মেট্রোপলিটন ইউনিভার্সিটি
Metropolitan University

PROJECT REPORT

Course Name: Data Structure Lab
Course Code: CSE 134

Submitted to:

Rishad Amin Pulok
Lecturer
Department of Computer Science & Engineering
Metropolitan University, Sylhet.

Submitted by:

Fahima Akter (232-115-028)
Nadia Muntaha (232-115-031)
Jannatul Ferdaus (232-115-040)
Batch: 59
Section: A

Department of Computer Science & Engineering
Metropolitan University, Sylhet.

Date of submission: 13.11.2024

PROJECT REPORT

CalcMate(Simple Calculator)

1.Introduction

The Simple Calculator project is a console-based application written in C. It provides a user-friendly interface for performing basic arithmetic operations. This project is designed to help beginners understand key programming concepts such as user input/output, control flow, functions, and error handling.

2.Why I Choose This Project?

The primary goal behind creating a simple calculator application in C was to understand and implement basic concepts of programming such as user input handling, conditional logic, function creation, and error management. This project serves as a practical exercise in utilizing the `switch` statement, `while` loop, mathematical functions (like `pow`, division, modulus), and handling invalid inputs effectively in C.

Additionally, it allowed the exploration of edge cases like division by zero and modulus operations, displaying appropriate error messages for these situations. This project provides a hands-on way to solve basic problems with math operations, laying the foundation for more advanced C programming

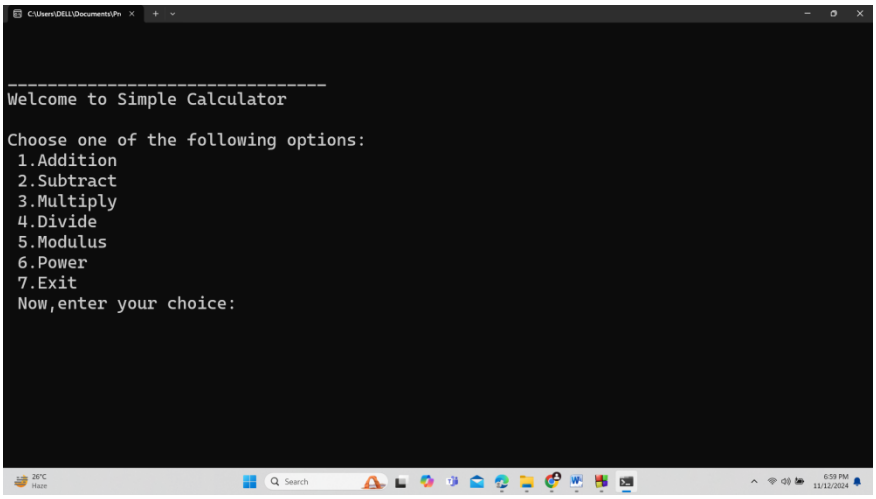
3. Features

The calculator program supports the following functionalities: Addition, Subtraction, Multiplication, Division, Modulus, Power & Exit

All Output Explanation:

When the user runs the program, the following outputs are expected:

