Simple Calculator in C

This project report details the development of a basic calculator program using C. The calculator performs fundamental arithmetic operations and incorporates error handling for robustness. This project serves as an excellent resource for learning basic programming concepts in C.

Program Overview

Functionality

The program allows users to perform addition, subtraction, multiplication, and division.

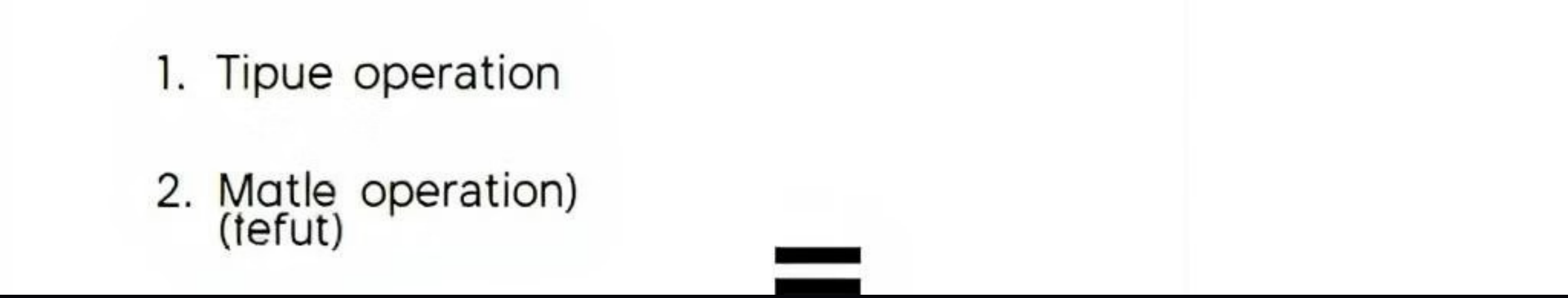
Error Handling

It includes error-handling techniques to manage invalid operator inputs and division by zero errors.

Data Structures

Utilizes essential data structures and control flow mechanisms for efficient processing.

The calculator efficiently handles inputs, processes calculations, and delivers accurate results, making it user-friendly.



Problem Statement & Algorithm

1 Input Handling

Prompt the user to enter an arithmetic operator and two numerical operands.

2 Processing

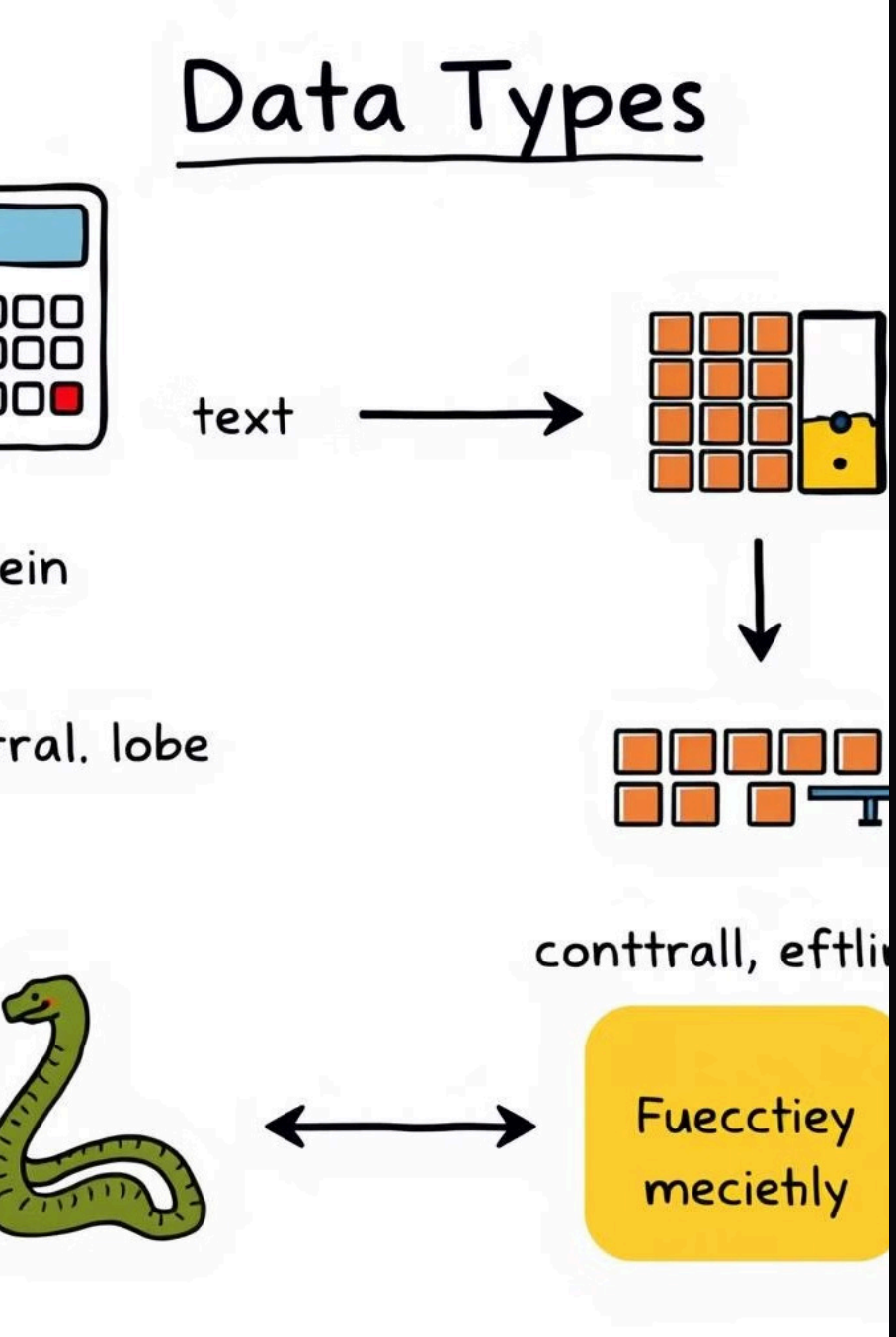
Use a switch statement to determine the operation based on the operator input.

