Ketaki Chhayendra Mahajan

**Batch:** P1-2

**Roll Number:** 16010622050

**Experiment / assignment / tutorial No. 1**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE:**  Write a program for:   1. Program to find area and circumference of various geometric shapes. 2. Program to calculate EMI (Equated Monthly Instalment) of loan amount if principal, rate of interest and time in years is given by the user.   (E = (P \* r \* (1+r) n) / ((1+r) n – 1) |

**Aim:**

Write a program for:

1. Program to find area and circumference of various geometric shapes.
2. Program to calculate EMI (Equated Monthly Instalment) of loan amount if principal, rate of interest and time in years is given by the user.

E = (P \* r \* (1+r) n) / ((1+r) n – 1)

**Expected OUTCOME of experiment:**

The objective of the first program is to calculate the area and perimeter of various kinds of shapes including circle, square, rectangle and triangle. Therefore, the expected outcome is to input lengths from the user and print the correct values of area and perimeter for each respective geometric shape and display it onto the screen.

The next program expects user to enter the loan amount, rate of interest and time (in years) to calculate EMI. Therefore, the expected outcome is to print the correct value of EMI using the formula given and display it onto the screen.

Resolving both the problems stated in the experiment will also help clear basic concepts of C programming (further discussed in conclusion).

**Books/ Journals/ Websites referred:**

1. Programming in ANSI C, E. Balagurusamy, 7th Edition, 2016, McGraw-Hill Education, India.
2. Structured Programming Approach, Pradeep Dey and Manas Ghosh, 1st Edition, 2016, Oxford University Press, India.
3. Let Us C, Yashwant Kanetkar, 15th Edition, 2016, BPB Publications, India.

**Problem Definition:**

**Problem 1:** Area and circumference of any shape **will be given by instructor** (e.g. circle). Ask the user to enter the value of the radius of a circle. Put the values in the formula for finding area of a circle and circumference of a circle and print the outcome for area of a circle and circumference of a circle

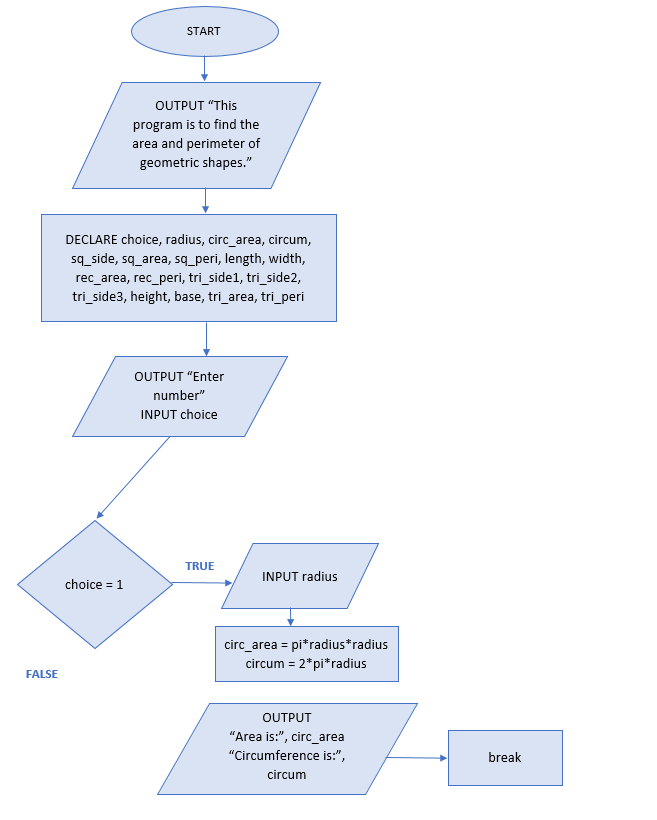
**Problem 2:** Calculating EMI

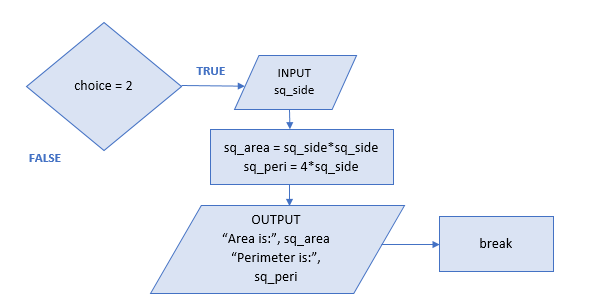
Ask the user to enter the value of principal amount, rate of interest and time (in years). Store the value in E and print the final monthly instalment E as an outcome.

