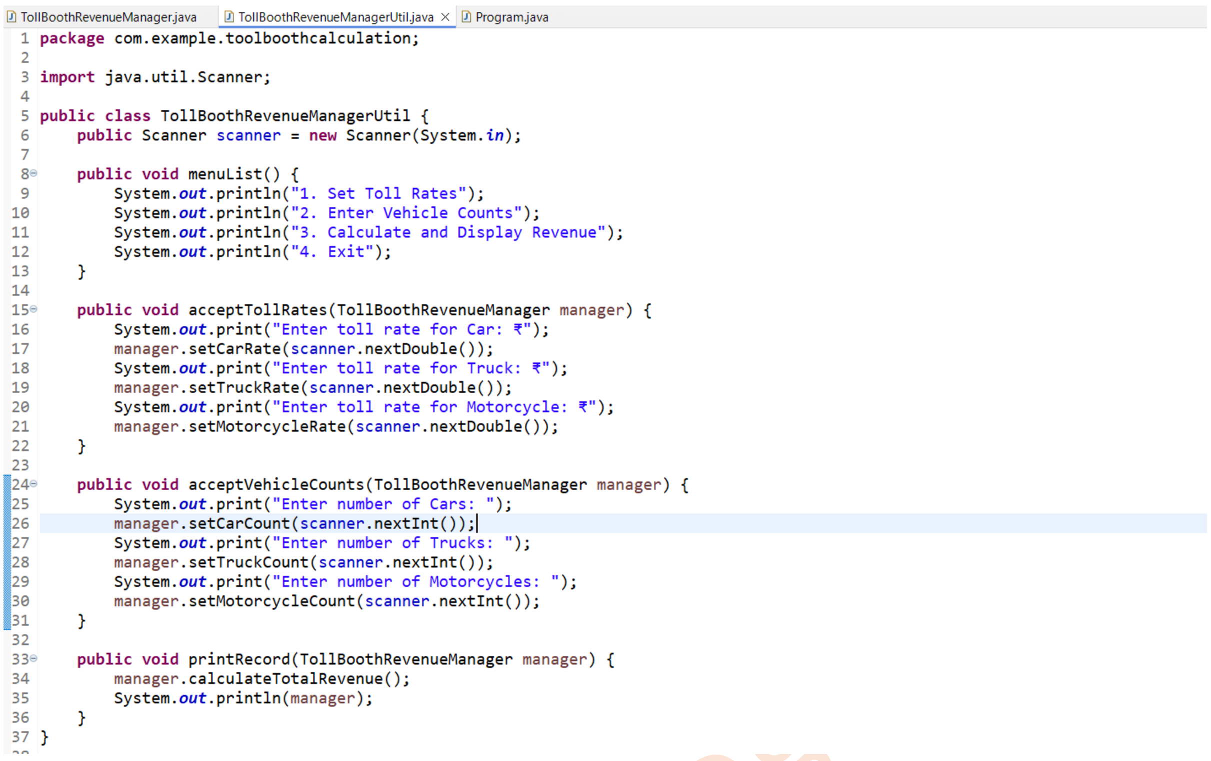
package com.example.toolboothcalculation;

import java.util.Scanner;

public class TollBoothRevenueManagerUtil {

```
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Set Toll Rates");
        System.out.println("2. Enter Vehicle Counts");
        System.out.println("3. Calculate and Display Revenue");
        System.out.println("4. Exit");
    }

    public void acceptTollRates(TollBoothRevenueManager manager) {
        System.out.print("Enter toll rate for Car: ₹");
        manager.setCarRate(scanner.nextDouble());
        System.out.print("Enter toll rate for Truck: ₹");
        manager.setTruckRate(scanner.nextDouble());
        System.out.print("Enter toll rate for Motorcycle: ₹");
        manager.setMotorcycleRate(scanner.nextDouble());
    }

    public void acceptVehicleCounts(TollBoothRevenueManager manager) {
        System.out.print("Enter number of Cars: ");
        manager.setCarCount(scanner.nextInt());
        System.out.print("Enter number of Trucks: ");
        manager.setTruckCount(scanner.nextInt());
        System.out.print("Enter number of Motorcycles: ");
        manager.setMotorcycleCount(scanner.nextInt());
    }

    public void printRecord(TollBoothRevenueManager manager) {
        manager.calculateTotalRevenue();
        System.out.println(manager);
    }
}
```

