MINI PROJECT

Student Name: LOVEPREET SINGH **UID: 24BCA10245**
Branch: UIC **Section/Group: 24BCA-5(B)**
Semester: 1
Subject Name: Computer Programming **Subject Code: 24CAH-101**

Topic:

Case Study on Developing a Simple Calculator Program in C

INTRODUCTION

Background: C programming is widely known for its simplicity and

efficiency in managing computational tasks and mathematical operations. As a foundational programming language, C provides robust support for arithmetic operations, making it an ideal choice for building basic calculators. This case study focuses on leveraging C programming's features—such as control structures, input/output functions, and switch cases—to develop a simple calculator that performs fundamental arithmetic calculations efficiently.

Purpose/Objective: The primary objective of this project is to design and implement a basic calculator program in C that can perform essential operations, including addition, subtraction, multiplication, and division. By creating this calculator, the project aims to deepen understanding of C programming fundamentals, such as variable handling, control structures, and user input processing.

TASKS TO BE DONE

Input Two Numbers and Select an Arithmetic Operation

The program will prompt the user to enter two numbers on which they want to perform a calculation.
After entering the numbers, the user will be presented with a menu of arithmetic operations (addition, subtraction, multiplication, and division).
The user will select one of these operations to apply to the entered numbers.

Display the Result of the Chosen Operation

Once the user selects an operation, the program will perform the

corresponding calculation on the two numbers.
The program will then display the result in a clear format, showing both the operation and the outcome.
For example, if the user inputs 5 and 3 and selects addition, the program will display: `5 + 3 = 8`.

Allow the User to Perform Multiple Operations Until They Choose to Exit

After displaying the result, the program will ask the user if they want to perform another calculation.
If the user selects "Yes," the program will start over, allowing them to input new numbers and select another operation.
If the user chooses "No," the program will terminate and display a farewell message.
This looping structure enables the user to perform multiple calculations in a single session without restarting the program.

CU CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University
NAAC GRADE A+ ACCREDITED UNIVERSITY

STEPS FOLLOWED IN MAKING THE PROJECT

Define the Program Objective

The first step involved understanding the program's purpose: creating a simple calculator capable of performing basic arithmetic operations (addition, subtraction, multiplication, and division).
This clarified the program requirements and guided the subsequent development steps.

Set Up the Development Environment

Installed and configured a C programming environment. This included setting up a code editor or IDE (such as Visual Studio Code or Code :: Blocks) to write, compile, and test the C code.

Define the Program Structure

Planned the structure of the code by identifying the primary functions and control structures needed.
Decided to use a `switch` statement to handle different arithmetic operations and a loop to allow multiple calculations within a single program run.

Write the Menu Display Function

Created a function or code block to display the menu to the user, listing all available operations (addition, subtraction, multiplication, and division).
Ensured the menu was clear and included instructions on how to make a selection.

Implement User Input for Operation and Numbers

Wrote code to capture user input for both the desired operation and the numbers to be calculated.
Included prompts to guide the user in selecting an operation and entering two numerical values for calculation.

Program the Arithmetic Operations

Added code to perform each arithmetic operation based on the user's selection.

CU CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University
NAAC GRADE A+ ACCREDITED UNIVERSITY

Used a `switch` statement to execute the correct calculation and stored the result in a variable.

Add Error Handling for Division

Included an error-handling condition to prevent division by zero, which would cause a runtime error.
Displayed an error message if the user attempted to divide by zero and prompted them to enter a different number.

Display the Result

Implemented a step to display the result of the calculation to the user.
Formatted the output to include both the operation performed and the calculated result for clarity.

Add Loop for Multiple Calculations

Added a loop that asks the user if they want to perform another calculation.
If the user chose to continue, the program restarted the menu; if they chose to exit, the program terminated.

Test and Debug the Program

Ran the program multiple times to ensure that each arithmetic operation produced the correct result.
Tested edge cases, such as dividing by zero, to verify that error handling worked as expected.

Finalize and Document the Code

Added comments to the code to document each section and explain its purpose.
Organized and formatted the code for readability and ease of maintenance.