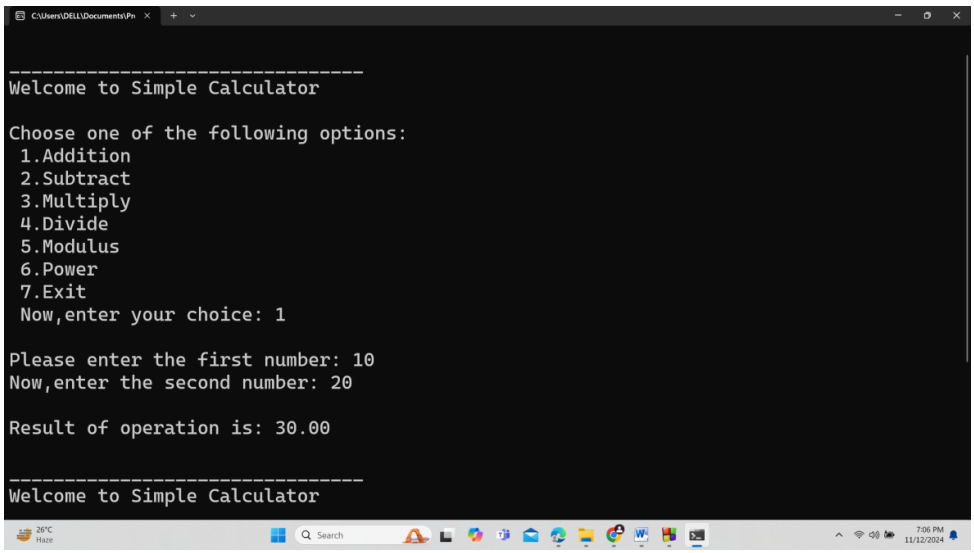
- **Menu Display**: The program first prints a menu with various arithmetic operations (addition, subtraction, multiplication, division, modulus, power) and an option to exit.



i)ADDITION (Option 1):

Prompts the users for two numbers and calculates their sum.

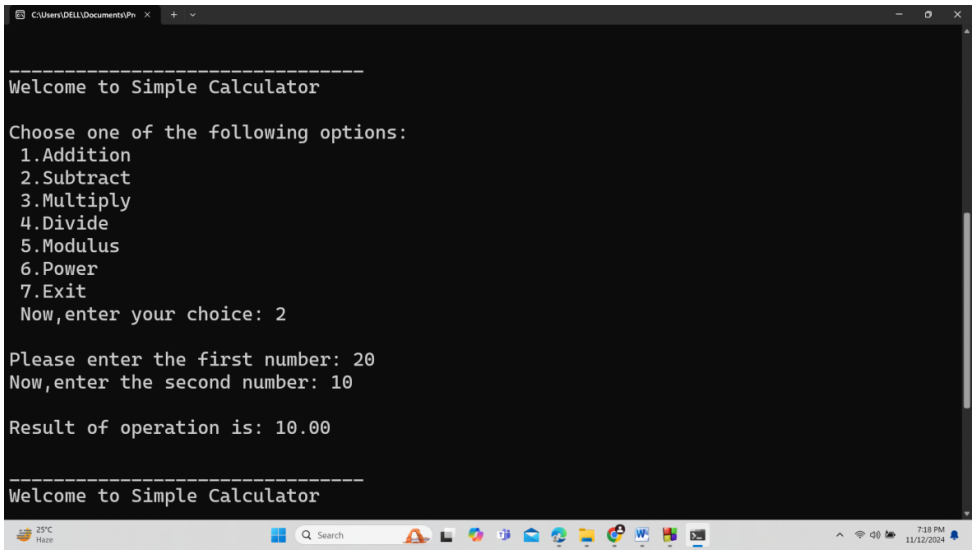
Example Output:



ii) SUBTRACTION (Option 2)

Subtracts the second number from the first.

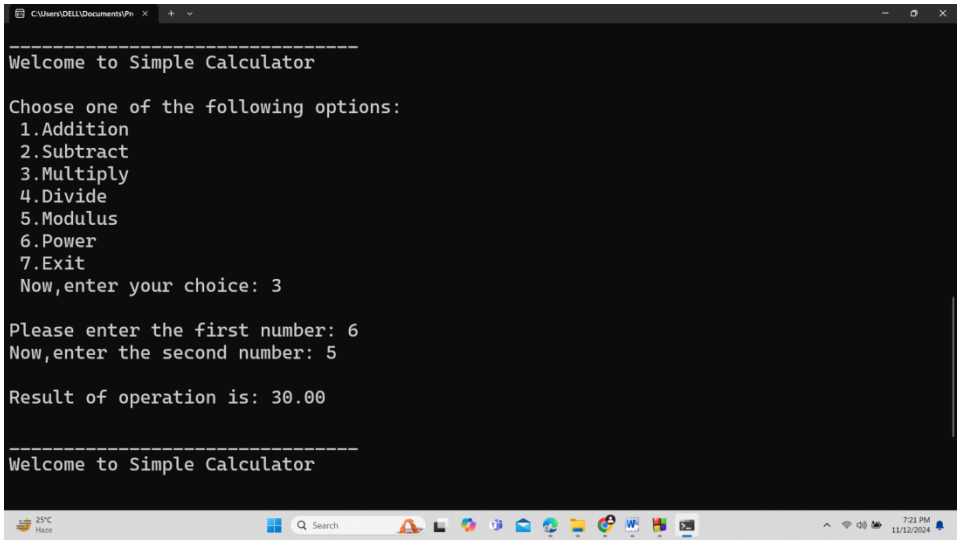
Example output:



