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.

Program.JAVA:

```
package org.assignment4;


import java.util.Scanner;


public class Program {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        LoanAmortizationCalculatorUtil calculatorUtil = new
LoanAmortizationCalculatorUtil();
```

```
        LoanAmortizationCalculator calculator = null;  // Initialize the
LoanAmortizationCalculator

        int choice;


        do {
            // Display menu
            System.out.println("1. Enter Loan Details");
            System.out.println("2. Display Loan Amortization");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();


            switch (choice) {
                case 1:
                    // Accept loan details from the user
                    calculator = calculatorUtil.acceptRecord();
                    break;


                case 2:
                    // Ensure loan details are entered before printing
                    if (calculator != null) {
                        calculatorUtil.printRecord(calculator);
                    } else {
                        System.out.println("Please enter loan details first (Option 1).");
                    }
```

```
                break;


            case 3:
                System.out.println("Exiting the program...");
                break;


            default:
                System.out.println("Wrong Choice! Please try again.");
            }
        } while (choice != 3);  // Loop until the user chooses to exit


        sc.close();
        System.out.println("Thanks ....");
    }
}
```

LoanAmortizationCalculator.JAVA:

```
package org.assignment4;


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

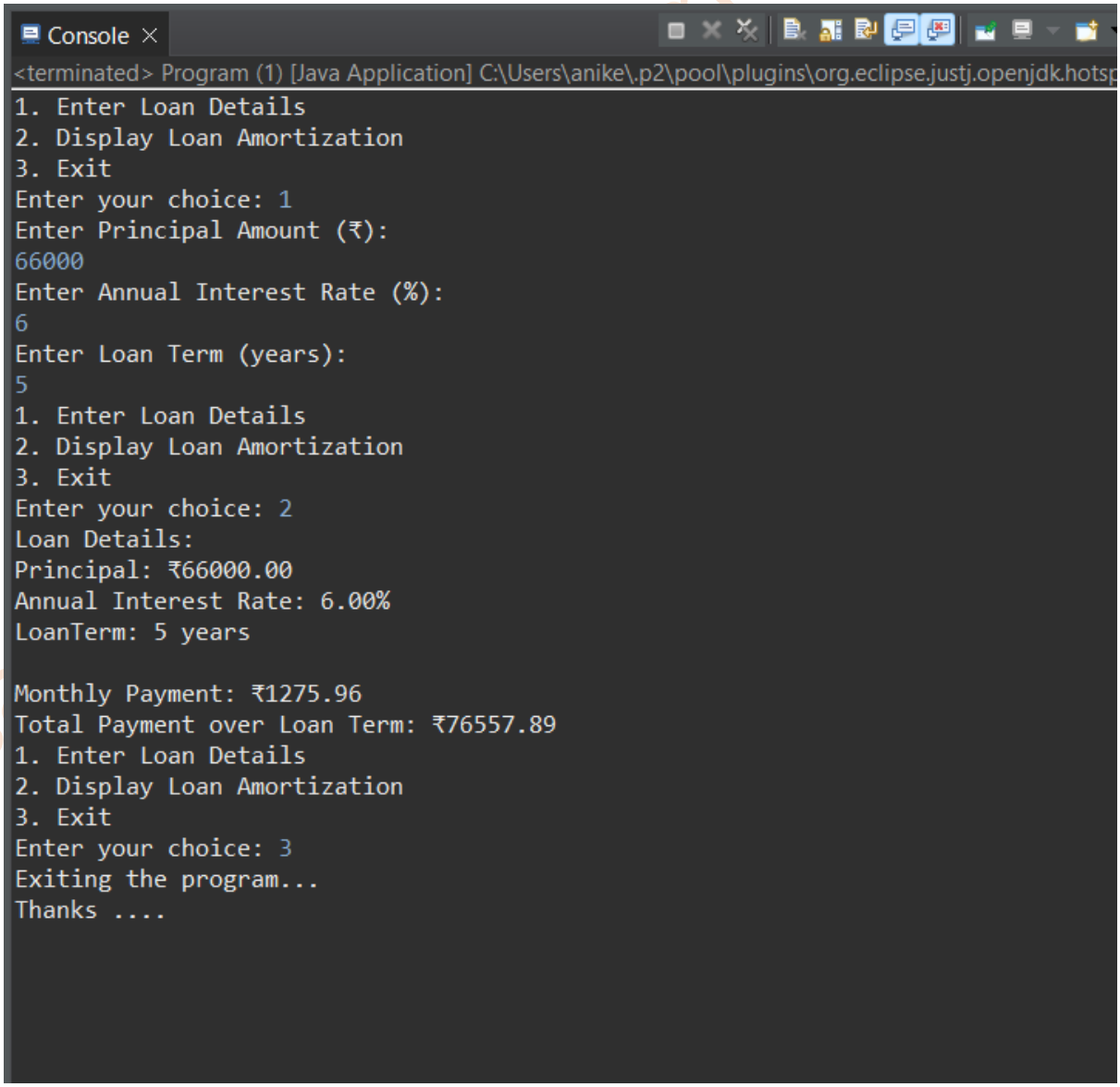
```
    public LoanAmortizationCalculator(double principal, double
annualInterestRate, int loanTerm) {


