

Multi-Functional Calculator Project

Developed by Sana

Under the expert guidance of **Ms. Naina Devi**

A powerful command-line calculator crafted with precision in C
Programming Language



Agenda

01

Project Introduction

Understanding the vision and core functionality

02

Tools & Technologies

Development environment and technical stack

03

Project Working

Architecture, logic flow, and implementation

04

Live Demonstration

Screenshots and interactive screen recording

05

Applications

Real-world use cases and practical scenarios

06

Advantages

Key benefits and technical strengths

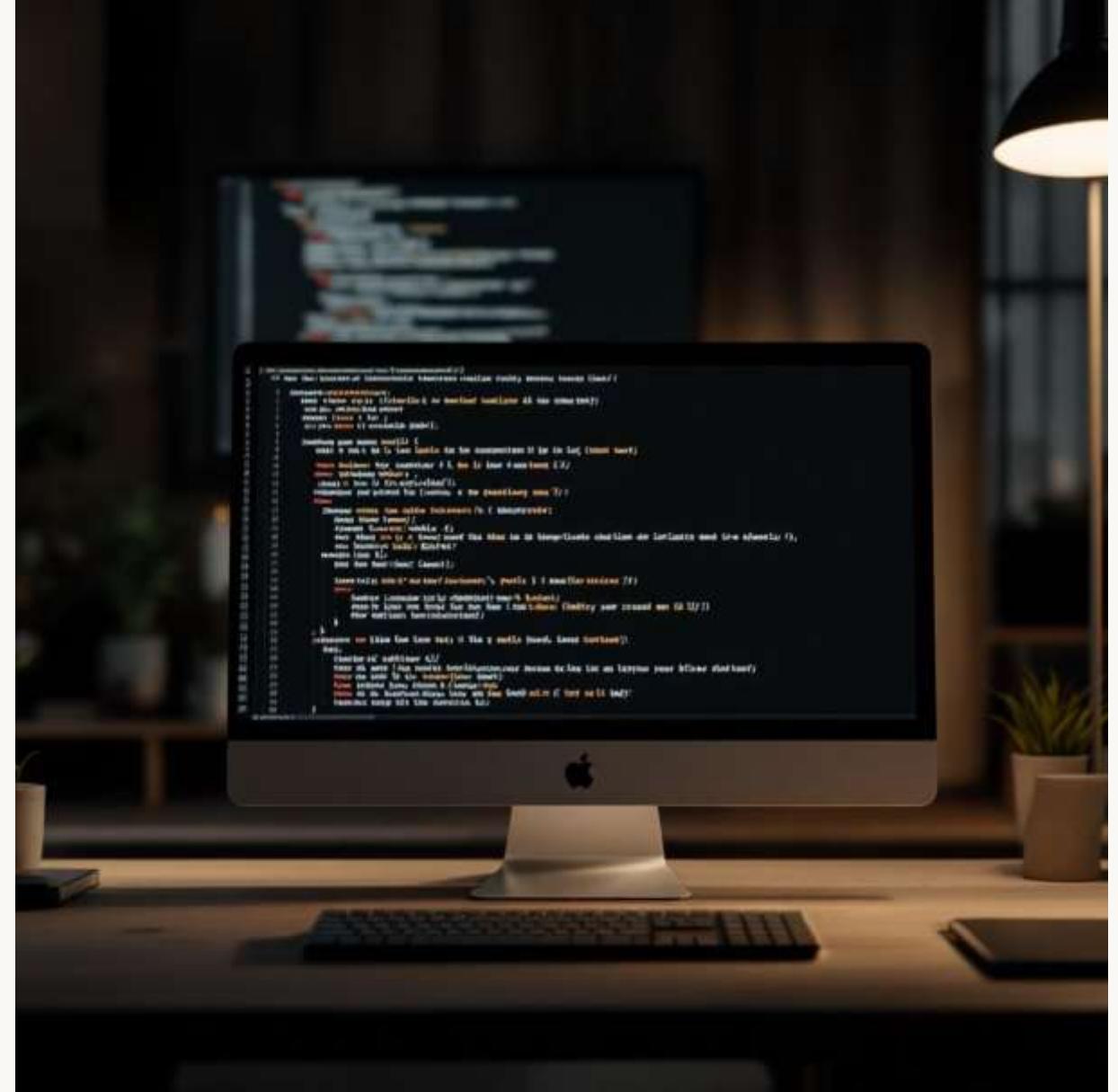
Project Introduction

Building Intelligence Through Simplicity

A comprehensive multi-functional calculator that transforms mathematical operations into seamless command-line experiences. Built with the power and efficiency of C programming, this project demonstrates how fundamental programming concepts create practical, real-world solutions.