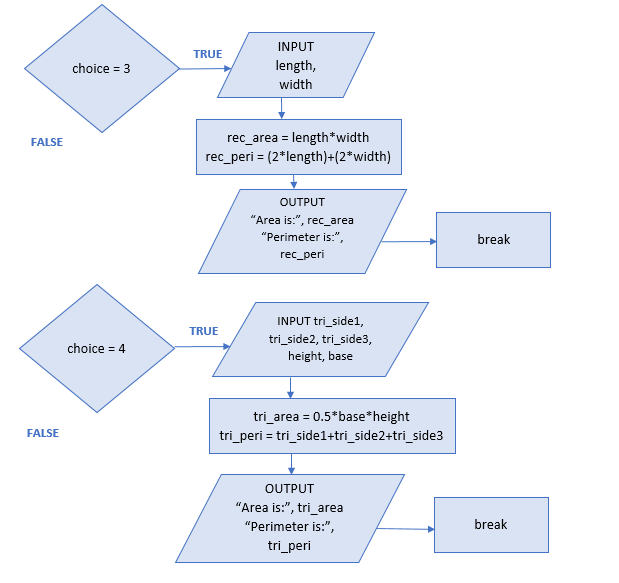
Formula to be used: E = (P \* r \* (1+r) n) / ((1+r) n – 1)

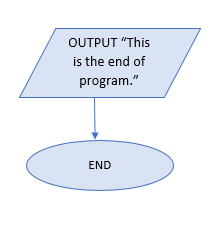
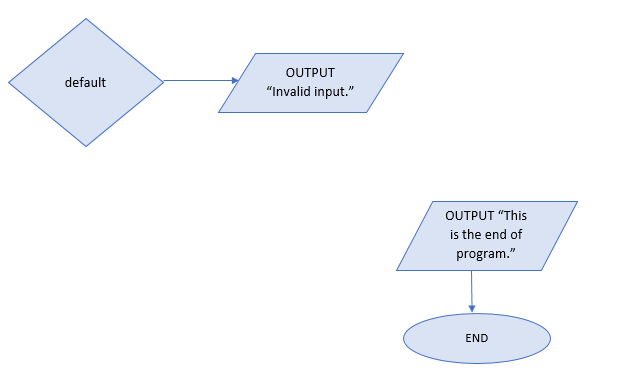
**Flowchart:**

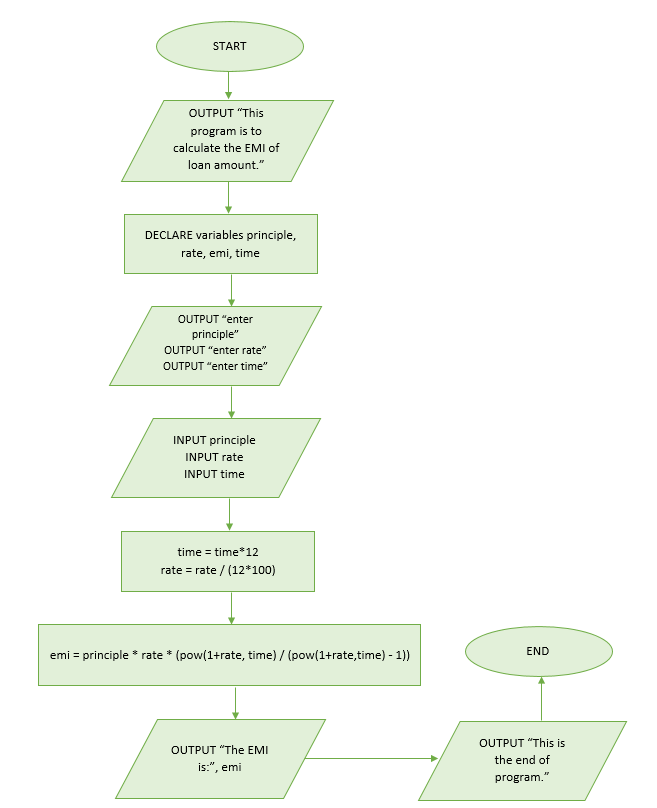
1. **Flowchart for calculating area and circumference of various shapes –**