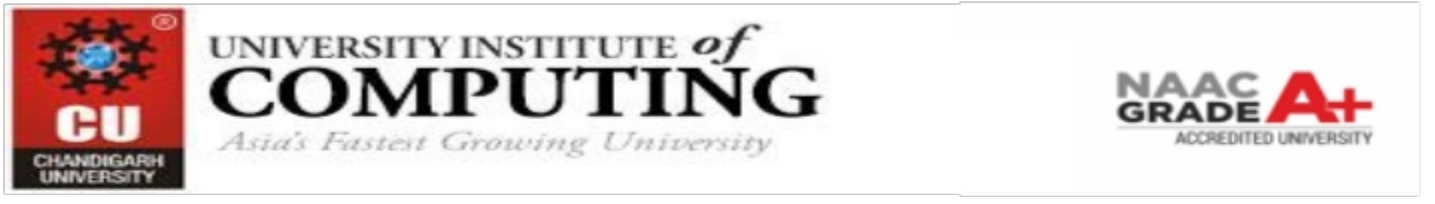
CODE :

```
#include <stdio.h> int main() {
char operator; double
num1, num2, result;

   // Display options to the user printf("Simple Calculator\n");
printf("Choose an operation (+, -, *, /): "); scanf(" %c", &operator); // Notice
the space before %c to avoid newline issues

   // Take two numbers as input
printf("Enter first number: ");
scanf("%lf", &num1); printf("Enter
second number: "); scanf("%lf",
&num2);
```

```
   // Perform the chosen operation
switch (operator) {
        case '+':
            result = num1 + num2;            printf("%.2lf + %.2lf =
%.2lf\n", num1, num2, result);
            break;
        case '-':
```




```
            result = num1 - num2;            printf("%.2lf - %.2lf =
%.2lf\n", num1, num2, result);
            break;
case '*':
            result = num1 * num2;            printf("%.2lf * %.2lf =
%.2lf\n", num1, num2, result);
            break;
case '/':
            // Check for division by zero            if (num2 != 0) {
result = num1 / num2;                printf("%.2lf / %.2lf =
%.2lf\n", num1, num2, result);
            } else {
                printf("Error: Division by zero is not allowed.\n");
            }
            break;
default:
            printf("Invalid operator. Please choose +, -, *, or /.\n");
    }
```

```
    return 0;
}
```

SUMMARY

The Simple Calculator Program in C project aimed to create a basic yet functional calculator that allows users to perform fundamental arithmetic operations: addition, subtraction, multiplication, and division. Using structures like loops, conditional statements, and error handling, this project demonstrates foundational C programming concepts and skills.

The program is designed to prompt the user to input two numbers and then select an arithmetic operation to perform on these numbers. Based on the user's choice, the program calculates and displays the result. The calculator program also includes a loop that enables users to perform multiple calculations in one session, only ending when the user chooses to exit. Additionally, error handling was incorporated to prevent division by zero, ensuring reliable performance.

This project provided hands-on experience with essential C programming features, such as data input and output, decision-making structures, and loops. By implementing these components, the project highlights the practical applications of C programming in developing interactive and user-friendly applications.

RESULT

The Simple Calculator Program successfully performs basic arithmetic operations based on user input. After entering two numbers and selecting an operation, the program calculates and displays the correct result. Additionally, the program's loop structure allows the user to perform multiple calculations in a single session, enhancing its usability. Error handling for division by zero ensures stability and provides helpful feedback to the user.

Overall, this project demonstrates a functional, user-friendly calculator, showcasing core C programming concepts like control structures, loops, and error checking.

LEARNING OUTCOME

- **Programming Fundamentals**: Gained hands-on experience with basic C programming concepts such as variables, control structures, and functions.
- **Problem-Solving Skills**: Developed the ability to break down a problem into manageable parts and create algorithms to solve it.
- **User Interaction**: Learned how to implement user input and output effectively to create a functional interface.
- **Error Handling**: Understood the importance of input validation and managing errors, such as handling division by zero.
- **Code Organization**: Enhanced skills in writing organized and well-documented code for better readability and maintenance.

Evaluation Grid:
