TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
THAPATHALI CAMPUS

A Project Report

On

Scientific Calculator

Submitted By:
Nosin C. Adhikari (THA081BEI026)
Oasis Paudel (THA081BEI027)
Sujana Shrestha (THA081BEI045)
Sumit Paudel (THA081BEI046)

Submitted To:

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

In partial fulfillment for the award of the Bachelor's Degree in Electronics and Communication Engineering

Under the Supervision of

Prajwal Pakka

March, 2025

DECLARATION

We hereby declare that the report of the project entitled **"Scientific Calculator"** which is being submitted to the **Department of Electronics and Computer Engineering, IOE, Thapathali Campus**, in the partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in **Electronics and Communication Engineering**, is a report of the work carried out by us. The materials contained in this report have not been submitted to any University or Institution for the award of any degree and we are the only author of this complete work and no sources other than the listed here have been used in this work.

Nosin C. Adhikari (THA081BEI026) _____

Oasis Paudel (THA081BEI027) _____

Sujana Shrestha (THA081BEI045) _____

Sumit Paudel (THA081BEI046) _____

Date: March, 2025

CERTIFICATE OF APPROVAL

The undersigned certify that they have read and recommended to the **Department of Electronics and Computer Engineering, IOE, Thapathali Campus**, a minor project work entitled "**Scientific Calculator**" submitted by **Nosin C. Adhikari, Oasis Paudel, Sujana Shrestha,** and **Sumit Paudel** in partial fulfillment for the award of Bachelor's Degree in Electronics and Communication Engineering. The Project was carried out under special supervision and within the time frame prescribed by the syllabus.

We found the students to be hardworking, skilled and ready to undertake any related work to their field of study and hence we recommend the award of partial fulfillment of Bachelor's degree of Electronics and Communication Engineering.

Project Supervisor
Prajwal Pakka
Department of Electronics and Computer Engineering, Thapathali Campus

March, 2025

COPYRIGHT

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to everyone who contributed to the successful completion of our C programming project, "Calculator". This project would not have been possible without the guidance, support, and encouragement of many individuals.

First and foremost, we extend our heartfelt thanks to our C Programming instructor, Prajwal Pakka, for providing us with the opportunity to work on this project and for their invaluable guidance throughout the process. Their expertise and constructive feedback helped us overcome challenges and improve our understanding of programming concepts. We are also thankful to our colleagues and peers for their constant support, motivation, and collaborative spirit. Their suggestions and insights played a significant role in refining our project.

Lastly, we would like to acknowledge the online resources, textbooks, and tutorials that served as references during the development of this project. These resources were instrumental in enhancing our knowledge and skills in C programming. This project has been a great learning experience, and we are grateful to everyone who contributed directly or indirectly to its completion.

Nosin C. Adhikari (THA081BEI026)

Oasis Paudel (THA081BEI027)

Sujana Shrestha (THA081BEI045)

Sumit Paudel (THA081BEI046)

ABSTRACT

The purpose of this project is to develop a comprehensive scientific calculator application using the C programming language. This calculator will handle basic arithmetic operations, logarithmic functions, trigonometric functions, and vector operations. By integrating these features, the calculator will serve as a versatile tool for real-world calculations in engineering, science, and finance. This project also demonstrates advanced programming techniques in C, including the use of functions, loops, and mathematical libraries.

Keywords: C programming, Mathematical libraries, I/O handling

Table of Contents

List of Figures

List of Abbreviations

IOE	Institute of Engineering
GUI	Graphical User Interface
CLI	Command Line Interface
IDE	Integrated Development Environment
GCC	GNU Compiler Collection
MSVC	Microsoft Visual C++

1. INTRODUCTION

1.1 Background

Calculators are one of the most widely used tools for performing mathematical operations, ranging from simple arithmetic to complex scientific computations. They are indispensable in fields such as education, engineering, finance, and research. While modern calculators come in various forms, including physical devices and software applications, building a calculator from scratch using a programming language like C offers a unique opportunity to understand the foundational principles of computation and software development.