1. F**lowchart for calculating EMI –**

**Implementation details:**

1. **Implementation for calculating area and perimeter –**

// Experiment 1 - program to find area and circumference of various geometric shapes

#include<stdio.h>

#define pi 3.14

main()

{

printf(" This program is to find the area and perimeter of various shapes such as rectangle, square, triangle and circle.");

int choice;

float radius, circ\_area, circum, sq\_side, sq\_area, sq\_peri, length, width, rec\_area, rec\_peri, tri\_side1, tri\_side2, tri\_side3, height, base, tri\_area, tri\_peri;

printf("\n\n Choose a number from 1 to 4 to find their respective areas and perimeter. \n 1: circle\n 2: square\n 3: rectangle\n 4: triangle");

printf("\n\n Enter number: ");

scanf("%d", &choice);

switch(choice)

{

case 1:

printf("\n You have chosen circle.");

printf("\n Enter the radius of circle: ");

scanf("%f", &radius);

circ\_area = pi\*radius\*radius;

circum = 2\*pi\*radius;

printf(" The area of circle is: %f", circ\_area);

printf("\n The perimeter of circle is: %f", circum);

break;

case 2:

printf("\n You have chosen square.");

printf("\n Enter the side of square: ");

scanf("%f", &sq\_side);

sq\_area = sq\_side\*sq\_side;

sq\_peri = 4\*sq\_side;

printf(" The area of square is: %f", sq\_area);

printf("\n The perimeter of square is: %f", sq\_peri);

break;

case 3:

printf("\n You have chosen rectangle.");

printf("\n Enter the length of rectangle: ");

scanf("%f", &length);

printf(" Enter the width of rectangle: ");

scanf("%f", &width);

rec\_area = length\*width;

rec\_peri = (2\*length)+(2\*width);

printf(" The area of rectangle is: %f", rec\_area);

printf("\n The perimeter of rectangle is: %f", rec\_peri);

break;

case 4:

printf("\n You have chosen triangle.");

printf("\n Enter the lengths of sides of triangle: \n ");

scanf("%f %f %f", &tri\_side1, &tri\_side2, &tri\_side3);

printf(" Enter the height of triangle: ");

scanf("%f", &height);

printf(" Enter the base of triangle: ");

scanf("%f", &base);

tri\_area = 0.5\*base\*height;

tri\_peri = tri\_side1+tri\_side2+tri\_side3;

printf(" The area of triangle is: %f", tri\_area);

printf("\n The perimeter of triangle is: %f", tri\_peri);

break;

default:

printf(" Invalid input");

}

printf("\n\n This is the end of program.");

}

1. **Implementation for calculating EMI -**

//Experiment 1 - program to find the EMI

#include<stdio.h>

#include <math.h>

main()

{

printf(" This program is to calculate the EMI of loan amount.");

float principle, rate, emi, time;

printf("\n\n Enter principle amount: ");

scanf("%f", &principle);

printf(" Enter the rate of interest: ");

scanf("%f", &rate);

printf(" Enter the number of years: ");

scanf("%f", &time);

time = time\*12; //formula uses time in months, converting time in years from user to time in months

rate = rate/(12\*100); //formula uses monthly rate, converting yearly rate from user to monthly period

