Assignment 4

Getters and Setters Method

**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.

**package** cdac.daily;

**import** java.util.Scanner;

**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;

**double** monthlyPayment = principal \* (monthlyInterestRate \* Math.*pow*(1 + monthlyInterestRate, numberOfMonths)) /

(Math.*pow*(1 + monthlyInterestRate, numberOfMonths) - 1);

**return** monthlyPayment;

}

**public** **double** totalLoanPayment() {

**return** calculateMonthlyPayment() \* loanTerm \* 12;

}

**public** String toString() {

**return** "Principal Amount : "+principal+ "\n"+

"Annual Interest Rate : "+ annualInterestRate + "\n"+

"Loan term : "+ loanTerm +" years\n"+

"Monthly payment : "+ calculateMonthlyPayment() +"\n"+

"Total Payment : "+ totalLoanPayment();

}

}

**class** LoanAmortizationCalculatorUtil{

**private** LoanAmortizationCalculator loancal;

Scanner sc = **new** Scanner(System.***in***);

**public** **void** acceptRecord() {

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 terms in year: ");

**int** loanTerm = sc.nextInt();

loancal = **new** LoanAmortizationCalculator(principal, annualInterestRate, loanTerm);

}

**public** **void** printRecord() {

**if** (loancal != **null**) {

System.***out***.println("\nLoan Amortization Details:");

System.***out***.println(loancal.toString());

}

**else** {

System.***out***.println("Please enter loan details first!");

}

}

**public** **int** menuList() {

System.***out***.println("0. Exit");

System.***out***.println("1. Enter Loan Details");

System.***out***.println("2. Show Loan Amortization");

System.***out***.println("Enter Choise : ");

**int** choise = sc.nextInt();

**return** choise;

}

}

**public** **class** LoanProgram {

**public** **static** **void** main(String[] args) {

LoanAmortizationCalculatorUtil utilobj = **new** LoanAmortizationCalculatorUtil();

**int** choise;

**while**((choise = utilobj.menuList()) != 0){

**switch**(choise){

**case** 1:

utilobj.acceptRecord();

**break**;

**case** 2:

utilobj.printRecord();

**break**;

**default**:

System.***out***.println("Invalid choise please try again.");

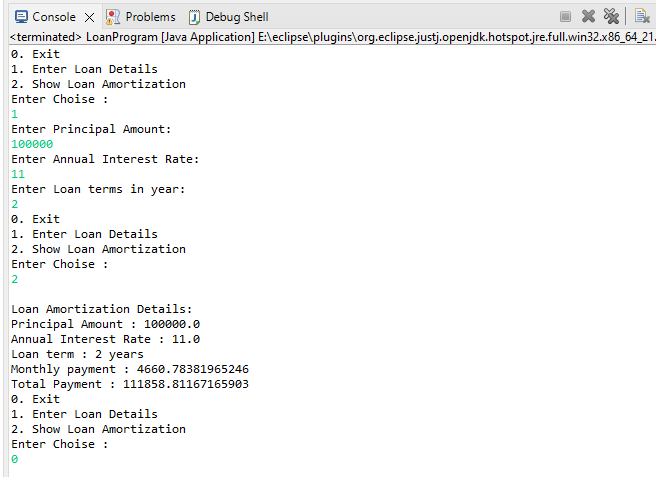
**break**;

}

}

}

}



**2. Compound Interest Calculator for Investment**

Develop a system to compute the future value of an investment with compound interest. The system should:

1. Accept the initial investment amount, annual interest rate, number of times the interest is compounded per year, and investment duration (in years) from the user.
2. Calculate the future value of the investment using the formula:
   * **Future Value Calculation:**
     + futureValue = principal \* (1 + annualInterestRate / numberOfCompounds)^(numberOfCompounds \* years)
   * **Total Interest Earned:** totalInterest = futureValue - principal
3. Display the future value and the total interest earned, in Indian Rupees (₹).