iii)MULTIPLICATION (Option 3):

Multiplies the two numbers.

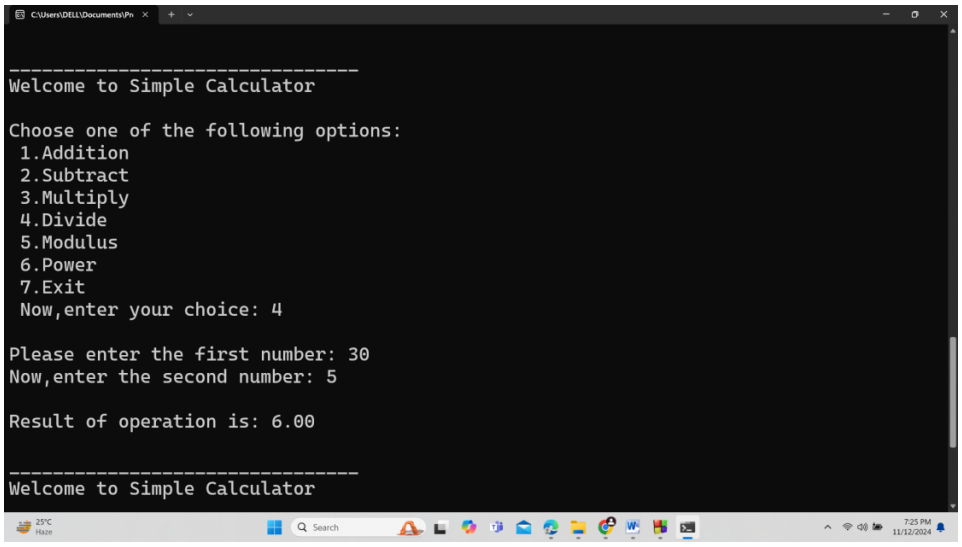
Example output:



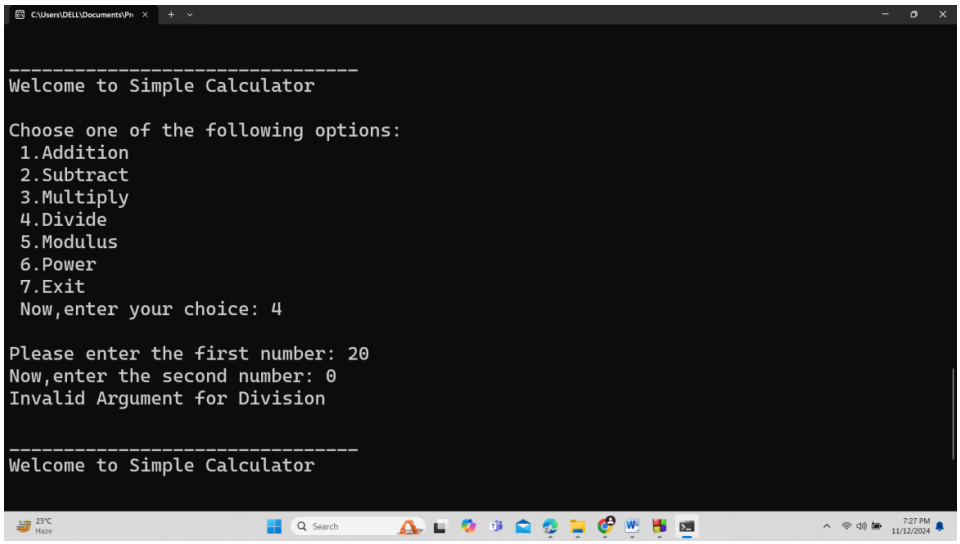
iv)DIVISION (Option 4):

Divides the first number by the second. Includes error handling for division by zero.