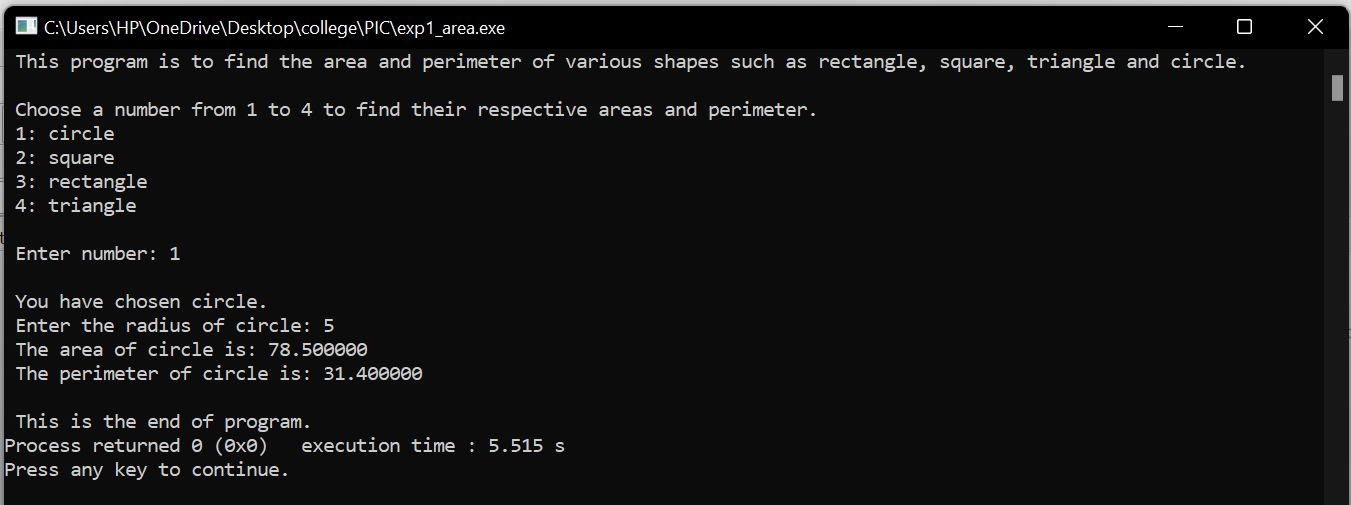
emi = principle \* rate \* (pow(1+rate,time) / (pow(1+rate,time) - 1));