3 Output Result

Display the result of the arithmetic operation.

4 Error Handling

Display an error message for invalid operators and prevent division by zero.



Key Concepts

Data Types

char: Stores the operator input. **double**: Allows decimal calculations.

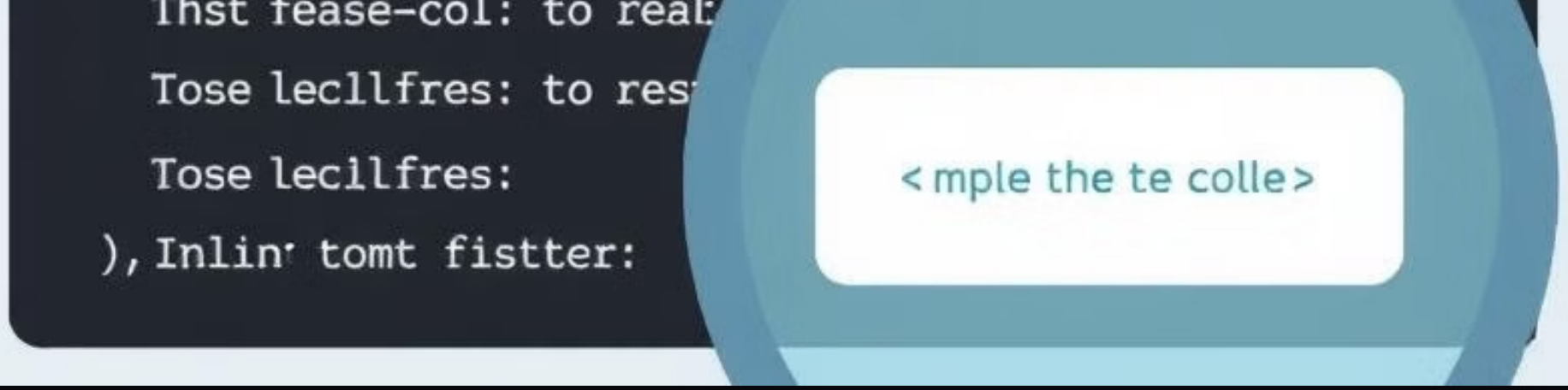
Control Structures

switch: Handles multiple operator checks. **if**: Manages division by zero.

Input/Output

scanf: Accepts user inputs. **printf**: Displays results and errors.

The program uses **char** and **double** data types, along with **switch** and **if** control structures, to efficiently handle user input, process operations, and display results.



Advantages of the Program

Simplicity

Straightforward and easy to understand for beginners.

Precision

Uses **double** to ensure accurate decimal calculations.

Robustness

Includes error handling for invalid inputs and division by zero.

The program's simplicity, precision, and robustness make it an excellent learning tool. It can also be extended to support additional operators.

Potential Improvements

1

Enhanced Input Validation

Ensure users input valid operators and numeric operands.

