**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.

package org.test.emp;

import java.util.Scanner;

class Loan{

private double principal;

private double annualInterestRate;

private int loanTerm;

public void acceptRecord(){

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

System.***out***.print("enter the principal amount: ");

principal = sc.nextDouble();

System.***out***.print("enter the annual interest rate: ");

annualInterestRate = sc.nextDouble();

System.***out***.print("Enter the loan term (in years): ");

loanTerm = sc.nextInt();

sc.close();

}

public double calculatePayment() {

double monthIntRate = annualInterestRate / 12 /100;

int numOfMonth = loanTerm \* 12;

double monthlyPayment = principal \* (monthIntRate \* Math.*pow*(1 + monthIntRate, numOfMonth))

/ (Math.*pow*(1 + monthIntRate, numOfMonth) - 1);

return monthlyPayment;

}

public void printRecord() {

double monthlyPayment = calculatePayment();

double totalAmountPaid = monthlyPayment \* loanTerm \* 12;

System.***out***.printf("%.2f\n", monthlyPayment);

System.***out***.printf("%.2f\n", totalAmountPaid);

}

}

public class Calculator {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

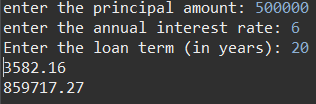
Loan l1 = new Loan();

l1.acceptRecord();

l1.printRecord();

}

}



**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.test.emp;

import java.util.Scanner;

class CompoundInterestCalculator {

private int principal;

private float annualInterestRate;

private int numberOfCompounds;

private int years;

private double totalInterest;

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

void acceptRecord () {

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

this.principal = sc.nextInt();

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

this.annualInterestRate = sc.nextFloat();

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

this.numberOfCompounds = sc.nextInt();

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

this.years = sc.nextInt();

}

void calculateFutureValue() {

double futureValue = principal \* Math.*pow*((1 + annualInterestRate / numberOfCompounds),(numberOfCompounds \* years));

this.totalInterest = futureValue - principal;

}

void printRecord () {

System.***out***.println("Total Interest is " + this.totalInterest);

}

}

public class CompoundInterest{

public static void main(String[] args) {

CompoundInterestCalculator c = new CompoundInterestCalculator();

c.acceptRecord();

c.calculateFutureValue();

c.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.

package org.test.emp;

import java.util.Scanner;

class BMI {

private float height;

private float weight;

private double bmi;

private String classification;

public void acceptRecord() {

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

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

float height = sc.nextFloat();

this.height = height;

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

float weight = sc.nextFloat();

this.weight = weight;

sc.close();

calculationBMI();

}

public void calculationBMI() {

if (height > 0) {

bmi = weight / (height \* height);

bmi = weight / (height \* height);

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 if (bmi >= 30) {

classification = "Obese";

} else {

System.***out***.println("Height must be greater than zero.");

classification = "Invalid";

}

}

}

public void printRecord() {

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

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

}

}

public class Index {

public static void main(String[] args) {

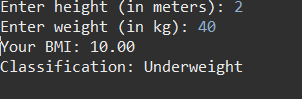
BMI tracker = new BMI();

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.

package org.test.emp;

import java.util.Scanner;

class Discount {

private double price;

private double discount;

private double discountAmount;

private double finalPrice;

public void acceptRecord() {

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

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

price = sc.nextDouble();

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

discount = sc.nextDouble();

calculateDiscount();

}

public void calculateDiscount() {

discountAmount = price \* (discount / 100);

finalPrice = price - discountAmount;

}

public void printRecord() {

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

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

}

}

public class Retailsales{

public static void main(String[] args) {

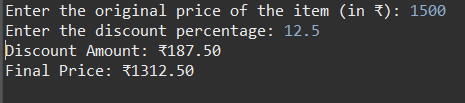
Discount discountCalculator = new Discount();

discountCalculator.acceptRecord();

discountCalculator.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.test.emp;

import java.util.Scanner;

class TollBoothRevenueManager{

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

private double carRate;

private double truckRate;

private double motorcycleRate;

private int numCars;

private int numTrucks;

private int numMotorcycles;

public void acceptRecord() {

System.***out***.print("enter the no of car passing :");

numCars = sc.nextInt();

System.***out***.print("enter the no of truck passing :");

numTrucks = sc.nextInt();

System.***out***.print("enter the no of motorcylce passing :");

numMotorcycles = sc.nextInt();

setTollRates();

}

public void setTollRates() {

System.***out***.print("enter the toll rate for car (in ruppee) :");

carRate = sc.nextDouble();

System.***out***.print("enter the toll rate for truck (in ruppee) :");

truckRate = sc.nextDouble();

System.***out***.print("enter the toll rate for motorcycle (in ruppee) :");

motorcycleRate = sc.nextDouble();

sc.close();

calculateRevenue();

}

public double calculateRevenue() {

return (numCars\*carRate)+(numTrucks\*truckRate)+(numMotorcycles\*motorcycleRate);

}

public void printRecord() {

int totalVehicales = numCars + numTrucks + numMotorcycles;

double totalRevenue = calculateRevenue();

System.***out***.println("total no of vehicles: "+ totalVehicales);

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

}

}

public class Revenue {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

TollBoothRevenueManager t1 = new TollBoothRevenueManager();

t1.acceptRecord();

t1.printRecord();

}

}

