**Note:**

* The assignment is designed to practice class, fields, and methods only.
* Create a separate project for each question.
* Do not use getter/setter methods or constructors for these assignments.
* Define two classes: one class to implement the logic and another class to test it.

**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.

Program.java

package org.example.loan;

public class Program {

public static void main(String[] args) {

AcceptRecord rec = null;

rec = new AcceptRecord( );

rec.takeRecord();

rec.calcMonthlyPay();

rec.printRecord();

AcceptRecord.*sc*.close();

}

}

AcceptRecord.java

package org.example.loan;

import java.lang.Math;

import java.util.Scanner;

public class AcceptRecord {

float loanAmm;

float anInt;

float loanTerm;

float monthlyPay;

float monthlyInt;

float noOfMonths;

float totalAm;

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

public void takeRecord() {

System.out.println("Enter loan ammount: ");

loanAmm = sc.nextLong();

System.out.println("Enter annual interest rate: ");

anInt = sc.nextFloat();

System.out.println("Enter loan term: ");

loanTerm = sc.nextFloat();

}

public void calcMonthlyPay() {

monthlyInt = (float) ((Math.pow((1.0+(anInt/100)), 1.0/12.0)) - 1.0);

noOfMonths = loanTerm\*12;

monthlyPay = (float) ((loanAmm \* monthlyInt)/(1.0 - (Math.pow((1.0+monthlyInt), -noOfMonths))));

totalAm = monthlyPay \* noOfMonths;

}

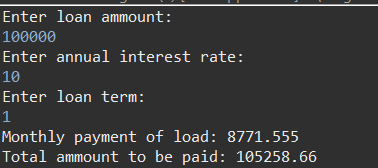
public void printRecord() {

System.out.println("Monthly payment of load: " + monthlyPay);

System.out.println("Total ammount to be paid: " + totalAm);

}

}



**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.

package org.example.comp.interest;

//package org.example.bmi;

import java.util.Scanner;

class Calculator{

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

float initInvAmm, annInt, noOfCompounded, dur;

double futureValue, totalInterest;

public void getInput() {

System.***out***.println("Enter initial investment amount: ");

initInvAmm = sc.nextLong();

System.***out***.println("Enter annual interest rate: ");

annInt = sc.nextFloat();

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

noOfCompounded = sc.nextFloat();

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

dur = sc.nextFloat();

}

public void calculateTotalEarn() {

futureValue = (initInvAmm \* Math.*pow*(((1 + annInt)/noOfCompounded), (noOfCompounded\*dur)));

totalInterest = futureValue - initInvAmm;

}

public void displayOutput() {

System.***out***.println("Future value is : " + futureValue + " Total interest earned is : " + totalInterest);

}

}

public class Program {

public static void main(String[] args) {

Calculator cl = new Calculator();

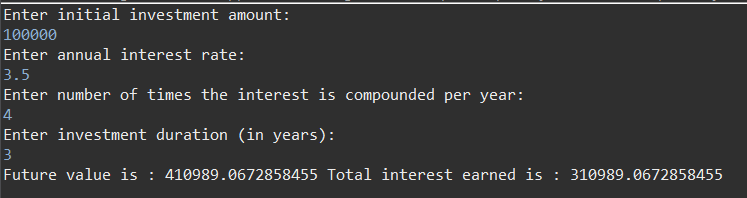
cl.getInput();

cl.calculateTotalEarn();

cl.displayOutput();

}

}



**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.

package org.example.bmi;

import java.util.Scanner;

class BMITracker {

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

float height, weight, bmi;

public void acceptRecord() {

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

weight = sc.nextLong();

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

height = sc.nextFloat();

}

public void calculateBMI() {

bmi = weight / (height \* height);

}

public void printRecord () {

String status;

if(bmi < 18.5) {

status = "Underweight";

}

else if(bmi > 18.5 && bmi < 24.9) {

status = "Normal weight";

}

else if(bmi >= 25.0 && bmi < 29.9) {

status = "Overweight";

}

else {

status = "Obese";

}

System.***out***.println("BMI status is : " + status + ".");

}

}

public class Program {

public static void main(String[] args) {

BMITracker bt = new BMITracker();

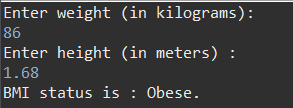
bt.acceptRecord();

bt.calculateBMI();

bt.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.

package org.example.discount;

import java.util.Scanner;

class DiscountCalculator{

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

float ogPrice, dcPerc;

float dcAmmount, finalPrice;

public void acceptRecord() {

System.***out***.println("Enter Original Ammount: ");

ogPrice = sc.nextLong();

System.***out***.println("Enter Discount Percentage: ");

dcPerc = sc.nextFloat();

}

public void calculateDiscount() {

dcAmmount = ogPrice \* (dcPerc/100);

finalPrice = ogPrice - dcAmmount;

}

public void printRecord() {

System.***out***.println("Discounted ammount is : " + dcAmmount + " Final price is : " + finalPrice);

}

}

public class Program {

public static void main(String[] args) {

DiscountCalculator cl = new DiscountCalculator();

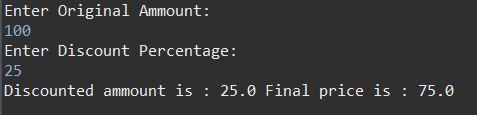
cl.acceptRecord();

cl.calculateDiscount();

cl.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.

package org.example.revevnue;

import java.util.Scanner;

class TollBoothRevenueManager{

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

float carRate, truckRate, motoRate, totalRev;

int noOfCars, noOfTrucks, noOfMoto, totalVeh;

public void setTollRates() {

System.***out***.println("Enter toll rate for cars: ");

carRate = sc.nextFloat();

System.***out***.println("Enter toll rate for truck: ");

truckRate = sc.nextFloat();

System.***out***.println("Enter toll rate for motorcycle: ");

motoRate = sc.nextFloat();

}

public void acceptRecord() {

System.***out***.println("Enter no. of cars: ");

noOfCars = sc.nextInt();

System.***out***.println("Enter no. of truck: ");

noOfTrucks = sc.nextInt();

System.***out***.println("Enter no. of motorcycle: ");

noOfMoto = sc.nextInt();

}

public void calculateRevenue() {

totalVeh = noOfCars + noOfTrucks + noOfMoto;

totalRev = (noOfCars\*carRate) + (noOfTrucks\*noOfCars) + (noOfMoto\*motoRate);

printRecord(totalVeh, totalRev);

}

public void printRecord(int totalVehPara, float totalRevPara) {

System.***out***.println("Total vehicle are : " + totalVeh + " Total revanue is : " + totalRev);

}

}

public class Program {

public static void main(String[] args) {

TollBoothRevenueManager rm = new TollBoothRevenueManager();

rm.acceptRecord();

rm.setTollRates();

rm.calculateRevenue();

}

}

