Assignment : 4

**Que1:**

**Solution :-**

1. **Loan Amortization Calculator :**

**public** **class** LoanAmortizationCalculator {

**private** **static** **double** *Principal*;

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

**private** **static** **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** calculateTotalAmountPaid() {

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

}

**public** String toString() {

**return** String.*format*("Principal: ₹%.2f\nAnnual Interest Rate: %.2f%%\nLoan Term: %d years",

*Principal*, *annualInterestRate*, *LoanTerm*);

}

1. **Compound Interest Calculator for Investment :**

**package** program13;

**import** java.util.Scanner;

**public** **class** CompoundInterestCalculatorUtil {

**public** **static** CompoundInterestCalculator getUserInput() {

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

System.***out***.print("Enter the initial investment amount (₹): ");

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

System.***out***.print("Enter the annual interest rate (as a decimal, e.g., 0.05 for 5%): ");

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

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** **class** Program {

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

CompoundInterestCalculator calculator = CompoundInterestCalculatorUtil.*getUserInput*();

**double** futureValue = calculator.calculateFutureValue();

**double** totalInterest = calculator.calculateTotalInterest();

System.***out***.println("Investment Details:");

System.***out***.println(calculator);

System.***out***.println("Future Value: ₹" + String.*format*("%.2f", futureValue));

System.***out***.println("Total Interest Earned: ₹" + String.*format*("%.2f", totalInterest));

}

}

}

3.**package** program14;

**public** **class** CoumpoundInterestCalculator {

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

CompoundInterestCalculator calculator = CompoundInterestCalculatorUtil.getUserInput();

**double** futureValue = calculator.calculateFutureValue();

**double** totalInterest = calculator.calculateTotalInterest();

System.***out***.println("Investment Details:");

System.***out***.println(calculator);

System.***out***.println("Future Value: ₹" + String.*format*("%.2f", futureValue));

System.***out***.println("Total Interest Earned: ₹" + String.*format*("%.2f", totalInterest));

}

}

Output :

Investment Details:

Principal: ₹10000.00, Annual Interest Rate: 5.00%, Compounded: 4 times/year, Duration: 3 years

Future Value: ₹11274.94

Total Interest Earned: ₹1274.94

1. **BMI (Body Mass Index) Tracker:**

Solu: -

1. **package** program14;

**public** **class** BMICalculator {

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

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

**public** BMICalculator(**double** weight, **double** height) {

**this**.weight = weight;

**this**.height = height;

}

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

**return** weight / (height \* height);

}

**public** String classifyBMI() {

**double** bmi = calculateBMI();

**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** "Obesity";

}

}

@Override

**public** String toString() {

**return** String.*format*("Weight: %.2f kg, Height: %.2f m", weight, height);

}

}

Output :

Enter weight (in kilograms): 70

Enter height (in meters): 1.75

BMI Details:

Weight: 70.00 kg, Height: 1.75 m

BMI: 22.86

Classification: Normal weight

1. **Discount Calculation for Retail Sales :**

**Solu:-**

**package** program14;

**public** **class** DiscountCalculator {

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

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

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

**this**.originalPrice = originalPrice;

**this**.discountRate = discountRate;

}

**public** **double** calculateDiscountAmount() {

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

}

**public** **double** calculateFinalPrice() {

**return** originalPrice - calculateDiscountAmount();

}

**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;

}

@Override

**public** String toString() {

**return** String.*format*("Original Price: ₹%.2f, Discount Rate: %.2f%%", originalPrice, discountRate);

}

}

Output:

Enter the original price of the item (₹): 1000

Enter the discount percentage: 20

Discount Details:

Original Price: ₹1000.00, Discount Rate: 20.00%

Discount Amount: ₹200.00

Final Price: ₹800.00

1. **Toll Booth Revenue Management :-**

**Solu:-**

**package** program15;

**public** **class** TollBoothRevenueManager {

**private** **double** carRate;

**private** **double** truckRate;

**private** **double** motorcycleRate;

**private** **int** carCount;

**private** **int** truckCount;

**private** **int** motorcycleCount;

// Constructor

**public** TollBoothRevenueManager(**double** carRate, **double** truckRate, **double** motorcycleRate) {

**this**.carRate = carRate;

**this**.truckRate = truckRate;

**this**.motorcycleRate = motorcycleRate;

}

**public** **int** getTruckCount() {

**return** truckCount;

}

**public** **void** setTruckCount(**int** truckCount) {

**this**.truckCount = truckCount;

}

**public** **int** getMotorcycleCount() {

**return** motorcycleCount;

}

**public** String toString() {

**return** String.*format*("Car Rate: ₹%.2f, Truck Rate: ₹%.2f, Motorcycle Rate: ₹%.2f\n" +

"Cars: %d, Trucks: %d, Motorcycles: %d",

carRate, truckRate, motorcycleRate, carCount, truckCount, motorcycleCount);

}

**public** **void** setMotorcycleCount(**int** motorcycleCount) {

**this**.motorcycleCount = motorcycleCount;

}

}

**Output:-**

Toll Booth Summary:

Car Rate: ₹50.00, Truck Rate: ₹100.00, Motorcycle Rate: ₹30.00

Cars: 120, Trucks: 30, Motorcycles: 50

Total Revenue: ₹10500.00

Total Vehicles: 200