Core Capabilities:

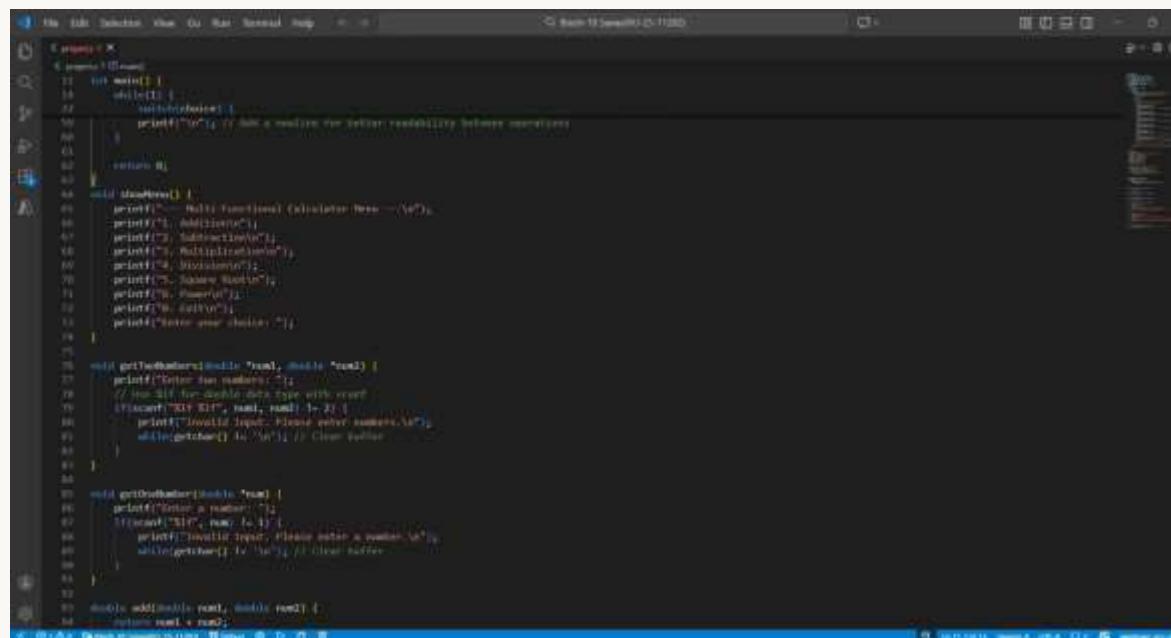
- Arithmetic operations: addition, subtraction, multiplication, division
- Advanced power functions for exponential calculations
- Robust error handling and input validation
- Modular architecture enabling easy extensibility