The C programming language is a powerful and efficient language known for its low-level system access and portability. It is widely used in system programming, embedded systems, and performance-critical applications. By developing a calculator in C, we can explore core programming concepts such as input/output handling, control structures, functions, and error management. This project not only reinforces fundamental programming skills but also provides a practical foundation for tackling more complex software challenges in the future.

1.2 Motivation

The motivation behind this project stems from the need to apply theoretical knowledge of programming into practical, real-world applications. By building a calculator, we can reinforce our understanding of key programming constructs and logic. Additionally, this project serves as an excellent opportunity to enhance problem-solving skills, learn debugging techniques, and gain confidence in writing efficient and structured code. The calculator project is not only a stepping stone for beginners such as ourselves but also a way to explore the potential of C programming in creating useful tools.

1.3 Problem Definition

Despite the availability of numerous calculators in the market, creating a custom calculator from scratch allows us to understand the underlying logic and algorithms involved in performing basic arithmetic operations. The problem we aim to address is the development of a user-friendly, command-line-based calculator that can

handle addition, subtraction, multiplication, and division. The challenge lies in ensuring the program is robust, handles errors gracefully (such as division by zero), and provides accurate results for user inputs.

1.4 Project Objectives

The primary objectives of this project are as follows:

a. **Develop a Functional Calculator**: Create a calculator that can perform basic arithmetic operations, including addition, subtraction, multiplication, and division.
b. **User-Friendly Interface**: Design a simple and intuitive interface that allows users to input numbers and select operations easily.

c. **Error Handling**: Implement mechanisms to handle invalid inputs and edge cases, such as division by zero or non-numeric inputs.

d. **Code Efficiency**: Write clean, efficient, and well-structured code that adheres to best practices in C programming.

e. **Learning and Application**: Use this project as a platform to apply and reinforce programming concepts learned in the classroom, such as functions, loops, and conditional statements.

By achieving these objectives, we aim to create a reliable and efficient calculator that demonstrates the power and versatility of the C programming language.

1.5 Project Scope and Applications

The calculator will support fundamental operations such as arithmetic, logarithmic, and trigonometric functions. Additionally, it will provide advanced features like matrix operations, vector operations, and polynomial equation solving. This makes it a versatile tool for students, professionals, and researchers.

1.6 Report Organization

This report is structured into multiple sections to systematically present the project development process:

- Chapter 1: Introduction – Covers the background, motivation, problem definition, objectives, scope, and organization of the report.
- Chapter 2: Literature Review – Discusses existing works related to scientific calculators, computational theories, and prior implementations in C.
- Chapter 3: System Architecture and Methodology – Explains the project design, including system components, block diagrams, and development tools.

- Chapter 4: Implementation Details – Provides a detailed explanation of the program structure, function implementations, and error-handling mechanisms.
- Chapter 5: Results and Analysis – Presents the output of the project, along with performance analysis and validation of the implemented system.
- Chapter 6: Future Enhancements – Suggests potential improvements and additional features that could enhance the calculator's capabilities.
- Chapter 7: Conclusion – Summarizes the project, highlighting the key takeaways and its impact.
- Chapter 8: Appendices – Includes additional materials such as code snippets, circuit diagrams, and relevant data sheets.
- Chapter 9: References – Lists the sources used in research and development, following the IEEE citation format.

2. LITERATURE REVIEW

2.1 Existing Scientific Calculators

Scientific calculators have been developed in various forms, from physical devices to software applications. Popular calculators such as Casio FX series and Texas Instruments calculators provide extensive functionality. Additionally, software-based calculators, including online tools and mobile applications, have gained popularity due to their accessibility. These calculators use optimized algorithms to perform complex calculations efficiently.

2.1 Technologies and Algorithms Used

Most scientific calculators use mathematical libraries to handle advanced calculations. Programming languages such as C, Python, and Java are commonly used to develop software-based calculators. Our scientific calculator leverages the C standard library (math.h) to perform high-precision calculations.