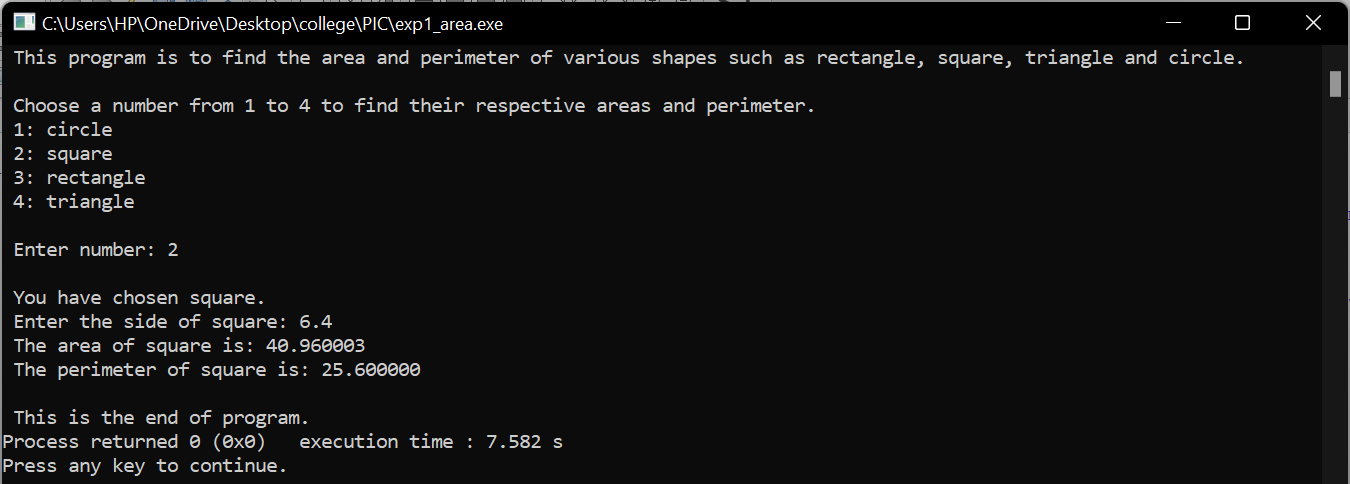
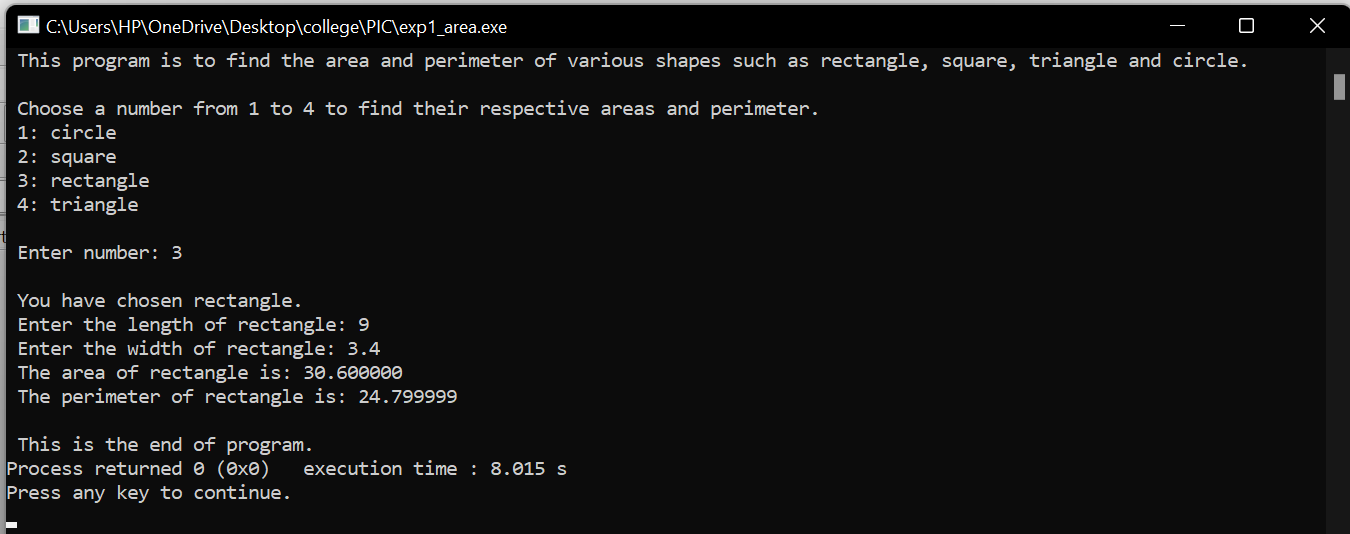
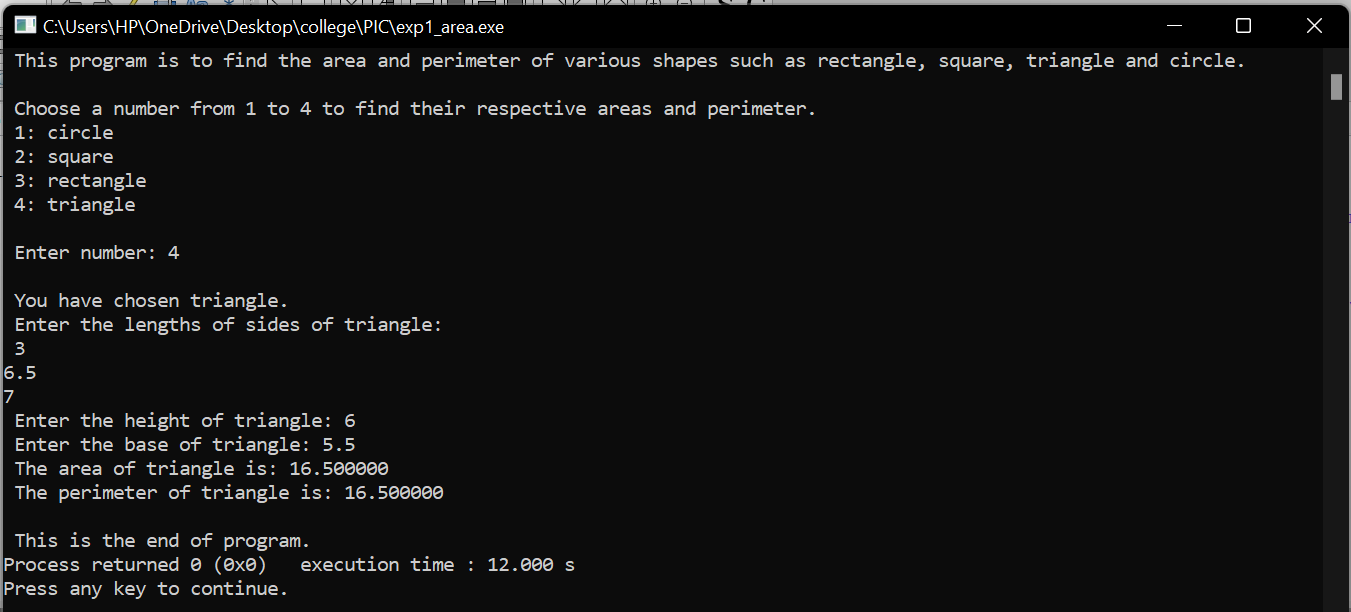
printf("\n Equated Monthly Installment (EMI) of loan amount when principal is %f, rate of interest is %f in %f years is: %f", principle, rate, time, emi);

