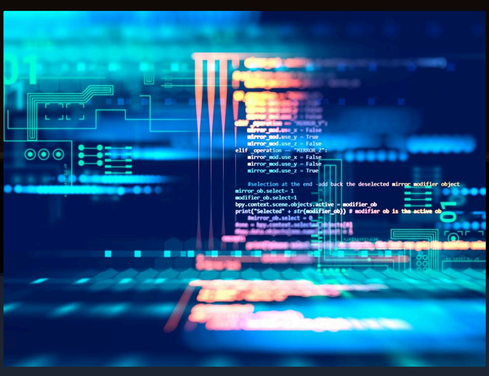

Simple Command-Line Calculator in C

1 Abstract

This project presents a simple command-line calculator program developed in C that performs basic arithmetic operations, including addition, subtraction, multiplication, and division. The main aim is to provide users with a straightforward tool for performing these calculations based on inputted numbers and operators.

2 Core Points

The program is designed to handle common issues such as invalid operator input and division by zero errors, enhancing its robustness. By developing this calculator, users can efficiently execute essential arithmetic operations while receiving error messages for invalid inputs.

3 Aim

To develop a simple command-line calculator in C that can perform basic arithmetic operations such as addition, subtraction, multiplication, and division.

4 Objectives

- User Input Handling: Allow users to input two numbers and an operator to perform the desired calculation. - Basic Operations: Implement functions for addition, subtraction, multiplication, and division - Error Handling: Handle errors gracefully, such as division by zero and invalid operator inputs. - Continuous Operation: Allow the user to perform multiple calculations without restarting the program. - User-Friendly Interface: Provide clear prompts and output results in a readable format - To perform basic arithmetic operations (+, - /) based on the chosen operator.

Features

- Basic Arithmetic Functions: Functions for addition, subtraction, multiplication, and division. - User Interaction: Simple console input and output for user engagement. - Loop for Continuous Calculation: A loop that allows repeated calculations until the user chooses to exit. - Error Messages: Display error messages for invalid inputs or operations (e.g., division by zero). - Clear Code Structure: Modular code with separate functions for different operations to enhance readability and maintainability.

Observations

- Ease of Use: A command-line interface makes it straightforward for users to perform calculations. - Modularity: The use of functions helps in organizing code logically, making it easier to debug and extend. - Scalability: The basic framework can be expanded to include more complex operations (e.g., exponentiation, square roots) or a graphical user interface. - Error Handling Importance: Proper error handling is crucial to ensure the program runs smoothly and provides meaningful feedback to users. - Learning Opportunity: This project provides a hands-on opportunity to learn about basic programming constructs, such as loops, conditionals, and functions in C. - The program prompts the user to enter an operator and two numbers.

Outcome

- Depending on the operator entered, it calculates the result of the operation: - If + is entered, it adds numl and num2 o If- is entered, it subtracts num2 from numl o If' is entered, it multiplies numl and num2 o If / is entered, it divides numl by num2 unless num2 is zero (in which case, it displays an error message). - If an incorrect operator is entered, the program displays an error message indicating invalid operator input. - The program successfully performs the calculation and displays the result or error message.

PROCEDURE CODE:

#include

int main char operator, double numl, num2, result;

// Display the menu printf 'Enter an operator(+,- *J): scanf (' O/d',&operator);

// Ask for numbers print.' Entertwo numbers: scanf ('0/01f 0 /olfl &num2);

// Perform calculation based on the operator switch(operator) { case '+' result = numl + num2; printf "0/0.21f + 0/0.21f = 0/0.21f\n", numl , num2, result); break;

0/0.21f\n", numl, num2, result);

/0.21f* /0.21f = 0/0.21fin", numl, num2, result);

result = numl / num2; printf "0/0.21f / 0/0.21f = 0/0.21fin" numl, num2, result); else printf "Error! Division by zero.\n'

break; default printf "Error! Operator is not correct.\n'

return 0;

Output:

main.c

Run

Output

Clear

```c
#include <stdio.h>

int main() {
    char operator;
    double num1, num2, result;

    // Display the menu
    printf("Enter an operator (+, -, *, /): ");
    scanf(" %c", &operator);

    // Ask for numbers
    printf("Enter two numbers: ");
    scanf("%lf %lf", &num1, &num2);

    // Perform calculation based on the
        operator
    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 '/':
            if (num2 != 0) {
                result = num1 / num2;
                printf("%.2lf / %.2lf = %
                    .2lf\n", num1, num2, result
                    );
            } else {
                printf("Error! Division by zero
                    .\n");
            }
            break;
        default:
            printf("Error! Operator is not
                correct.\n");
    }

    return 0;
}
```

```
/tmp/5bVx9HzdEG.o
Enter an operator (+, -, *, /): +
Enter two numbers: 10 5
10.00 + 5.00 = 15.00


=== Code Execution Successful ===
```

by bhekumuzi