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

**package** cdac.daily;

**import** java.util.Scanner;

**class** CompoundInterestCalculator1 {

**private** **double** principal;

**private** **double** annualInterestRate;

**private** **int** numberOfCompounds;

**private** **int** years;

**public** CompoundInterestCalculator1(**double** principal, **double** annualInterestRate, **int** numberOfCompounds, **int** years) {

**super**();

**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;

}

// Calculations

**public** **double** calculateFutureValue() {

**return**(principal \* Math.*pow*(1 + (annualInterestRate / 100) / numberOfCompounds, numberOfCompounds \* years));

}

**public** **double** totalInterestEarned() {

**return**(calculateFutureValue() - principal);

}

**public** String toString() {

**return** String.*format*("Future Value of Investment : %.2f \nTotal Interest earned : %.2f", calculateFutureValue(), totalInterestEarned());

}

}

**class** CompoundInterestCalculatorUtil {

**private** CompoundInterestCalculator1 cical;

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

**public** **void** acceptRecord() {

System.***out***.println("Enter Initial Investment Amount : ");

**double** principal = *sc*.nextDouble();

System.***out***.println("Enter Annual Interest rate : ");

**double** annualInterestRate = *sc*.nextDouble();

System.***out***.println("Enter number of times the interest is compounded per year");

**int** numberOfCompounds = *sc*.nextInt();

System.***out***.println("Enter the investment duration : ");

**int** years = *sc*.nextInt();

cical = **new** CompoundInterestCalculator1( principal, annualInterestRate, numberOfCompounds, years);

}

**public** **void** printRecord() {

System.***out***.printf(cical.toString());

}

**public** **int** menuList() {

System.***out***.println("\n\n 0. Exit");

System.***out***.println("1. Enter Interest Details");

System.***out***.println("2. Show the Future value of Investment and the Total Interest Earned");

System.***out***.println("Enter Choise : ");

**int** choise = *sc*.nextInt();

**return** choise;

}

}

**public** **class** CIProgram {

**public** **static** **void** main(String[] args) {

CompoundInterestCalculatorUtil utilobj = **new** CompoundInterestCalculatorUtil();

**int** choise;

**while**((choise = utilobj.menuList()) != 0){

**switch**(choise){

**case** 1:

utilobj.acceptRecord();

**break**;

**case** 2:

utilobj.printRecord();

**break**;

**default**:

System.***out***.println("Invalid choise please try again.");

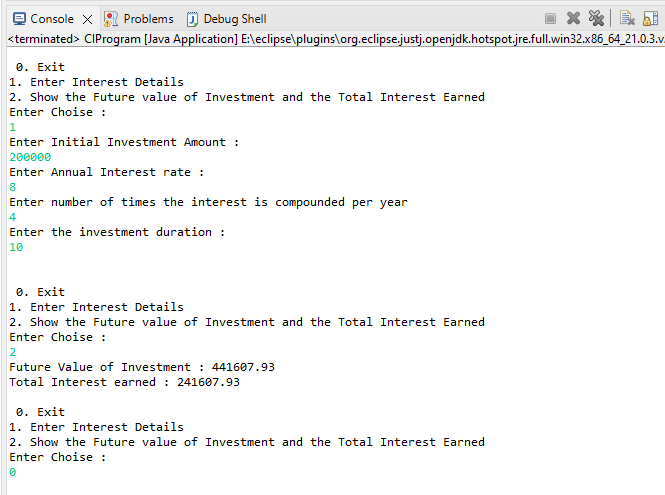
**break**;

}

}

}

}



**3. BMI (Body Mass Index) Tracker**

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

1. Accept weight (in kilograms) and height (in meters) from the user.
2. Calculate the BMI using the formula:
   * **BMI Calculation:** BMI = weight / (height \* height)
3. Classify the BMI into one of the following categories:
   * Underweight: BMI < 18.5
   * Normal weight: 18.5 ≤ BMI < 24.9
   * Overweight: 25 ≤ BMI < 29.9
   * Obese: BMI ≥ 30
4. Display the BMI value and its classification.

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

**package** cdac.daily;

**import** java.util.Scanner;

**class** BMITracker1{

**private** **double** weight;

**private** **double** height;

**public** BMITracker1(**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;

}

// Calculations

**public** **double** calculateBMI() {

**double** bmi= weight / (height \* height);

**return** bmi;

}

**public** String classifyBMI(**double** bmi) {

**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** String toString() {

**double** bmi = calculateBMI();

**return** String.*format*("Your BMI is : %.2f\n BMI Classification : ", calculateBMI(), classifyBMI(bmi));

}

}

**class** BMITrackerUtil {

**private** BMITracker1 bmical;

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

**public** **void** acceptRecord() {

System.***out***.println("Enter your weight in kilograms : ");

**double** weight = *sc*.nextDouble();

System.***out***.println("Enter your height in meters : ");