2

Additional Operators

Include operators like modulus (%) and exponentiation (^).

3

Modular Design

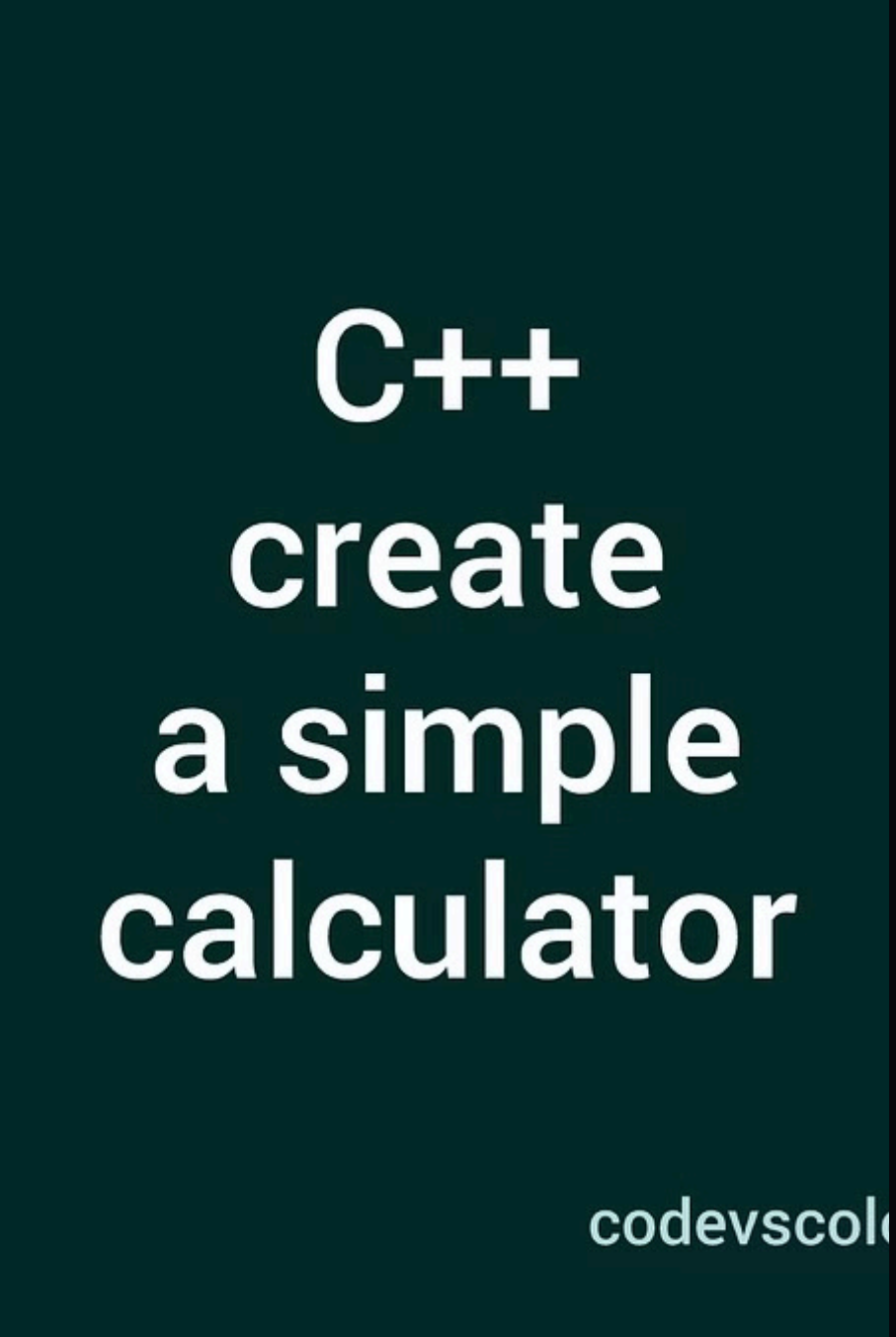
Use functions for each arithmetic operation.

4

Interactive Experience

Allow multiple calculations without restarting.

The program can be improved by adding input validation, supporting more operators, using a modular design, and providing a more interactive user experience.



Sample Input and Output

Here's an example of how the program interacts with the user:

This demonstrates the program's ability to take user input, perform calculations, and display the result or an error message.

Conclusion

This project successfully implements a basic calculator in C, demonstrating the use of control structures, data types, and error handling. With further enhancements, it can evolve into a comprehensive calculator application. Its simplicity and versatility make it a valuable resource for learning basic programming concepts in C.