**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.

=>

package org.assignment;

import java.util.Scanner;

class LoanAmortizationCalculator {

private float principal;

private float annualInterestRate ;

private int loanTerm;

public void acceptRecord(float principal,float annualInterestRate ,int loanTerm) {

this.principal=principal;

this.annualInterestRate=annualInterestRate ;

this.loanTerm=loanTerm;

}

public double calculateMonthlyPayment() {

float 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 void printRecord () {

System.out.println("Monthly Payment is: "+calculateMonthlyPayment());

}

}

public class Programm {

public static void main(String[] args) {

LoanAmortizationCalculator loan=new LoanAmortizationCalculator();

Scanner sc = new Scanner(System.in);

float principal;

System.out.println("Enter principal:");

principal=sc.nextFloat();

float monthlyInterestRate;

System.out.println("Enter annual Interest Rate:");

float annualInterestRate =sc.nextFloat();

int numberOfYears;

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

int loanTerm =sc.nextInt();

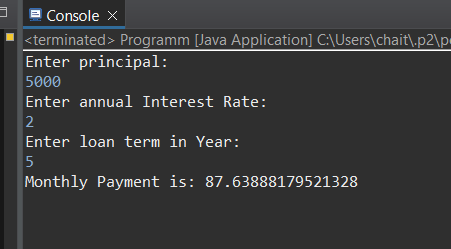
loan.acceptRecord(principal,annualInterestRate ,loanTerm);

loan.calculateMonthlyPayment();

loan.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.assignmentRetailSales;

import java.util.Scanner;

class CompoundInterestCalculator {

private float principal ;

private float annualInterestRate ;

private float numberOfCompounds;

private int years;

public void acceptRecord(float principal, float annualInterestRate, float numberOfCompounds, int years) {

this.principal=principal;

this.annualInterestRate=annualInterestRate;

this.numberOfCompounds=numberOfCompounds;

this.years=years;

}

public double calculateFutureValue () {

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

return futureValue - principal;

}

public void printRecord() {

System.out.println(this.calculateFutureValue());

}

}

public class Programm {

public static void main(String[] args) {

CompoundInterestCalculator compdIntCal=new CompoundInterestCalculator();

Scanner sc=new Scanner(System.in);

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

float principal=sc.nextFloat();

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

float annualInterestRate=sc.nextFloat();

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

float numberOfCompounds=sc.nextFloat();

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

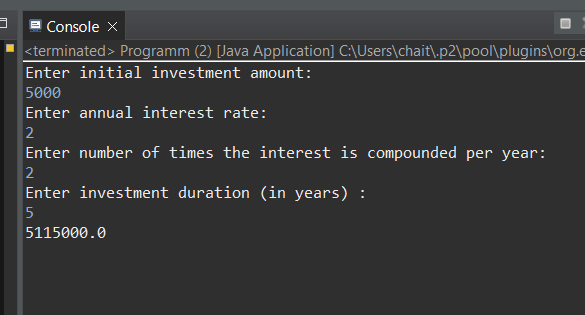
int years=sc.nextInt();

compdIntCal.acceptRecord(principal, annualInterestRate,numberOfCompounds,years);

compdIntCal.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.assignmentBMI;

import java.util.Scanner;

class BMITracker {

private float weight;

private float height;

public void acceptRecord(float weight, float height) {

this.weight = weight;

this.height = height;

}

public float calculateBMI() {

float BMI;

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

}

public String classifyBMI() {

float BMI = this.calculateBMI();

if (BMI < 18.5) {

return "Underweight: BMI less than " + 18.5;

} else if (BMI >= 18.5 && BMI < 24.9) {

return "Normal weight: BMI is between " + 18.5 + " and " + 24.9;

} else if (BMI >= 24.9 && BMI < 30) {

return "Overweight: BMI is between " + 24.9 + " and 30";

} else {

return "Obese: BMI is 30 or more";

}

}

public void printRecord() {

System.out.println(this.classifyBMI());