**double** height = *sc*.nextDouble();

bmical = **new** BMITracker1(weight, height);

}

**public** **void** printRecord() {

**double** bmi = bmical.calculateBMI();

System.***out***.printf("Your BMI is : %.2f\n", bmi);

System.***out***.printf("BMI Classification : " + bmical.classifyBMI(bmi));

}

**public** **int** menuList() {

System.***out***.println("\n\n 0. Exit");

System.***out***.println("1. Enter the details to calculate BMI");

System.***out***.println("2. Calculate the BMI and show in which category I exist");

System.***out***.println("Enter Choise : ");

**int** choise = *sc*.nextInt();

**return** choise;

}

}

**public** **class** BMIProgram {

**public** **static** **void** main(String[] args) {

BMITrackerUtil utilobj = **new** BMITrackerUtil();

**int** choise;

**while**((choise = utilobj.menuList()) != 0){

**switch**(choise){

**case** 1:

utilobj.acceptRecord();

**break**;

**case** 2:

utilobj.printRecord();

**break**;

**default**:

System.***out***.println("Invalid choise please try again.");

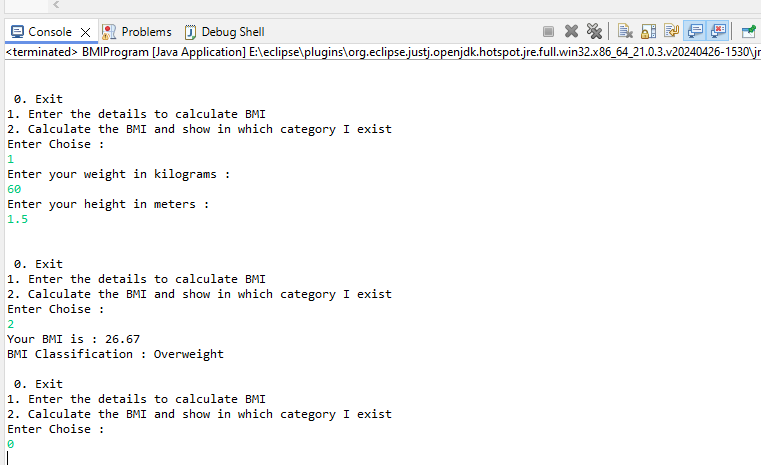
**break**;

}

}

}

}



**4. Discount Calculation for Retail Sales**

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

1. Accept the original price of an item and the discount percentage from the user.
2. Calculate the discount amount and the final price using the following formulas:
   * **Discount Amount Calculation:** discountAmount = originalPrice \* (discountRate / 100)
   * **Final Price Calculation:** finalPrice = originalPrice - discountAmount
3. Display the discount amount and the final price of the item, in Indian Rupees (₹).

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

**package** cdac.daily;

**import** java.util.\*;

**class** DiscountCalculator1 {

**private** **double** originalPrice;

**private** **double** discountRate;

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

**public** DiscountCalculator1(**double** originalPrice, **double** discountRate) {

**this**.originalPrice = originalPrice;

**this**.discountRate = discountRate;

}

**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** calculateDiscount () {

**return** originalPrice \* (discountRate / 100);

}

**public** String toString() {

**double** discountAmount = calculateDiscount();

**return** String.*format*("Discount Amount : %.2f\n Final Price : %.2f\n", discountAmount, (originalPrice - discountAmount));

}

}

**class** DiscountCalculatorUtil {

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

DiscountCalculator1 dcal;

**public** **void** acceptRecord() {

System.***out***.println("Enter original price of an item : ");

**double** originalPrice = *sc*.nextDouble();

System.***out***.println("Enter discount percentage : ");

**double** discountRate = *sc*.nextDouble();

dcal = **new** DiscountCalculator1(originalPrice, discountRate);

}

**public** **void** printRecord() {

System.***out***.printf(dcal.toString());

}

**public** **int** menuList() {

System.***out***.println("\n\n 0. Exit");

System.***out***.println("1. Enter the details to check the Distcount");

System.***out***.println("2. Calculate Discount Amount and the Final Amount");

System.***out***.println("Enter Choise : ");

**int** choise = *sc*.nextInt();

**return** choise;

}

}

**public** **class** RetailProgram {

**public** **static** **void** main(String[] args) {

DiscountCalculatorUtil utilobj = **new** DiscountCalculatorUtil();