ASSIGNMENT NO.4

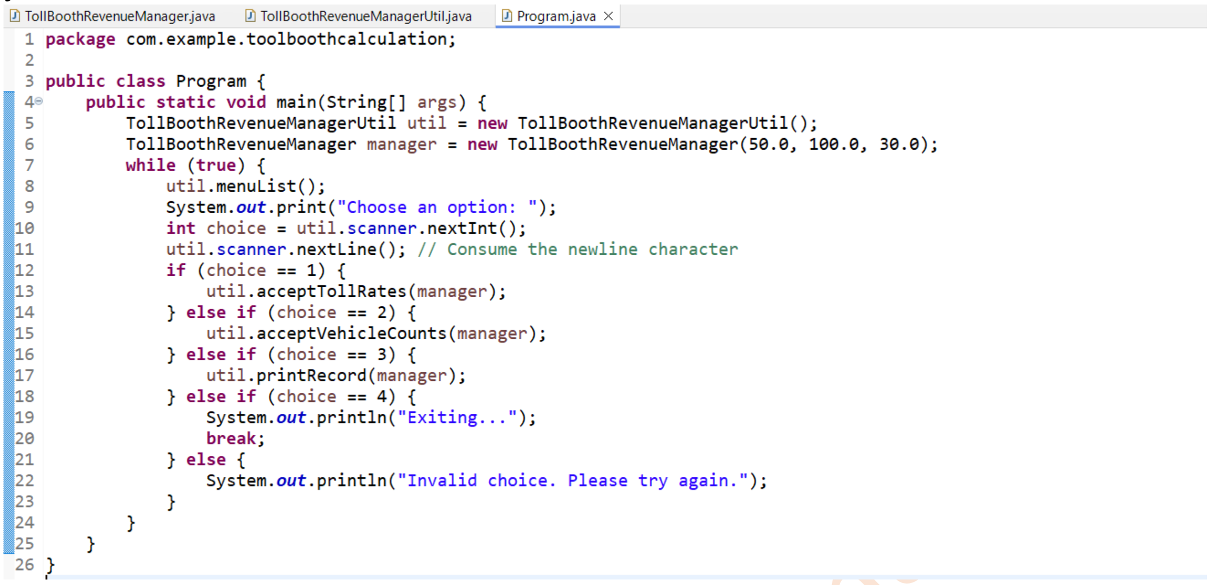

TollBoothRevenueManager.java
TollBoothRevenueManagerUtil.java
Program.java

```java
package com.example.toolboothcalculation;

import java.util.Scanner;

public class TollBoothRevenueManagerUtil {
    public Scanner scanner = new Scanner(System.in);

    public void menuList() {
        System.out.println("1. Set Toll Rates");
        System.out.println("2. Enter Vehicle Counts");
        System.out.println("3. Calculate and Display Revenue");
        System.out.println("4. Exit");
    }

    public void acceptTollRates(TollBoothRevenueManager manager) {
        System.out.print("Enter toll rate for Car: ₹");
        manager.setCarRate(scanner.nextDouble());
        System.out.print("Enter toll rate for Truck: ₹");
        manager.setTruckRate(scanner.nextDouble());
        System.out.print("Enter toll rate for Motorcycle: ₹");
        manager.setMotorcycleRate(scanner.nextDouble());
    }

    public void acceptVehicleCounts(TollBoothRevenueManager manager) {
        System.out.print("Enter number of Cars: ");
        manager.setCarCount(scanner.nextInt());
        System.out.print("Enter number of Trucks: ");
        manager.setTruckCount(scanner.nextInt());
        System.out.print("Enter number of Motorcycles: ");
        manager.setMotorcycleCount(scanner.nextInt());
    }

    public void printRecord(TollBoothRevenueManager manager) {
        manager.calculateTotalRevenue();
        System.out.println(manager);
    }
}
```