        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;

        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 String.format("Loan Details:\nPrincipal: ₹%.2f\nAnnual Interest
Rate: %.2f%%\nLoanTerm: %d years\n", principal, annualInterestRate, loanTerm);


    }

}
```

LoanAmortizationCalculatorUtil.JAVA:

```
package org.assignment4;


import java.util.Scanner;


public class LoanAmortizationCalculatorUtil {


	public static LoanAmortizationCalculator acceptRecord() {

		Scanner sc = new Scanner(System.in);

		System.out.println("Enter Principal Amount (₹): ");

		double principal = sc.nextDouble();

		System.out.println("Enter Annual Interest Rate (%): ");

		double annualInterestRate = sc.nextDouble();

		System.out.println("Enter Loan Term (years): ");

		int loanTerm = sc.nextInt();

		return new LoanAmortizationCalculator(principal, annualInterestRate,
loanTerm);


	}


	public static void printRecord(LoanAmortizationCalculator calculator) {

		double monthlyPayment = calculator.calculateMonthlyPayment();

		double totalPayment = calculator.calculateTotalPayment();

		System.out.println(calculator);

		System.out.printf("Monthly Payment: ₹%.2f\n", monthlyPayment);

		System.out.printf("Total Payment over Loan Term: ₹%.2f\n",
totalPayment);
```

```
        }
        public static void menuList() {
                LoanAmortizationCalculator calculator = acceptRecord();
                printRecord(calculator);

        }
}
```



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.

PROGRAM.JAVA:

```
package org.assignment4;

import java.util.Scanner;