Example output (Valid Division) :



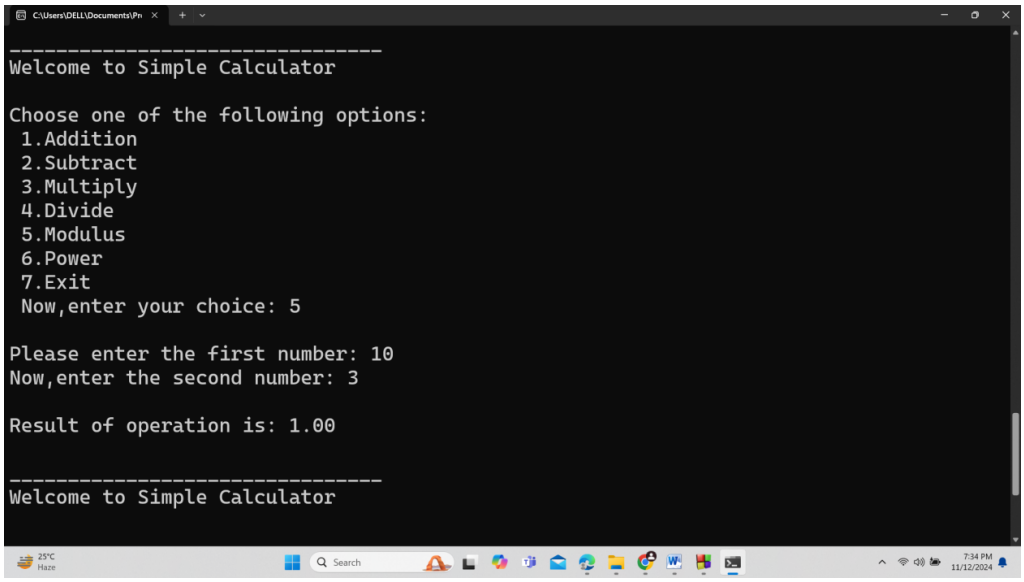
Example Output (Division by Zero):



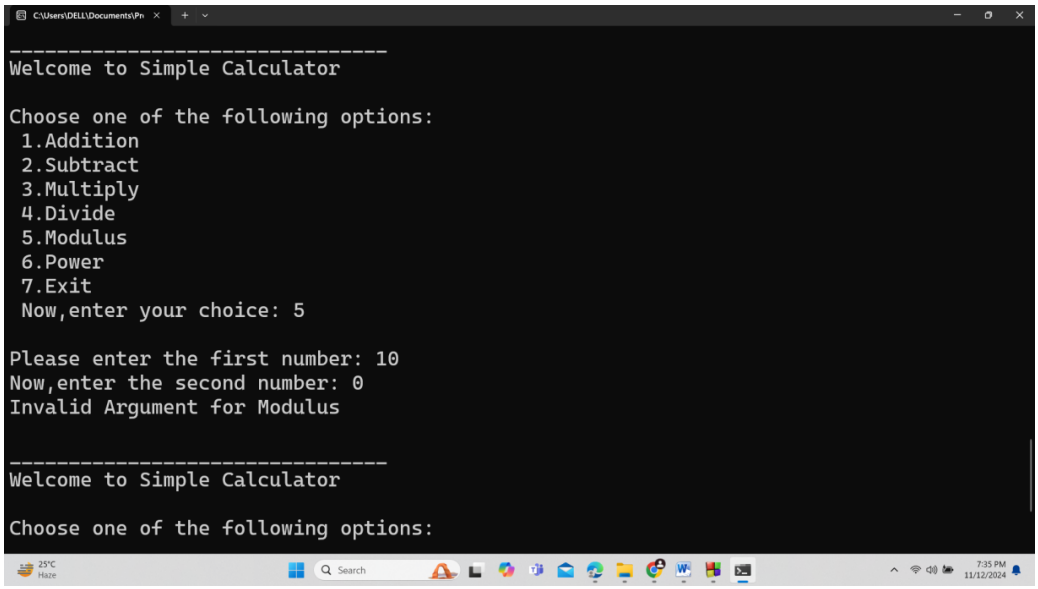
v) MODULUS (Option 5)