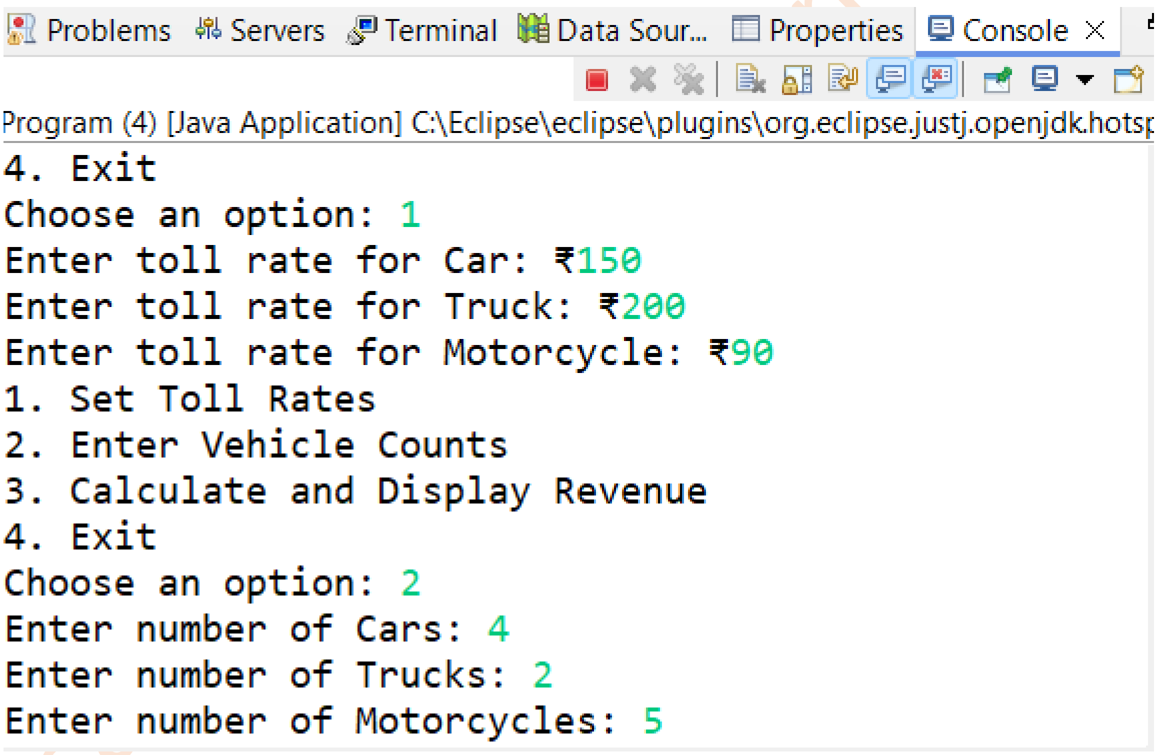
Program.java

```java
package com.example.toolboothcalculation;

public class Program {
   public static void main(String[] args) {
      TollBoothRevenueManagerUtil util = new TollBoothRevenueManagerUtil();
      TollBoothRevenueManager manager = new TollBoothRevenueManager(50.0, 100.0,
30.0);
      while (true) {
         util.menuList();
         System.out.print("Choose an option: ");
         int choice = util.scanner.nextInt();
         util.scanner.nextLine(); // Consume the newline character
         if (choice == 1) {
            util.acceptTollRates(manager);
         } else if (choice == 2) {
            util.acceptVehicleCounts(manager);
         } else if (choice == 3) {
            util.printRecord(manager);
         } else if (choice == 4) {
            System.out.println("Exiting...");
            break;
         } else {
            System.out.println("Invalid choice. Please try again.");
         }
      }
```

```
  }
}
```


TollBoothRevenueManager.java
TollBoothRevenueManagerUtil.java
Program.java

```java
package com.example.toolboothcalculation;

public class Program {
    public static void main(String[] args) {
        TollBoothRevenueManagerUtil util = new TollBoothRevenueManagerUtil();
        TollBoothRevenueManager manager = new TollBoothRevenueManager(50.0, 100.0, 30.0);
        while (true) {
            util.menuList();
            System.out.print("Choose an option: ");
            int choice = util.scanner.nextInt();
            util.scanner.nextLine(); // Consume the newline character
            if (choice == 1) {
                util.acceptTollRates(manager);
            } else if (choice == 2) {
                util.acceptVehicleCounts(manager);
            } else if (choice == 3) {
                util.printRecord(manager);
            } else if (choice == 4) {
                System.out.println("Exiting...");
                break;
            } else {
                System.out.println("Invalid choice. Please try again.");
            }
        }
    }
}
```



```
Program (4) [Java Application] C:\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotsp
4. Exit
Choose an option: 1
Enter toll rate for Car: ₹150
Enter toll rate for Truck: ₹200
Enter toll rate for Motorcycle: ₹90
1. Set Toll Rates
2. Enter Vehicle Counts
3. Calculate and Display Revenue
4. Exit
Choose an option: 2
Enter number of Cars: 4
Enter number of Trucks: 2
Enter number of Motorcycles: 5
```