**int** choise;

**while**((choise = utilobj.menuList()) != 0){

**switch**(choise){

**case** 1:

utilobj.acceptRecord();

**break**;

**case** 2:

utilobj.printRecord();

**break**;

**default**:

System.***out***.println("Invalid choise please try again.");

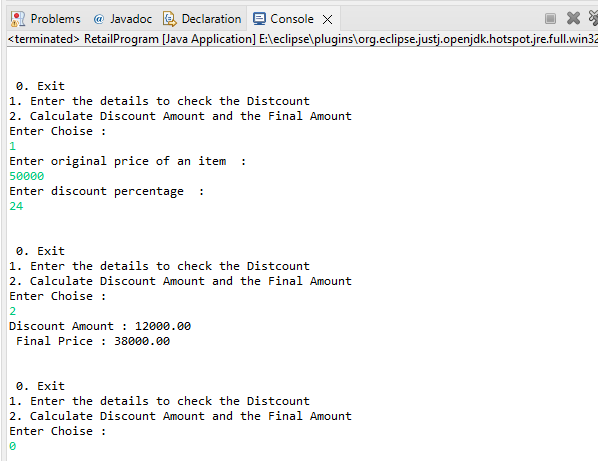
**break**;

}

}

}

}



**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.

**package** cdac.daily;

**import** java.util.Scanner;

**class** TollBoothRevenueManager1 {

**private** **double** carRate, truckRate, motorcycleRate;

**private** **int** carCount, truckCount, motorcycleCount;

**public** TollBoothRevenueManager1(**double** carRate, **double** truckRate, **double** motorcycleRate, **int** carCount,

**int** truckCount, **int** motorcycleCount) {

**super**();

**this**.carRate = carRate;

**this**.truckRate = truckRate;

**this**.motorcycleRate = motorcycleRate;

**this**.carCount = carCount;

**this**.truckCount = truckCount;

**this**.motorcycleCount = motorcycleCount;

}

**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;

}

// Calculations

**public** **double** calculateRevenue () {

**return** (carRate \* carCount) + (truckRate \* truckCount) + (motorcycleRate \* motorcycleCount);

}

**public** String toString() {

**int** totalVehicles = carCount + truckCount + motorcycleCount;

**double** totalRevenue = calculateRevenue();

**return** String.*format*("Total number of vehicles : %d\nTotal revenue collected: ₹ %.2f\\n", totalVehicles, totalRevenue);

}

}

**class** TollBoothRevenueManagerUtil {

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

TollBoothRevenueManager1 tbrm;

**public** **void** acceptRecord() {

System.***out***.print("Toll rate for Car : ");

**double** carRate = *sc*.nextDouble();

System.***out***.print("Toll rate for Truck : ");

**double** truckRate = *sc*.nextDouble();

System.***out***.print("Toll rate for Motorcycle : ");

**double** motorcycleRate = *sc*.nextDouble();

System.***out***.print("Enter the number of Cars: ");

**int** carCount = *sc*.nextInt();

// Accepting the number of trucks

System.***out***.print("Enter the number of Trucks: ");

**int** truckCount = *sc*.nextInt();

// Accepting the number of motorcycles

System.***out***.print("Enter the number of Motorcycles: ");

**int** motorcycleCount = *sc*.nextInt();

tbrm = **new** TollBoothRevenueManager1(carRate, truckRate, motorcycleRate, carCount, truckCount, motorcycleCount);

}

**public** **void** printRecord() {

System.***out***.printf(tbrm.toString());

}

**public** **int** menuList() {

System.***out***.println("\n\n 0. Exit");

System.***out***.println("1. Enter the details of Vehicle Toll rate and there count");

System.***out***.println("2. Show the Revenue Details ");

System.***out***.println("Enter Choise : ");

**int** choise = *sc*.nextInt();

**return** choise;

}

}

**public** **class** TollBoothProgramm {

**public** **static** **void** main(String[] args) {

TollBoothRevenueManagerUtil utilobj = **new** TollBoothRevenueManagerUtil();

**int** choise;

**while**((choise = utilobj.menuList()) != 0){

**switch**(choise){

**case** 1:

utilobj.acceptRecord();

**break**;

**case** 2:

utilobj.printRecord();

**break**;

**default**:

System.***out***.println("Invalid choise please try again.");

**break**;

}

}

}

}