public class Program {
	public static void main(String[] args) {
		CompoundInterestCalculatorUtil util = new
CompoundInterestCalculatorUtil();
		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 the program...");
                break;
                default:
                System.out.println("Invalid choice. Please try again.");
                }
                } while (choice != 3);
                sc.close();
                }
}
```

CompoundInterestCalculator.java:

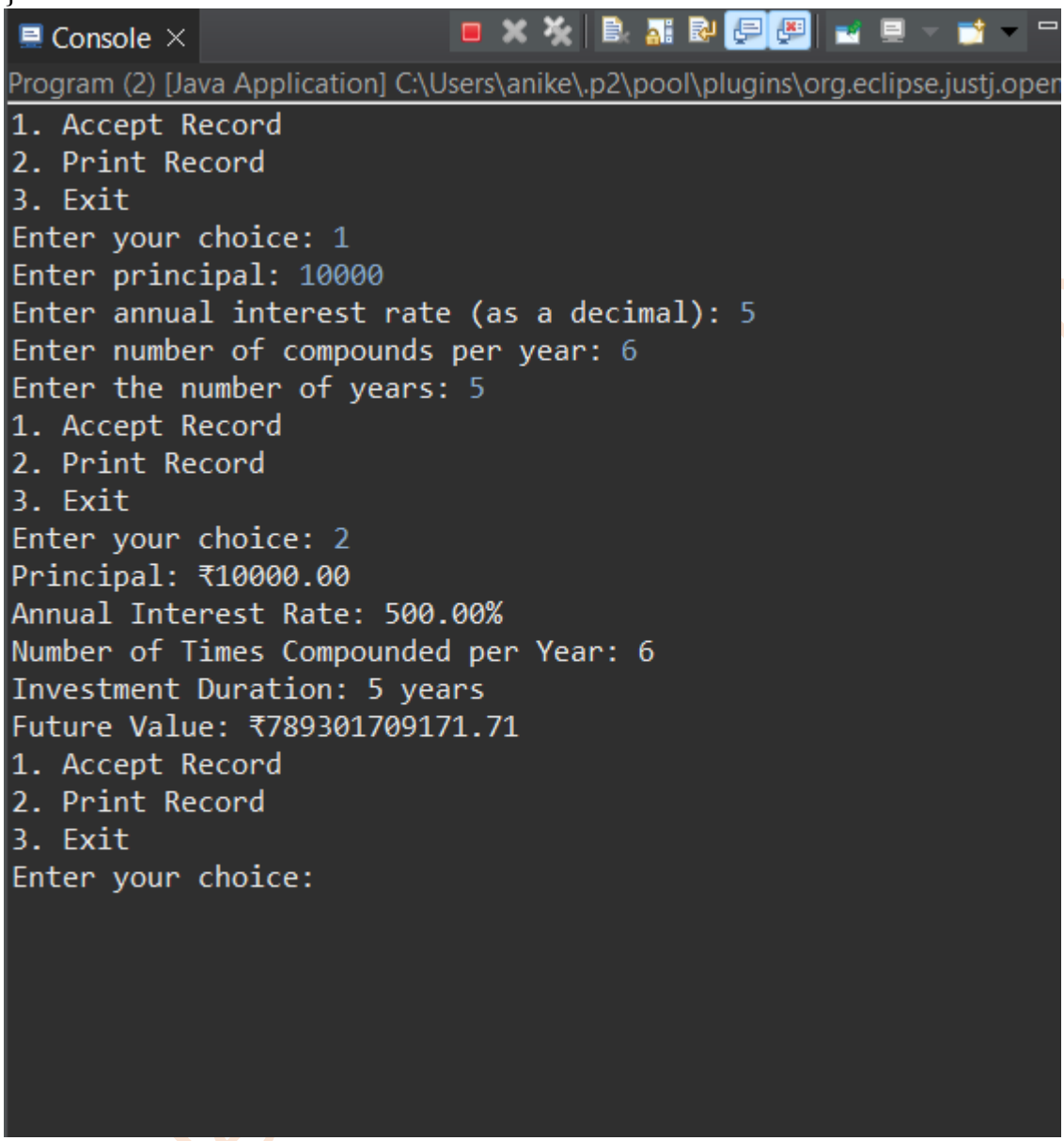
```
package org.assignment4;

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);
        }
        @Override
        public String toString() {
        return String.format("Principal: ₹%.2f\nAnnual Interest Rate: %.2f%%\nNumber of
Times Compounded per Year: %d\nInvestment Duration: %d years",
        principal, annualInterestRate * 100, numberOfCompounds, years);
        }
}
```

CompoundInterestCalculatorUtil.java:

```
package org.assignment4;

import java.util.Scanner;

public class CompoundInterestCalculatorUtil {private CompoundInterestCalculator
calculator;
private Scanner sc = new Scanner(System.in);
public void menuList() {
System.out.println("1. Accept Record");
System.out.println("2. Print Record");
System.out.println("3. Exit");
}
public void acceptRecord() {
System.out.print("Enter principal: ");
double principal = sc.nextDouble();
System.out.print("Enter annual interest rate (as a decimal): ");
double annualInterestRate = sc.nextDouble();
System.out.print("Enter number of compounds per year: ");
int numberOfCompounds = sc.nextInt();
System.out.print("Enter the number of years: ");
int years = sc.nextInt();
calculator = new CompoundInterestCalculator(principal, annualInterestRate,
numberOfCompounds, years);
}
public void printRecord() {
if (calculator != null) {
System.out.println(calculator.toString());
double futureValue = calculator.calculateFutureValue();
System.out.printf("Future Value: ₹%.2f\n", futureValue);
} else {
System.out.println("No record to display.");
}
```

```
}
}
```



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.

BMITracker.java:

```
package org.assignment4;


public class BMITracker {

	private double weight;

	private double height;

	private double BMI;

	public String Classification;

	public BMITracker(double weight, double height) {

	this.weight = weight;

	this.height = height;

	this.BMI = calculateBMI();

	this.Classification = classifyBMI();