Computes the remainder of integer division between the two numbers. Includes error handling for modulus by zero.

Example output (Valid Division) :



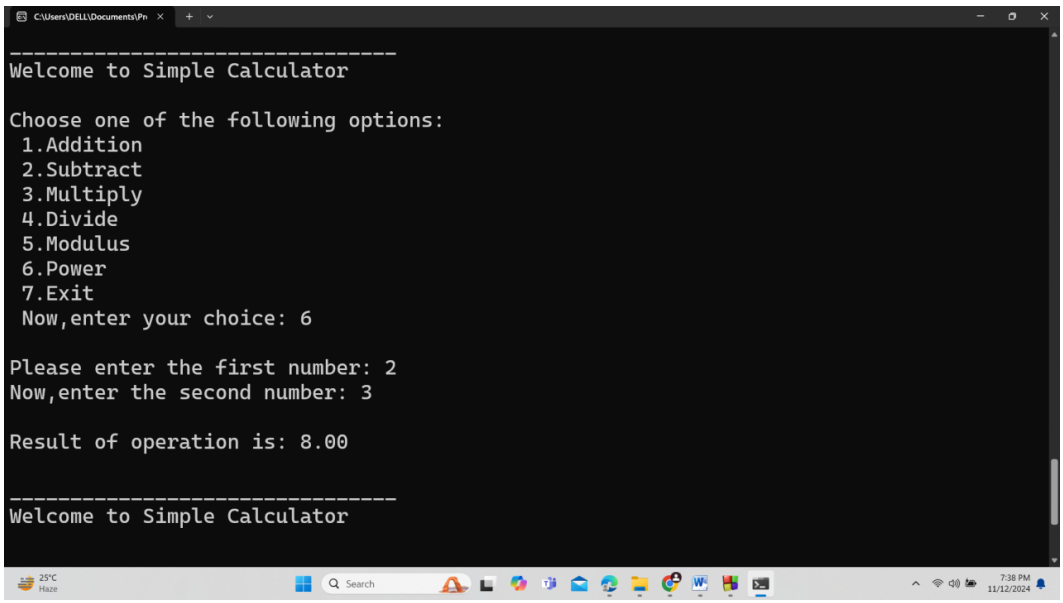
Example Output (Modulus by Zero):



vi)POWER (Option 6) :

Computes the first number raised to the power of the second using the pow() function.