Project Working: Core Architecture

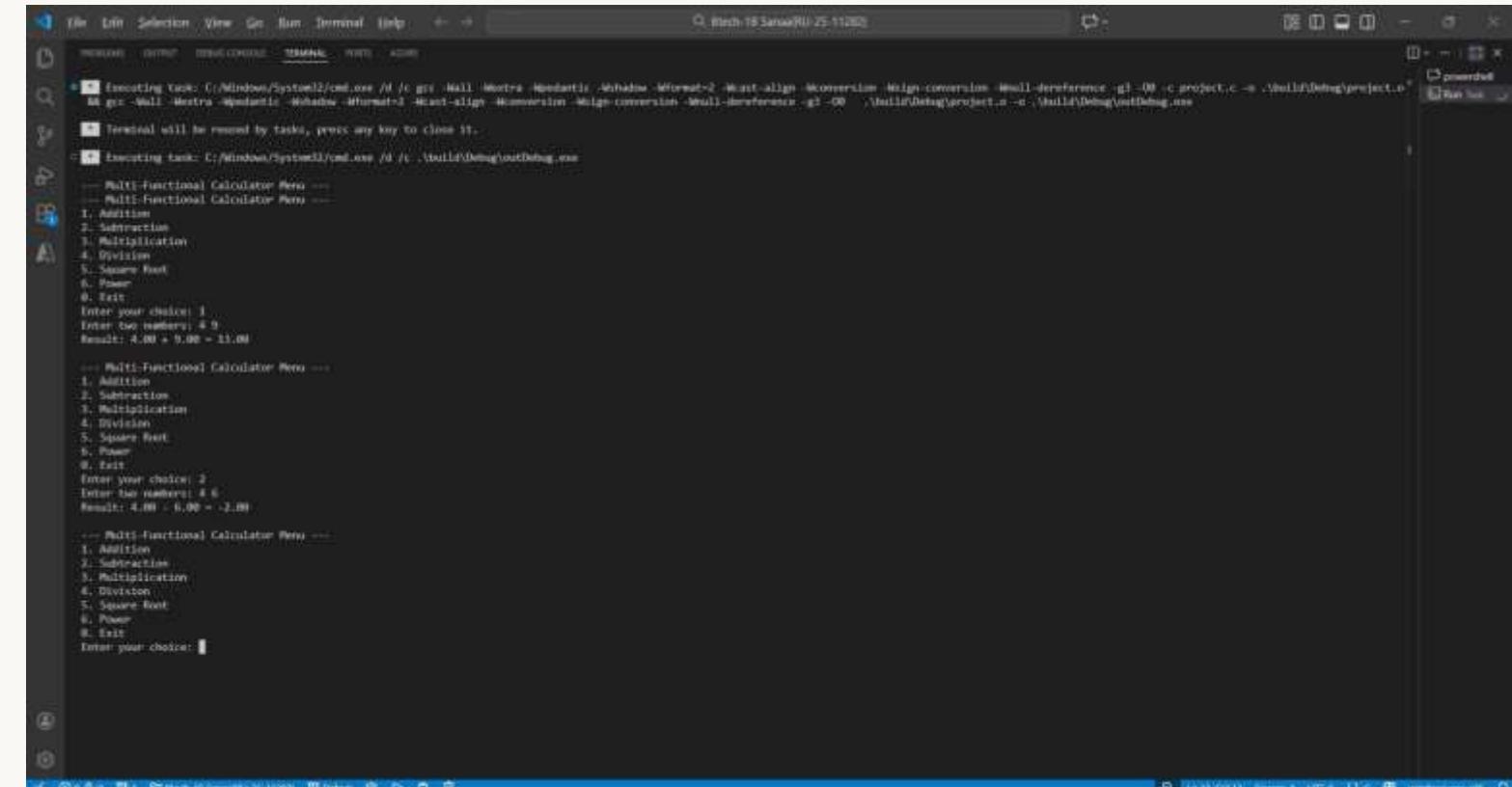
Delve into the foundational C code powering our multi-functional calculator, illustrating how specific operations are implemented and executed in the command line.

code:-



```
1 // Multi-functional Calculator
2
3 #include <cs50.h>
4 #include <stdio.h>
5
6 int main()
7 {
8     printf("Hello! Welcome to the calculator.\n");
9
10    menu();
11
12    return 0;
13 }
14
15 void menu()
16 {
17     printf("1. Addition\n");
18     printf("2. Subtraction\n");
19     printf("3. Multiplication\n");
20     printf("4. Division\n");
21     printf("5. Square Root\n");
22     printf("6. Power\n");
23     printf("7. Exit\n");
24     printf("Enter your choice: ");
25 }
26
27 void addition(double num1, double num2)
28 {
29     printf("Enter two numbers: ");
30     // Use %lf for double data type with scanf
31     scanf("%lf %lf", &num1, &num2);
32     printf("Initial Input: Please enter a number: ");
33     while(getchar() != '\n') // Clear buffer
34     {
35     }
36
37     double result = num1 + num2;
38
39     printf("Result: %f\n", result);
40 }
41
42 void subtraction(double num1, double num2)
43 {
44     printf("Enter two numbers: ");
45     scanf("%lf %lf", &num1, &num2);
46     printf("Initial Input: Please enter a number: ");
47     while(getchar() != '\n') // Clear buffer
48     {
49     }
50
51     double result = num1 - num2;
52
53     printf("Result: %f\n", result);
54 }
55
56 void multiplication(double num1, double num2)
57 {
58     printf("Enter two numbers: ");
59     scanf("%lf %lf", &num1, &num2);
60     printf("Initial Input: Please enter a number: ");
61     while(getchar() != '\n') // Clear buffer
62     {
63     }
64
65     double result = num1 * num2;
66
67     printf("Result: %f\n", result);
68 }
69
70 void division(double num1, double num2)
71 {
72     printf("Enter two numbers: ");
73     scanf("%lf %lf", &num1, &num2);
74     printf("Initial Input: Please enter a number: ");
75     while(getchar() != '\n') // Clear buffer
76     {
77     }
78
79     double result = num1 / num2;
80
81     printf("Result: %f\n", result);
82 }
83
84 void squareRoot(double num)
85 {
86     printf("Enter a number: ");
87     if(num < 0)
88     {
89         printf("Error: Invalid input. Please enter a positive number.\n");
90     }
91     else
92     {
93         double result = sqrt(num);
94
95         printf("Result: %f\n", result);
96     }
97 }
98
99 void power(double num1, double num2)
100 {
101     printf("Enter two numbers: ");
102     scanf("%lf %lf", &num1, &num2);
103     printf("Initial Input: Please enter a number: ");
104     while(getchar() != '\n') // Clear buffer
105     {
106     }
107
108     double result = pow(num1, num2);
109
110     printf("Result: %f\n", result);
111 }
112
113 void exitProgram()
114 {
115     printf("Exiting program...\n");
116 }
```