	}

	public double getWeight() {

	return weight;

	}

	public void setWeight(double weight) {

	this.weight = weight;
```

```
this.BMI = calculateBMI();
this.Classification = classifyBMI();
}
public double getHeight() {
return height;
}
public void setHeight(double height) {
this.height = height;
this.BMI = calculateBMI();
this.Classification = classifyBMI();
}
public double calculateBMI() {
return weight / (height * height);
}
public String classifyBMI() {
if (BMI < 18.5) {
return "Underweight";
} else if (BMI >= 18.5 && BMI < 24.9) {
return "Normal weight";
} else if (BMI >= 25 && BMI < 29.9) {
return "Overweight";
} else {
return "Obese";
}
}
```

```
	public double getBMI() {

	return BMI;

	}

	public String getClassification() {

	return Classification;

	}

	@Override

	public String toString() {

	return "BMITracker [BMI=" + BMI + ", Classification=" +
Classification + "]";

	}

}
```

BMITrackerUtil.java:

```
package org.assignment4;


import java.util.Scanner;


public class BMITrackerUtil {
   private double weight;
   private double height;
   private double BMI;
   private String Classification;
   Scanner sc = new Scanner(System.in);


   public void acceptRecord() {
```

```
        System.out.println("Enter Weight (in kg): ");
        weight = sc.nextDouble();
        System.out.println("Enter Height (in meters): ");
        height = sc.nextDouble();

        calculateBMI();
        classifyBMI();
    }

    public void calculateBMI() {
        if (height > 0) {
            BMI = weight / (height * height);
        } else {
            System.out.println("Height cannot be zero or negative!");
        }
    }

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

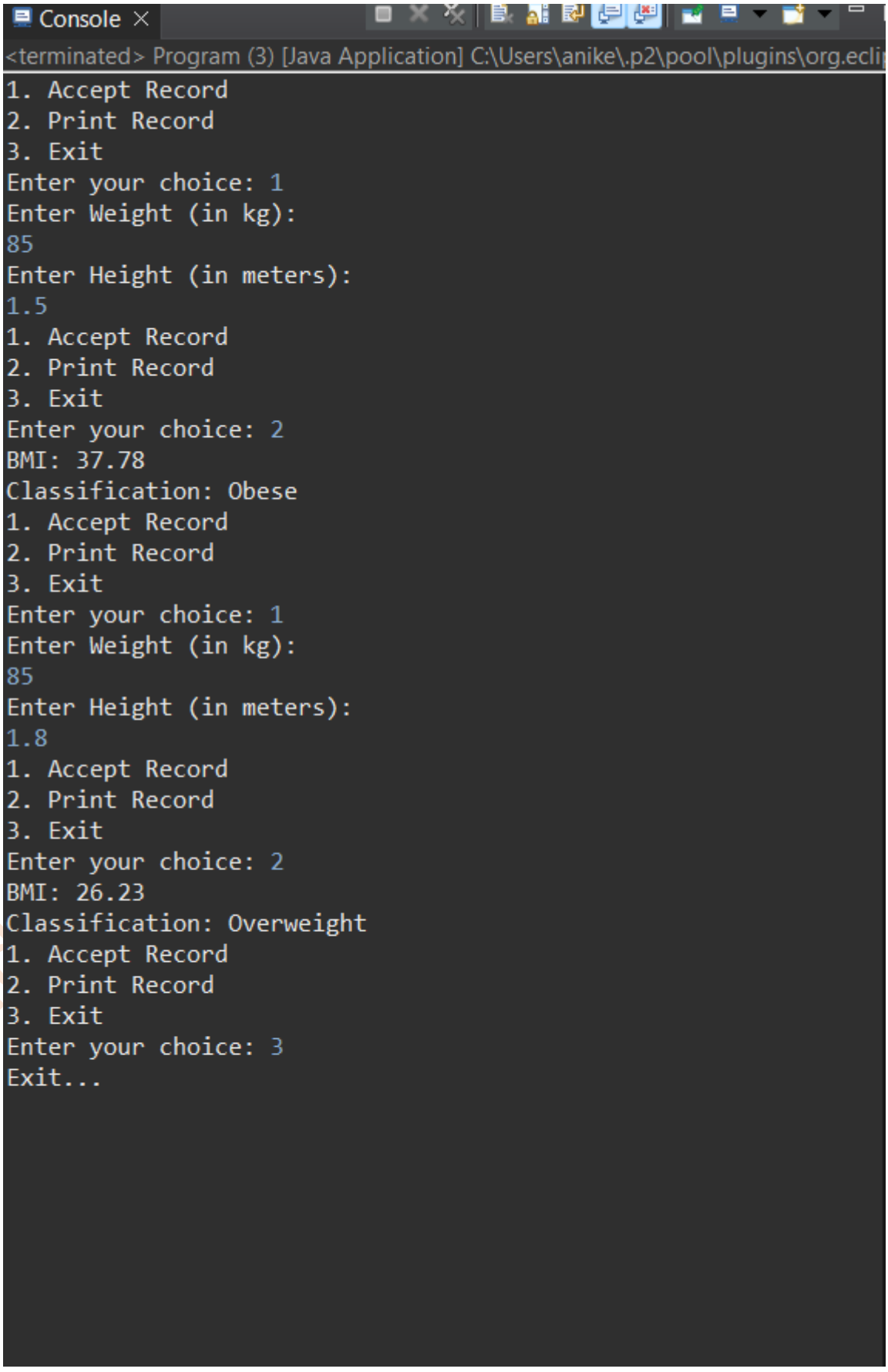
```
            Classification = "Obese";
        }
    }

    public void printRecord() {
        if (BMI > 0) {
            System.out.printf("BMI: %.2f%n", BMI);
            System.out.println("Classification: " + Classification);
        } else {
            System.out.println("No valid BMI record found. Please input weight
and height first.");
        }
    }

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

Program.java:

```
package org.assignment4;

import java.util.Scanner;
```

```
public class Program {

    public static void main(String[] args) {

        BMITrackerUtil util = new BMITrackerUtil();

        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("Exit...");

        break;

        default:

        System.out.println("Invalid choice. Please try again.");

        }

        } while (choice != 3);

        sc.close();

        }
```

```
}
```


Console
<terminated> Program (3) [Java Application] C:\Users\anike\.p2\pool\plugins\org.ecli
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 1
Enter Weight (in kg):
85
Enter Height (in meters):
1.5
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 2
BMI: 37.78
Classification: Obese
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 1
Enter Weight (in kg):
85
Enter Height (in meters):
1.8
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 2
BMI: 26.23
Classification: Overweight
1. Accept Record
2. Print Record
3. Exit
Enter your choice: 3
Exit...

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.

DiscountCalculator.java:

```
package org.assignment4;

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;
	this.discountAmount = calculateDiscountAmount();
	this.finalPrice = calculateFinalPrice();
	}
	public double getOriginalPrice() {
	return originalPrice;
	}
	public void setOriginalPrice(double originalPrice) {
	this.originalPrice = originalPrice;
	}
	public double getDiscountRate() {
	return discountRate;
	}
	public void setDiscountRate(double discountRate) {
	this.discountRate = discountRate;
	}
	public double getDiscountAmount() {
	return discountAmount;
	}
```

```
	public double getFinalPrice() {
	return finalPrice;
	}
	public double calculateDiscountAmount() {
	return originalPrice * (discountRate / 100);
	}
	public double calculateFinalPrice() {
	return originalPrice - discountAmount;
	}
	@Override
	public String toString() {
	return String.format("Original Price: ₹%.2f\nDiscount Rate: %.2f%%\nDiscount
Amount: ₹%.2f\nFinal Price: ₹%.2f",
	originalPrice, discountRate, discountAmount, finalPrice);
	}