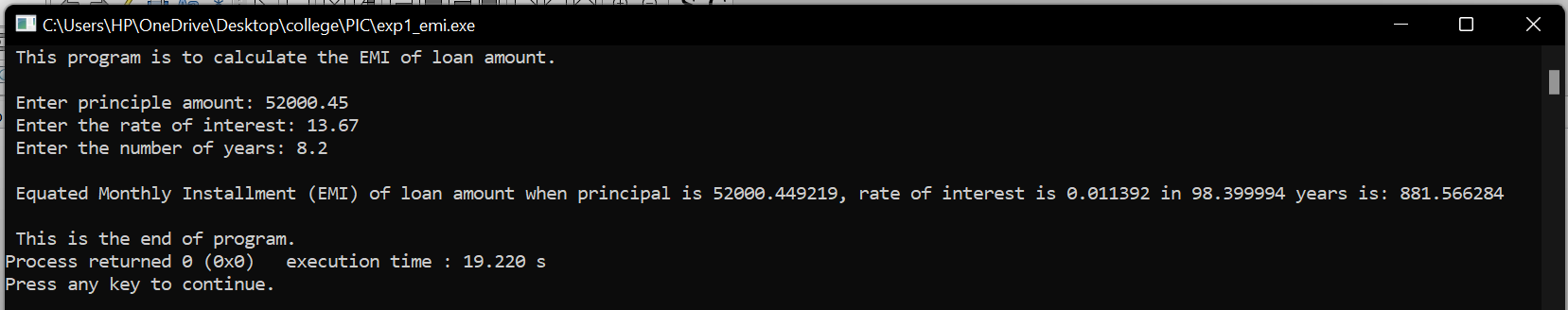
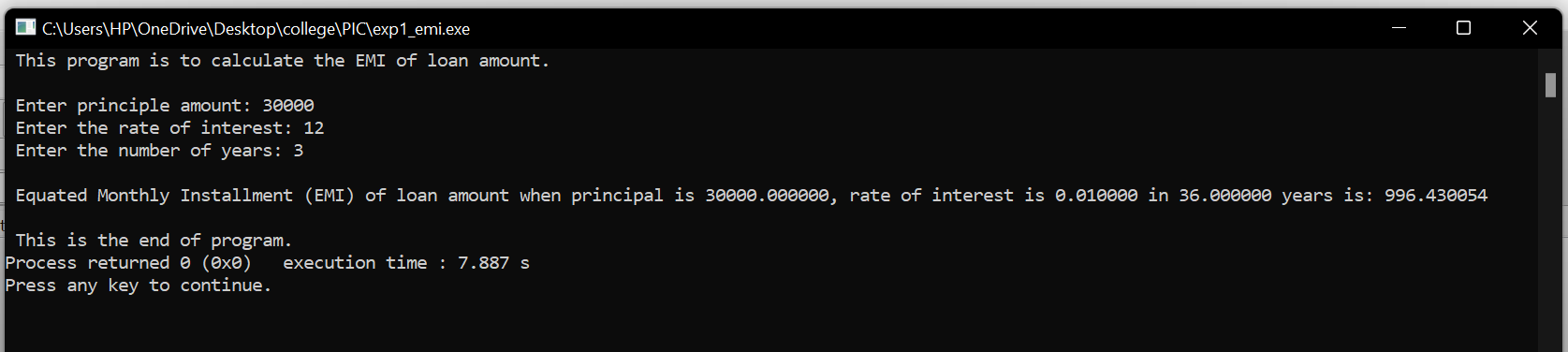
printf("\n\n This is the end of program.");

}

**Output(s):**

1. **Problem 1 –**



1. **Problem 2 – **

**Conclusion:**

Experiment 1 was successfully completed by resolving both the problems. Although the solution to the problems were important, the process of figuring out, and applying all my knowledge gained in programming, was much more significant due to the numerous learning outcomes.

