**Assignment 3**

**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 class LoanAmortizationCalculator with methods acceptRecord, calculateMonthlyPayment & printRecord and test the functionality in main method.

**Code:**

package org.example;

import java.util.Scanner;

public class LoanCalculator {

private double principleAmount;

private double annualRate;

private double loanTerm;

private double numberOfMonths;

private double monthlyInterestRate;

private double monthlyPayment;

public double getPrincipleAmount() {

return principleAmount;

}

public void setPrincipleAmount(double principleAmount) {

this.principleAmount = principleAmount;

}

public double getAnnualRate() {

return annualRate;

}

public void setAnnualRate(double annualRate) {

this.annualRate = annualRate;

}

public double getLoanTerm() {

return loanTerm;

}

public void setLoanTerm(double loanTerm) {

this.loanTerm = loanTerm;

}

public double getNumberOfMonths() {

return numberOfMonths;

}

public double getMonthlyInterestRate() {

return monthlyInterestRate;

}

public double getMonthlyPayment() {

return monthlyPayment;

}

public void acceptRecord() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter principleAmount: ");

setPrincipleAmount(sc.nextDouble());

System.out.print("Enter annualRate: ");

setAnnualRate(sc.nextDouble());

System.out.print("Enter loanTerm : ");

setLoanTerm(sc.nextDouble());

sc.close();

}

public void calculateMonthlyPayment() {

this.numberOfMonths = getLoanTerm() \* 12;

this.monthlyInterestRate = getAnnualRate() / 12 / 100;

double nume = getPrincipleAmount() \* monthlyInterestRate \* Math.pow(1 + monthlyInterestRate, numberOfMonths);

double deno = Math.pow(1 + monthlyInterestRate, numberOfMonths) - 1;

this.monthlyPayment = nume / deno;

}

public void displayRecord() {

System.out.printf("Number of months: %.3f%n", getNumberOfMonths());

System.out.printf("Monthly interest rate: %.3f%%%n", getMonthlyInterestRate() \* 100);

System.out.printf("Monthly payment: %.3f%n", getMonthlyPayment());

}

public static void main(String[] args) {

LoanCalculator calculator = new LoanCalculator();

calculator.acceptRecord();

calculator.calculateMonthlyPayment();

calculator.displayRecord();

}

}

**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 class CompoundInterestCalculator with methods acceptRecord , calculateFutureValue, printRecord and test the functionality in main method.

**Code:**

import java.util.Scanner;

class CompoundInterestCalculator {

private double principal;

private double annualInterestRate;

private int numberOfCompounds;

private int years;

public void acceptRecord() {

Scanner scan = new Scanner(System.in);

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

principal = scan.nextDouble();

System.out.print("Enter the annual interest rate (in %): ");

annualInterestRate = scan.nextDouble();

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

numberOfCompounds = scan.nextInt();

System.out.print("Enter the investment duration (in years): ");

years = scan.nextInt();

}

public double calculateFutureValue() {

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

}

public void printRecord() {

double futureValue = calculateFutureValue();

double totalInterest = futureValue - principal;

System.out.printf("Future Value of Investment: %.2f\n", futureValue);

System.out.printf("Total Interest Earned: %.2f\n", totalInterest);

}

public static void main(String[] args) {

CompoundInterestCalculator calculator = new CompoundInterestCalculator();

calculator.acceptRecord();

calculator.printRecord();

}

}

**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 class BMITracker with methods acceptRecord, calculateBMI, classifyBMI & printRecord and test the functionality in main method.

**Code:**

import java.util.Scanner;

class BMITracker {

private double weight;

private double height;

public void acceptRecord() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter your weight (in kg): ");

weight = scanner.nextDouble();

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

height = scanner.nextDouble();

}

public double calculateBMI() {

return weight / (height \* height);

}

public String classifyBMI(double bmi) {

if (bmi < 18.5) {

return "Underweight";

} else if (bmi < 24.9) {

return "Normal weight";

} else if (bmi < 29.9) {

return "Overweight";

} else {

return "Obese";

}

}

public void printRecord() {

double bmi = calculateBMI();

String classification = classifyBMI(bmi);

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

System.out.println("BMI Classification: " + classification);

}

public static void main(String[] args) {

BMITracker tracker = new BMITracker();

tracker.acceptRecord();

tracker.printRecord();

}

}

**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 class DiscountCalculator with methods acceptRecord, calculateDiscount & printRecord and test the functionality in main method.

**Code:**

import java.util.Scanner;

class DiscountCalculator {

private double originalPrice;

private double discountRate;

public void acceptRecord() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the original price of the item: ");

originalPrice = scanner.nextDouble();

System.out.print("Enter the discount percentage: ");

discountRate = scanner.nextDouble();

}

public double calculateDiscount() {

return originalPrice \* (discountRate / 100);

}

public double calculateFinalPrice() {

return originalPrice - calculateDiscount();

}

public void printRecord() {

double discountAmount = calculateDiscount();

double finalPrice = calculateFinalPrice();

System.out.printf("Discount Amount: %.2f\n", discountAmount);

System.out.printf("Final Price: %.2f\n", finalPrice);

}

public static void main(String[] args) {

DiscountCalculator calculator = new DiscountCalculator();

calculator.acceptRecord();

calculator.printRecord();

}

}

**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 class TollBoothRevenueManager with methods acceptRecord, setTollRates, calculateRevenue & printRecord and test the functionality in main method.

**Code:**

import java.util.Scanner;

class TollBoothRevenueManager {

private double carTollRate;

private double truckTollRate;

private double motorcycleTollRate;

private int carCount;

private int truckCount;

private int motorcycleCount;

public void setTollRates() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the toll rate for cars: ");

carTollRate = scanner.nextDouble();

System.out.print("Enter the toll rate for trucks: ");

truckTollRate = scanner.nextDouble();

System.out.print("Enter the toll rate for motorcycles: ");

motorcycleTollRate = scanner.nextDouble();

}

public void acceptRecord() {

Scanner scanner = new Scanner(System.in);

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

carCount = scanner.nextInt();

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

truckCount = scanner.nextInt();

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

motorcycleCount = scanner.nextInt();

}

public double calculateRevenue() {

return (carCount \* carTollRate) + (truckCount \* truckTollRate) + (motorcycleCount \* motorcycleTollRate);

}

public int calculateTotalVehicles() {

return carCount + truckCount + motorcycleCount;

}

public void printRecord() {

int totalVehicles = calculateTotalVehicles();

double totalRevenue = calculateRevenue();

System.out.println("Total number of vehicles: " + totalVehicles);

System.out.printf("Total revenue collected: %.2f\n", totalRevenue);

}

public static void main(String[] args) {

TollBoothRevenueManager manager = new TollBoothRevenueManager();

manager.setTollRates();

manager.acceptRecord();

manager.printRecord();

}

}