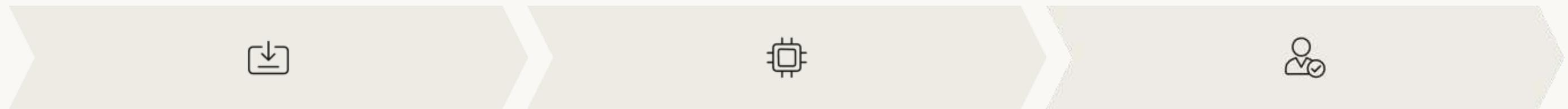
output:-



```
1 Executing task: C:/Windows/System32/cmd.exe /d /p /c g++ -Wall -Wextra -Wpedantic -Wshadow -Wformat=2 -Wcast-align -Wconversion -Wnull-dereference -g3 -O0 -c project.c -o ./build/Debug/project.o
2 Executing task: C:/Windows/System32/cmd.exe /d /p ./build/Debug/project
3
4 --- Multi-Functional Calculator Menu ---
5 1. Addition
6 2. Subtraction
7 3. Multiplication
8 4. Division
9 5. Square Root
10 6. Power
11 7. Exit
12 Enter your choice: 1
13 Enter two numbers: 4.0
14 Result: 4.00 + 5.00 = 9.00
15
16 --- Multi-Functional Calculator Menu ---
17 1. Addition
18 2. Subtraction
19 3. Multiplication
20 4. Division
21 5. Square Root
22 6. Power
23 7. Exit
24 Enter your choice: 2
25 Enter two numbers: 4.0
26 Result: 4.00 - 5.00 = -1.00
27
28 --- Multi-Functional Calculator Menu ---
29 1. Addition
30 2. Subtraction
31 3. Multiplication
32 4. Division
33 5. Square Root
34 6. Power
35 7. Exit
36 Enter your choice: 7
37
```

Project Working: Core Architecture

Intelligent Computational Flow



User Input Stage

System accepts two numerical values and operation selection through keyboard interface

Processing Engine

Dedicated functions execute operations: add(), subtract(), multiply(), divide(), power()

Validation & Output

Error checking handles edge cases; results displayed instantly with precision

Three Core Registers

- **Accumulator:** Stores intermediate results
- **Input Register:** Captures user-provided values
- **Operation Register:** Manages function selection

Robust Error Management

- Division by zero protection
- Invalid input detection and handling
- Graceful error messaging

Live Demonstration

Calculator CLI Interface in Action



Example Calculation Flow

Step 1: User enters first number: 12

Step 2: Selects operation: * (multiplication)

Step 3: Enters second number: 5

Step 4: Calculator displays result: 60

Applications & Use Cases

Educational Excellence

Perfect teaching tool for C programming fundamentals, demonstrating functions, control structures, and modular design through practical application

Professional Utility

Lightning-fast command-line calculations without GUI overhead—ideal for developers and power users who value efficiency

Embedded Systems

Optimized for resource-constrained environments where lightweight applications are essential for system performance

Foundation Platform

Extensible codebase serving as launching point for scientific calculators, graphical interfaces, and specialized computational tools



Advantages

1

Performance Excellence

C language delivers unmatched execution speed and minimal memory footprint, ensuring lightning-fast calculations even on modest hardware configurations

2

Modular Architecture

Function-based design enables effortless maintenance, debugging, and feature additions—each operation isolated for maximum code clarity and reusability

Future Scope & Enhancements

Phase 1: Advanced Mathematics

Implement factorial calculations, square roots, nth roots, logarithmic functions, and exponential operations for comprehensive mathematical coverage



Phase 3: Memory & History

Add memory storage functions (M+, M-, MR, MC) and calculation history tracking for enhanced user productivity and workflow optimization



Phase 2: Graphical Interface

Develop intuitive GUI using GTK+ or Qt framework, bringing visual elegance while maintaining computational power and responsiveness



Phase 4: Scientific Variant

Transform into full scientific calculator with trigonometric functions, unit conversions, constants library, and programmable functions



Phase 5: Embedded Integration

Port to microcontrollers and IoT devices, enabling calculator functionality in embedded applications, robotics, and industrial systems





Thank You!

Questions & Discussion Welcome

"Let's build smarter tools with C programming—where efficiency meets innovation, and code transforms ideas into reality."

Project Developer: Sana

Project Guide: Ms. Naina Devi

Powered by passion, precision, and elegant code