In the first problem, where area and perimeter of various shapes had to be displayed, I defined the constant, pi, meaning the same identifier will be used throughout the program, making it easier to read, debug and it keeps the program code consistent. I decided to experiment and use the control statement **switch-case-default**. In this, the user chose the shape by entering a number and to display a particular shapes area and perimeter instead of displaying values for all shapes. One problem I faced during execution was, all cases would print and be displayed onto the screen instead of just one case (chosen by user). Referring to the reference books helped me realize and learn about the **break** statement. When a break statement is encountered inside a loop, the loop is immediately terminated and the program resumes at the next statement following the loop. Hence, it can also be used to terminate a case in the switch statement. One thing I would like to change if I were to write this program again would be to use a loop system for the switch case so that the program doesn’t end after one choice and keeps asking for a number until terminated.

In the second problem, while performing the exponential operation to calculate EMI using its formula, I faced few errors and was getting all sorts of garbage values. The reason was, I was using “^” instead of **pow()** function as required in C language. In order for pow() to work correctly, **#include <math.h>** pre-processor directive was needed.

Although both the problems and their aims were different, I was able to learn a lot of the basics of C programming and programming in general, this includes –

1. Types of different variables in C program such integer variable, character variable etc.
2. Naming different variables correctly according to its rules and aptly to its purpose.
3. Comments in C program used to clarify purpose of a program or statements.
4. Using different data types such as int, float, char, double, void accordingly to the given problem and its purpose.
5. Using printf() function to display the calculated values, for which it is necessary to use #include <stdio.h> at the beginning of the program and the use of format specifiers such as %d, %f, %c to tell the compiler about the type of data to be printed on screen.
6. Compilation and execution.
7. Receiving inputs through user is achieved using a function called **scanf()**, for which it is necessary to use ampersand (&), also known as ‘address of’ operator.

To conclude, resolving experiment 1 has given me a better understanding of the basics and solidified my foundation in C programming.

**Post lab descriptive questions:**

1. **What are the basic data types in C?**

C language uses different types of variables (defined with different keywords) for storing data values. These must be specified by data type. The data type specifies the size and the type of information the variable will store. Few of the basic data types include int (integer), float (floating point), char (character), double (double) and void. These basic data types can be further categorized using 4 data type modifiers such as signed, unsigned, long and short.

A short description of the basic data types can be seen in the following table.

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Description** | **Size** |
| int | Stores whole numbers including positive, negative and zero values, any decimals are ignored. | 2 or 4 bytes |
| float | Stores fractional numbers containing one or more decimals (up till 7 decimal places). | 4 bytes |
| char | Stores a single bit of characters such as letters(A-Z or a-z), 0-9 digits and ASCII value (should not exceed 127). | 1 byte |
| double | Stores fractional numbers containing one or more decimals (up till 15 decimal places). | 8 bytes |
| void | Void has no value and this data type is used when we define functions or when a function does not return anything. | 0 bytes |

1. **What is a flowchart? What are the standard symbols used to draw a flowchart? Explain in brief.**

When designing a solution to a problem, the solution can be expressed (the algorithm) in various different forms. One of the ways to illustrate an algorithm is through flowcharts. A flowchart is a diagram with shapes linked together to represent the sequential steps of an algorithm. The flowchart presents the workflow or process in a step-by-step approach to solving a problem.

Few of the standard symbols used to draw a flow chart include –

|  |  |  |
| --- | --- | --- |
| **Name** | **Shape** | **Type of process** |
| Flow line |  | Shows the flow/direction from one step to the next in the logic flow of the problem. |
| Process |  | Represents an operation or set of operations that change the value, form, or location of data. |
| Input/output |  | Indicates the process of inputting and outputting data. |
| Decision | FALSE  TRUE  CONDITION | Shows a conditional operation that will determine which one of two paths the program will take. |
| Start/end |  | Marks the beginning and end of the entire process. |
| Pre-defined process |  | Indicates a named process that is defined elsewhere. |

The shape of the symbol represents the type of process that the symbol contains. Arrows are used to show the flow of execution, meaning that flowcharts can represent all the core concepts of programming, namely sequence, selection, and iteration.

A flowchart is helpful for understanding how a process is executed, study a process for improvement and document the process for reference and in order to communicate with others.

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_ Signature of faculty in-charge**