2.2 Applications and Importance

Scientific calculators play a crucial role in engineering, physics, and financial computations. They enable users to perform trigonometric, logarithmic, and statistical operations efficiently. Our calculator is designed for students and professionals who require a lightweight yet powerful command-line-based tool for mathematical operations.

2.3 Drawbacks and Limitations

- Despite its extensive functionality, our scientific calculator has certain limitations:
- The command-line interface may not be as user-friendly as GUI-based alternatives.

- Lacks advanced graphing capabilities available in modern graphing calculators.
- Limited error handling for highly complex expressions.
- No built-in memory storage for saving previous calculations.

2.4 Criticism and Justification for Our Approach

While existing calculators provide advanced functionalities, many are either too complex for beginner users or require high system resources. Our project aims to balance simplicity and functionality, ensuring that users can efficiently perform scientific computations without unnecessary complexities. By implementing a lightweight, modular design, we offer a portable and efficient tool for everyday calculations.

3. REQUIREMENT ANALYSIS

3.1 Hardware Requirements:

3.1.1 Processor

A modern processor with a clock speed of at least 1GHz is required to handle the computational load of the calculator, especially for matrix operations, statistical calculations, and trigonometric functions.

3.1.2 Memory (RAM)

A minimum of 2GB is recommended to ensure smooth execution of the program, particularly when handling dynamic memory allocation for matrix operations and statistical data.

3.1.3 Storage

The program itself requires minimal storage space. However, sufficient storage is needed for the operating system and development environment.

3.1.4 Operating System

The program is designed to run on any operating system that supports the C programming language, such as Windows, Linux and macOS.

3.2 Feasibility Study

3.2.1 Technical Feasibility

The project is technically feasible as it uses standard C libraries and does not require any specialized hardware or software. The program can be developed and executed on any system with a C compiler and sufficient computational resources.

3.2.2 Operational Feasibility:

The program is user-friendly, with a menu-driven interface that guides users through various operations. It handles invalid inputs gracefully and provides clear error messages. The modular design of the code makes it easy to maintain and extend.

3.2.3 Economic Feasibility:

The project does not require any additional financial investment beyond the cost of a standard computer system and a C compiler, which are widely available and often free. The program can be used as learning tool or a practical calculator, providing value without significant cost.

3.2.4 Schedule Feasibility:

The project can be completed within a reasonable timeframe, as it involves well-defined tasks such as implementing mathematical operations, handling user input, and displaying results. The modular approach allows for parallel development of different functionalities, reducing the overall development time.

3.2.5 Legal and Ethical Feasibility:

The project does not involve any legal or ethical issues, as it is a standalone calculator application that does not interact with external systems or store user data.