}

}

public class Programm {

public static void main(String[] args) {

BMITracker BMI=new BMITracker();

Scanner sc=new Scanner(System.in);

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

float weight=sc.nextFloat();

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

float height=sc.nextFloat();

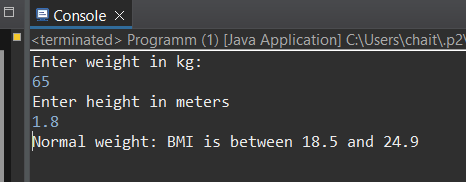
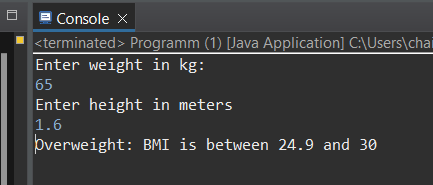
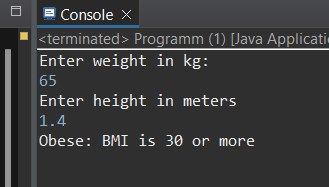
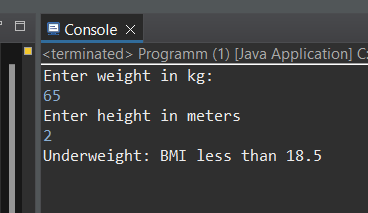
BMI.acceptRecord(weight, height);

BMI.calculateBMI();

BMI.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.assignmentInterestCalculator;

import java.util.Scanner;

class DiscountCalculator {

private float originalPrice ;

private float discountRate;

public void acceptRecord(float originalPrice, float discountRate) {

this.originalPrice = originalPrice;

this.discountRate = discountRate;

}

public float calculateDiscount () {

float finalPrice = originalPrice -( originalPrice \* (discountRate / 100));

return finalPrice;

}

public void printRecord() {

System.out.println(this.calculateDiscount());

}

}

public class Programm {

public static void main(String[] args) {

DiscountCalculator discCal=new DiscountCalculator();

Scanner sc=new Scanner(System.in);

System.out.println("Enter price: ");

float price=sc.nextFloat();

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

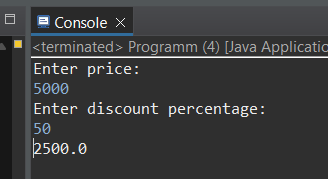
float discount=sc.nextFloat();

discCal.acceptRecord(price, discount);

discCal.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.assignmentTollBooth;

import java.util.Scanner;

class TollBoothRevenueManager {

private String vehicleTypes;

private int numberOfVehicles;

public void acceptRecord() {

Scanner sc = new Scanner(System.in);

System.out.println("Press number to select type of vehicle: \n1.Car 2.Truck 3.Mororcycle ");

int typeNo = sc.nextInt();

switch (typeNo) {

case 1:

this.vehicleTypes = "Car";

break;

case 2:

this.vehicleTypes = "Truck";

break;

case 3:

this.vehicleTypes = "Motorcycle";

break;

}

System.out.println("Enter number of vehicles:");

this.numberOfVehicles=sc.nextInt();

}

public double setTollRates() {

double tollRate=0.0;

String type=this.vehicleTypes;

switch (type) {

case "Car":

tollRate = 50.00;

break;

case "Truck":

tollRate = 100.00;

break;

case "Motorcycle":

tollRate = 30.00;

break;

default:

System.out.println("Something went wrong!");

}

return tollRate;

}

public double calculateRevenue() {

return (this.numberOfVehicles\* setTollRates());

}

public void printRecord() {

System.out.println("Your vehicle type: "+this.vehicleTypes+"\nNumber of vehicle: "+this.numberOfVehicles+"\nTotal Revenue:"+calculateRevenue());

}

}

public class Programm {

public static void main(String[] args) {

TollBoothRevenueManager tollBooth = new TollBoothRevenueManager();

tollBooth.acceptRecord();

tollBooth.setTollRates();

tollBooth.printRecord();

}

}

