Project 1: Calculator in C

Program Overview

The calculator program will:

1. Prompt the user to enter two numbers.

2. Ask the user to select an operation (e.g., +, -, *, /).

3. Perform the requested calculation and display the result.

4. Handle basic error cases (e.g., division by zero).

5. Optionally, allow the user to perform multiple calculations in a loop.

Design Considerations

• User-Friendly: Clear prompts and outputs for ease of use.

• Error Handling: Prevent division by zero errors.

• Simplicity: Focus on required skills without unnecessary complexity.

• Flexibility: Design should be easy to extend (e.g., adding operations).

• Input Validation: Ensure the operation is one of the four allowed symbols.

Program Structure

1. Main Components

• Variables: Two floating-point numbers, an operation character, and a result variable.

• Input Collection: Prompt and read two numbers and an operation.

• Operation Logic: Use conditionals to process the chosen operation.

• Output: Display the result clearly.

2. Flow of Execution

1. Show a welcome message.

2. Collect the first number.

3. Collect the second number.

4. Collect the operation.

5. Calculate and display the result.

6. (Optional) Ask if the user wants to repeat.

---

3. Error Handling

• Check for division by zero before division.

• Validate the operation input.

---

4. Potential Enhancements

• Loop to allow multiple calculations.

• Clear input buffer for smooth operation.

Sample Output

Welcome to the Calculator Program!

Enter first number: 5.5

Enter second number: 2.0

Enter operation **(**+, -, *, /**)**: +

5.50 + 2.00 = 7.50

Would you like to calculate again? **(**y/n**)**: y

Enter first number: 10

Enter second number: 0

Enter operation **(**+, -, *, /**)**: /

Error: Division by zero is not allowed!

Would you like to calculate again? **(**y/n**)**: n

Thank you for using the Calculator Program. Goodbye!

Potential Improvements

• Validate number inputs (e.g., reject letters).

• Add more operations like exponentiation or modulus.

• Use a numbered menu instead of symbols.

• Track and display calculation history.

Skills Demonstrated

• Input/Output: Handling user input and formatted output.

• Conditionals: Managing operation choices and errors.

• Arithmetic: Performing basic calculations with appropriate types.