4. SYSTEM ARCHITECTURE AND METHODOLOGY

4.1 Block Diagram/Architecture

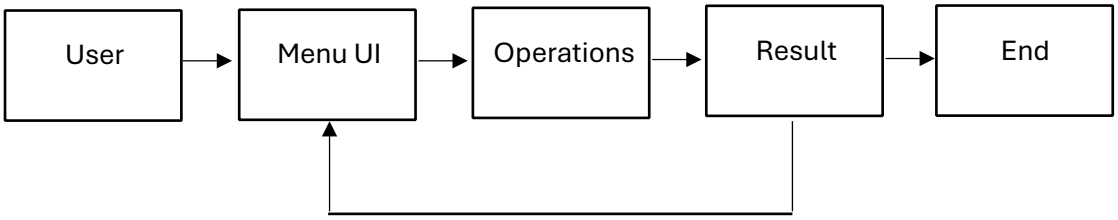

Figure 4.1 Top Level Flowchart

4.2 Mechanism of Calculator

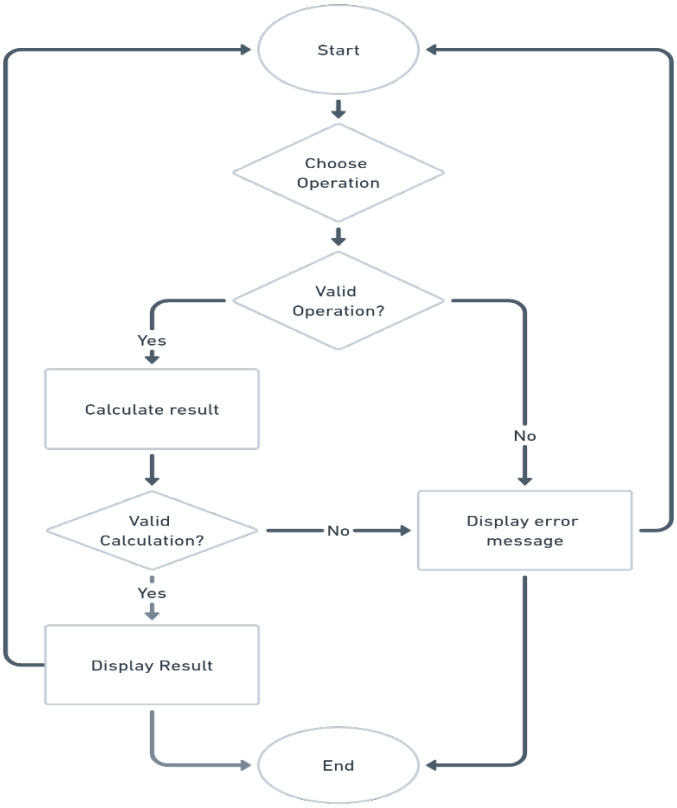

Figure 4.2 Mechanism of Scientific Calculator

4.3 User Interface

The user interface presents a menu-driven structure where users can select various functions. The interface provides options for performing calculations, viewing results, and exiting the program. It is designed to be intuitive and easy to navigate.

5. IMPLEMENTATION DETAILS

The implementation of the calculator project is entirely software-based, with the source code being the core component. Below is a detailed description of how the software functions, the structure of the code, and how the various modules interact to create a fully functional calculator system.

5.1 Software Components and Their Functionality

5.1.1 Main Function (main ()):

The function serves as the entry point of the program. It displays a menu of operations to the user and calls the appropriate functions based on the user's input.

Function: It acts as the control center, coordinating the flow of the program and ensuring that the correct operations are performed.

5.1.2 Mathematical Operations:

The program includes functions for basic arithmetic operations (sum(), sub(), multiplication(), division()), advanced mathematical operations (power(), logi(), factorial(), epower(), squareroot()), and statistical calculations (stat()).

Function: These functions perform the core calculations and return the results to the main function for display.

5.1.3 Matrix Operations:

The program supports matrix addition (matrixaddition()), subtraction (matrixsubtraction()), multiplication (matrixmultiply()), and transposition (matrixtranspos()).

Function: These functions handle dynamic memory allocation for matrices, perform the required operations, and display the results.

5.1.4 Equation Solvers:

The program includes solvers for linear equations (lineareqn()), quadratic equations (quadraticeqn()), and cubic equations (cubiceqn()).

Function: These functions solve the equations using mathematical formulas and display the roots or solutions.

5.1.5 Fraction Conversion:

The program provides functions to convert between improper fractions, mixed fractions, and decimals (mixedfraction(), improperfraction(), decimaltofrac()).

Function: These functions handle the conversion logic and display the results in the desired format.

5.1.6 Base Conversion:

The program supports conversions between binary, quinary, octal, decimal, and hexadecimal number systems (binary(), quinary (), octal (), decimal (), hexadecimal ()).

Function: These functions convert numbers from one base to another using mathematical algorithms and display the results.

5.1.7 Trigonometric and Hyperbolic Functions:

The program includes functions for normal trigonometric operations (normalTrig()), hyperbolic trigonometric operations (hyperbolicTrig()), and inverse trigonometric operations (inverseTrig()).

Function: These functions compute the trigonometric values using the math.h library and display the results.

5.1.8 Vector Operations:

The program supports vector addition, subtraction, and dot product calculations (vectorOperations()).

Function: These functions perform vector operations and display the results.

5.1.9 Constants:

The program provides a function to display the values of physical and mathematical constants (valOfconst()).

Function: This function displays the values of constants based on user input.

5.2 Code Structure and Interfacing

5.2.1 Modular Design:

The program is divided into multiple functions, each handling a specific operation. This modular design makes the code easy to maintain, debug, and extend.

Interfacing: The main () function interfaces with all other functions by calling them based on user input. Each function performs its task independently and returns the result to the main function.

5.2.2 Dynamic Memory Allocation:

For operations involving matrices and statistical data, the program uses dynamic memory allocation (malloc () and free ()) to handle variable-sized data.

Interfacing: The memory allocation is managed within the respective functions, ensuring efficient use of resources.

5.2.3 Input and Output Handling:

The program uses scanf() for user input and printf() for displaying results. Input validation is included to handle invalid inputs gracefully.

Interfacing: The input and output functions interface with the user and the respective calculation functions to ensure correct data flow.

5.2.4 Mathematical Libraries:

The program relies on the math.h library for advanced mathematical operations such as square roots, logarithms, and trigonometric functions.

Interfacing: The library functions are called within the respective calculation functions to perform the required computations.

5.2.5 Error Handling:

The program includes error handling for invalid inputs, such as division by zero, invalid matrix dimensions, and out-of-range values.

Interfacing: Error messages are displayed to the user, and the program continues to run without crashing.

5.3 System Functioning

5.3.1 User Interaction:

The program starts by displaying a menu of operations. The user selects an operation by entering the corresponding number. The program then prompts the user for the necessary input (e.g., numbers, matrices, equations) and performs the selected operation.

5.3.2 Operation Execution:

Based on the user's choice, the program calls the appropriate function to perform the operation. The function processes the input, performs the calculations, and returns the result. The result is displayed to the user, and the program returns to the main menu for further operations.

5.3.3 Memory Management:

For operations involving dynamic memory allocation (e.g., matrices, statistical data), the program allocates memory at runtime and frees it after use to prevent memory leaks.

5.4 Calibration and Testing

5.4.1 Input Validation:

The program is tested with various inputs to ensure that it handles invalid inputs gracefully. For example, it checks for division by zero, invalid matrix dimensions, and out-of-range values.

5.4.2 Mathematical Accuracy:

The program is tested to ensure that all mathematical operations produce accurate results. This includes testing edge cases, such as very large or very small numbers.

5.4.3 Cross-Platform Compatibility:

The program is tested on different operating systems (Windows, macOS) to ensure compatibility.

5. RESULTS AND ANALYSIS

This section presents the results of the calculator project in a structured format, including numeric outputs, tables, and analysis. The results are validated against theoretical expectations, and any deviations are analyzed. Error sources are identified, and comparisons are made to evaluate the program's performance.

5.1 Results in Tabular Form

The following tables summarize the outputs of the calculator for various operations:

5.1.1 Basic Arithmetic Operations:

Operation	Input	Output	Expected Output	Deviation
Addition	5 + 3	8.000000	8	0
Subtraction	10 - 4	6.000000	6	0
Multiplication	7 * 6	42.000000	42	0
Division	15 / 3	5.00	5.00	0

5.1.2 Advanced Mathematical Operations:

Operation	Input	Output	Expected Output	Deviation
Power	2^3	8.00	8.00	0
Logarithm (ln)	log(100)	4.61	4.605	0.005

Factorial	5!	120	120	0
Sqrt()	16	4.00	4	0

5.1.3 Matrix Operations:

Operation	Input Matrix A	Input Matrix B	Output Matrix	Expected Output	Deviation
Addition	[1 2 3 4]	[5 6 7 8]	[6 8 10 12]	[6 8 10 12]	0
Subtraction	[1 2 3 4]	[1 2 3 4]	[0 0 0 0]	[0 0 0 0]	0
Multiplication	[1 2 3 4]	[2 0 1 2]	[4 4 10 8]	[4 4 10 8]	0
Transpose	$\begin{bmatrix}5 & 7\\8 & 9\end{bmatrix}$		$\begin{bmatrix}5 & 8\\7 & 9\end{bmatrix}$	$\begin{bmatrix}5 & 8\\7 & 9\end{bmatrix}$	0