}
```

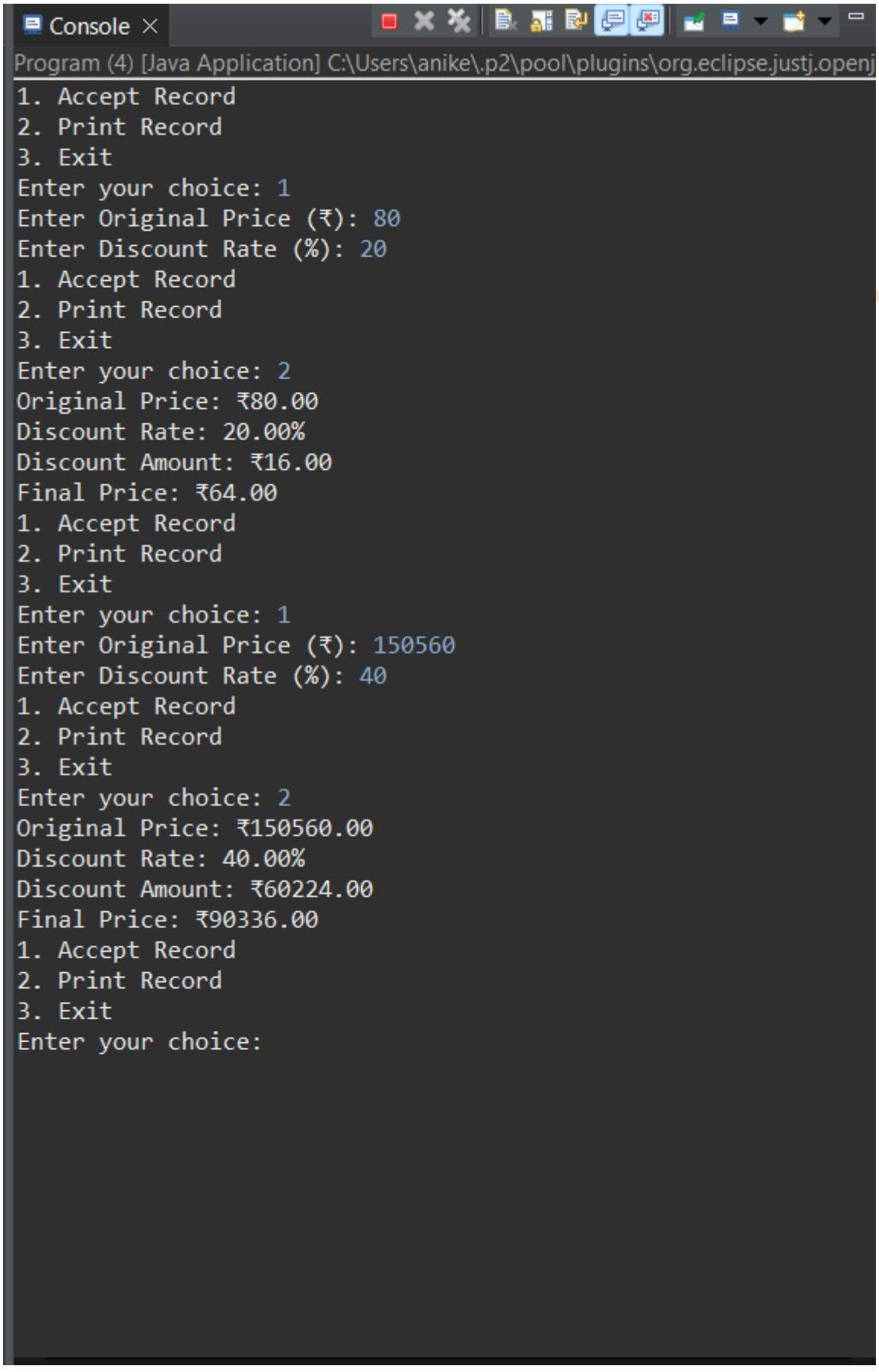
DiscountCalculatorUtil.java:

```
package org.assignment4;

import java.util.Scanner;

public class DiscountCalculatorUtil {
	private DiscountCalculator discountCalculator;
	Scanner sc = new Scanner(System.in);
	public void acceptRecord() {
	System.out.print("Enter Original Price (₹): ");
	double originalPrice = sc.nextDouble();
	System.out.print("Enter Discount Rate (%): ");
	double discountRate = sc.nextDouble();
	discountCalculator = new DiscountCalculator(originalPrice, discountRate);
	}
	public void printRecord() {
	if (discountCalculator != null) {
	System.out.println(discountCalculator.toString());
	} else {
	System.out.println("No record found. Please enter the original price and discount rate
first.");
	}
	}
	public void menuList() {
	System.out.println("1. Accept Record");
	System.out.println("2. Print Record");
	System.out.println("3. Exit");
	}
}
```

Program,java:

```
package org.assignment4;

import java.util.Scanner;

public class Program {
	public static void main(String[] args) {
		DiscountCalculatorUtil util = new DiscountCalculatorUtil();
		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("Exit...");
		break;
		default:
		System.out.println("Invalid choice. Please try again.");
		}
		} while (choice != 3);
		sc.close();
		}

}
```

ASSIGNMENT NO.4



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.

TollBoothRevenueManager.java:

```
package org.assignment4;

public class TollBoothRevenueManager {
	private double carTollRate;
	private double truckTollRate;
	private double motorcycleTollRate;
	private int numCars;
	private int numTrucks;
	private int numMotorcycles;
	private double totalRevenue;
	public TollBoothRevenueManager(double carTollRate, double truckTollRate, double
motorcycleTollRate) {
	this.carTollRate = carTollRate;
	this.truckTollRate = truckTollRate;
	this.motorcycleTollRate = motorcycleTollRate;
	this.numCars = 0;
	this.numTrucks = 0;
	this.numMotorcycles = 0;
	this.totalRevenue = 0;
	}
	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 getNumCars() {
return numCars;
}
public void setNumCars(int numCars) {
this.numCars = numCars;
}
public int getNumTrucks() {
return numTrucks;
}
public void setNumTrucks(int numTrucks) {
this.numTrucks = numTrucks;
}
public int getNumMotorcycles() {
return numMotorcycles;
}
public void setNumMotorcycles(int numMotorcycles) {
this.numMotorcycles = numMotorcycles;
}
public double getTotalRevenue() {
return totalRevenue;
}
public double calculateTotalRevenue() {
totalRevenue = (numCars * carTollRate) + (numTrucks * truckTollRate) +
(numMotorcycles * motorcycleTollRate);
return totalRevenue;
}
public int getTotalVehicles() {
return numCars + numTrucks + numMotorcycles;
}
@Override
public String toString() {
return String.format("Toll Rates:\nCar: Rs.%.2f\nTruck: Rs.%.2f\nMotorcycle:
Rs.%.2f\n\nTotal Vehicles: %d\nTotal Revenue: Rs.%.2f",
carTollRate, truckTollRate, motorcycleTollRate, getTotalVehicles(), totalRevenue);
}
```

```
}
```

TollBoothRevenueManagerUtil.java:

```
package org.assignment4;

import java.util.Scanner;

public class TollBoothRevenueManagerUtil {
	private TollBoothRevenueManager tollBoothManager;
	Scanner sc = new Scanner(System.in);
	public void acceptRecord() {
	System.out.print("Enter Toll Rate for Car (₹): ");
	double carRate = sc.nextDouble();
	System.out.print("Enter Toll Rate for Truck (₹): ");
	double truckRate = sc.nextDouble();
	System.out.print("Enter Toll Rate for Motorcycle (₹): ");
	double motorcycleRate = sc.nextDouble();
	tollBoothManager = new TollBoothRevenueManager(carRate, truckRate,
motorcycleRate);
	System.out.print("Enter Number of Cars: ");
	tollBoothManager.setNumCars(sc.nextInt());
	System.out.print("Enter Number of Trucks: ");
	tollBoothManager.setNumTrucks(sc.nextInt());
	System.out.print("Enter Number of Motorcycles: ");
	tollBoothManager.setNumMotorcycles(sc.nextInt());
	tollBoothManager.calculateTotalRevenue();
	}
	public void printRecord() {
	if (tollBoothManager != null) {
	System.out.println(tollBoothManager.toString());
	} else {
	System.out.println("No record found. Please enter the toll rates and vehicle numbers
first.");
	}
	}
	public void menuList() {
	System.out.println("1. Accept Record");
	System.out.println("2. Print Record");
	System.out.println("3. Exit");
	}
}
```

Program.java:

```
package org.assignment4;
```

ASSIGNMENT NO.4

```
import java.util.Scanner;

public class Program {
	public static void main(String[] args) {
		TollBoothRevenueManagerUtil util = new TollBoothRevenueManagerUtil();
		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("Exit...");
		break;
		default:
		System.out.println("Invalid choice. Please try again.");
		}
		} while (choice != 3);
		sc.close();
		}
}
```

ASSIGNMENT NO.4