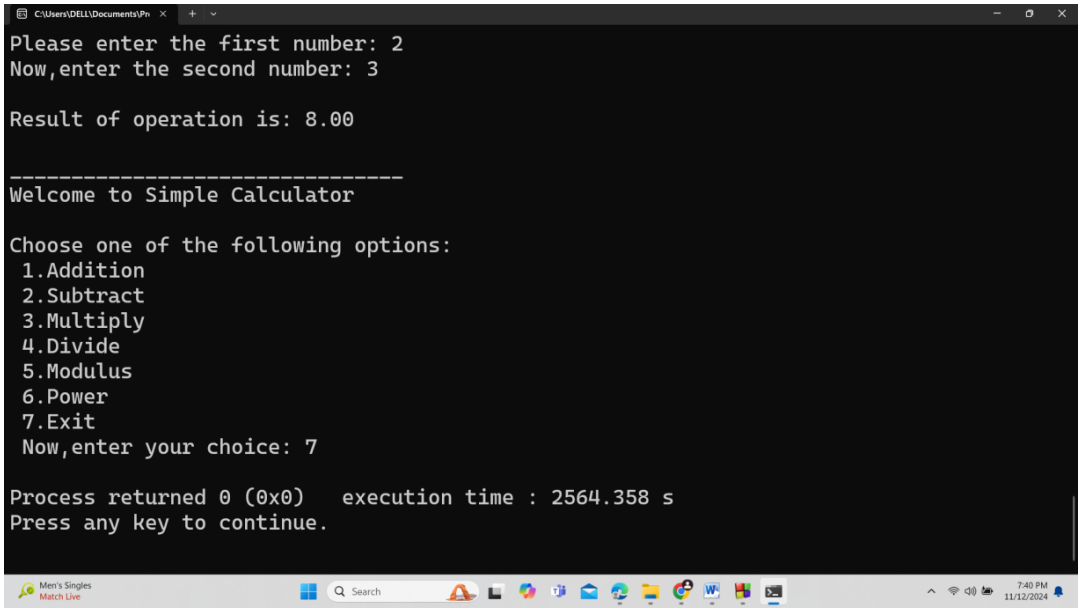
Example Output:



Vii)EXIT (Option 7):

Terminates the program.

Example:



4. Limitations

While the program implements basic arithmetic operations, several limitations must be addressed:

Data Type Limitations: The program only supports operations for a limited set of data types. For example, modulus operation is only supported for integers. This may limit its usability for users who want to use non-integer numbers with modulus.

No Input Validation: Although the program handles some errors (like division by zero), it does not validate if the user inputs a number at all. It could crash or behave unexpectedly if the user enters non-numeric input.

Limited User Interaction: The program operates in a linear fashion with only one user input sequence per session. There is no option for users to perform multiple calculations without restarting the program.

Limited Error Handling: While division by zero and modulus by zero are addressed, other error conditions (such as invalid arithmetic operations or out-of-range values) are not handled.

Floating-point Precision: The floating-point results are displayed with only two decimal places, which may not be accurate for certain calculations that require more precision.

5. Opportunities

The project offers several learning and enhancement opportunities:

Understanding Core Concepts: Working on this project helps learners deepen their understanding of control structures (like loops and conditionals), function creation, and error handling in C programming.

Enhancing User Experience: By adding features like detailed error messages, retry prompts, or more intuitive user prompts, the program can be made more user-friendly, guiding users through potential mistakes or issues.

Extensibility: The modular design of the calculator allows for easy addition of new operations, such as square roots, trigonometric functions, or logarithms, making the project highly adaptable and scalable for future improvements.

6. Future Plan

In the future, there are several improvements and features that can be added to the project:

Enhanced Input Validation: Implement better checks to handle non-numeric inputs and prompt users to enter valid numbers, preventing crashes or unexpected behavior.

Support for Complex Numbers: Extend the functionality to handle operations on complex numbers, allowing users to perform calculations involving both real and imaginary parts.

Advanced Mathematical Functions: Include advanced operations such as square roots, logarithms, trigonometric functions (like sine, cosine), and factorials, making the calculator more versatile for scientific use.

Graphical User Interface (GUI): Develop a GUI version of the calculator to provide a more user-friendly experience, making it accessible to a wider audience who may not be familiar with command-line tools.

Precision Control: Allow users to specify the desired precision for the results, ensuring more accurate outputs for complex or sensitive calculations.

Error Logging: Implement an error logging system that records all errors (such as invalid inputs or operation failures) to a file for future debugging and analysis, improving the maintainability of the program.

7.Conclusion

This project successfully demonstrates the creation of a simple yet functional calculator using C. It effectively highlights key programming concepts such as control structures, functions, error handling, and user input management. The project not only serves as a solid foundation for learning these essential skills but also offers numerous opportunities for future enhancements. With its modular design, the calculator can be easily expanded to include more complex operations and features. Overall, this project provides a great starting point for building more sophisticated software systems and improving one's programming expertise.

REFERENCE-

https://youtu.be/ShlJel8gOeU?si=kvw-aiSI2a4gA6PQ