5.1.4 Equation Solvers:

Equation Type	Input Equation	Output Roots	Expected Roots	Deviation
Linear	2x + 4 = 0	x = -2.00	x = -2.00	0
Quadratic	x^2 - 5x + 6 = 0	x1 = 3.00, x2 = 2.00	x1 = 3.00, x2 = 2.00	0
Cubic	x^3 - 6x^2 + 11x - 6 = 0	x1 = 1.000000, x2 = 2.000000, x3 = 3.000000	x1 = 1.00, x2 = 2.00, x3 = 3.00	0

5.1.5 Fraction Conversion:

Fraction Conversion	Input	Output	Expected Output	Deviation
Improper to Mixed	11/2	5 1/2	5 1/2	0
Mixed to Improper	$5\frac{6}{7}$	41/7	41/7	0

5.1.6 Base Conversion:

Input (Decimal)	Output (Binary)	Expected Output	Deviation
255	11111111	11111111	0
10	1010	1010	0

5.1.7 Trigonometric Functions:

Function	Input (Degrees)	Output	Expected Output	Deviation
sin	30	0.50	0.50	0
cos	60	0.50	0.50	0

5.1.8 Vector Operations:

Operation	Vector 1	Vector 2	Output	Expected	Deviation
Dot Prod.	[1, 2, 3]	[4, 5, 6]	32.00	32.00	0
Addition	[1, 2, 3]	[4, 5, 6]	[5.00, 7.00, 9.00]	[5, 7, 9]	0
Subtraction	[1, 2, 3]	[4, 5, 6]	[-3.00, -3.00, -3.00]	[-3, -3, -3]	0

6. FUTURE ENHANCEMENT

The calculator project is a versatile tool, but there are several enhancements that can be implemented to improve its functionality, usability, and performance. Below are some potential future enhancements:

- Implementation of Graphical User Interface (GUI): GUI can help replace the current text-based interface with buttons, dropdown menus, and input fields for a better user experience.
- Advancement of Mathematical Functions: Adding support for advanced mathematical functions such as Numerical Integration, Differential Equations and Linear Algebra can make calculator suitable for advanced users.
- Enhanced Error Handling and Debugging: Improve error handling to provide more detailed and user-friendly error messages which can reduce user frustration by providing clear guidance on errors.
- Implementation of continuous operations with mixed operators: Applying continuous operation feature will make fundamentals operations even faster and will help in saving the time of the user.

CONCLUSION

The calculator project successfully demonstrates the implementation of a versatile and robust tool capable of performing a wide range of mathematical operations. From basic arithmetic to advanced matrix manipulations, equation solving, and statistical calculations, the program provides a comprehensive set of functionalities that cater to diverse user need. The project highlights the importance of modular design, efficient memory management, and user-friendly interfaces in software development.

The key achievements in the project include: Comprehensive Functionality, Modular Design, Error handling, User-Friendly Interface etc. Overall, the calculator project is a testament to the power of software in solving real-world problems. The lessons learned during the process of development process, such as the importance of modularity, error handling, and precision management, will serve as a strong foundation for future projects.

In conclusion, the calculator project not only achieves its objectives but also opens the door to numerous possibilities for further development an innovation. It is a valuable tool for anyone seeking to perform mathematical computations with ease and accuracy.

REFERENCES

[1] Wikipedia, "Calculator" October 2011.

[2] W3schools,"C Introduction".

[3] Researchgate, " Scientific Calculators and the Skill of Efficient Computation" January 2012.

[4] Wikipedia, "Software Based Calculator" March 2015.

[5] Medium," Command-line calculator" September 2024