

Mini Project on Calculator Using Python

Project Title: Mini Project on Calculator Using Python

Name: Rudrappa gari vaishnavi

Course: python internship

Company: Slash mark IT solutions

Date: 20/5/2025-25/5/2025

Abstract

This mini project presents the development of a simple calculator using Python. The calculator performs basic arithmetic operations such as addition, subtraction, multiplication, and division. The project demonstrates fundamental programming concepts, including user input, functions, and conditional statements. The system is designed as a command-line application, ensuring ease of use and reliability.

Table of Contents

1. Introduction
2. Objectives
3. System Requirements
4. System Design
5. Implementation
6. Testing and Results
7. System's Performance
8. Comparison of Initial Goals and Actual Results
9. Challenges Faced
10. Conclusion
11. Future Scope
12. References/Bibliography
13. Appendices
14. Acknowledgments

1. Introduction

Calculators are essential tools for performing arithmetic operations efficiently. This project aims to build a basic calculator using Python, a popular programming language known for its simplicity and readability. The project serves as a practical exercise to reinforce programming fundamentals.

2. Objectives

- To develop a functional calculator using Python.
- To implement basic arithmetic operations: addition, subtraction, multiplication, and division.
- To enhance understanding of functions, user input, and control structures in Python

3. System Requirements

Hardware:

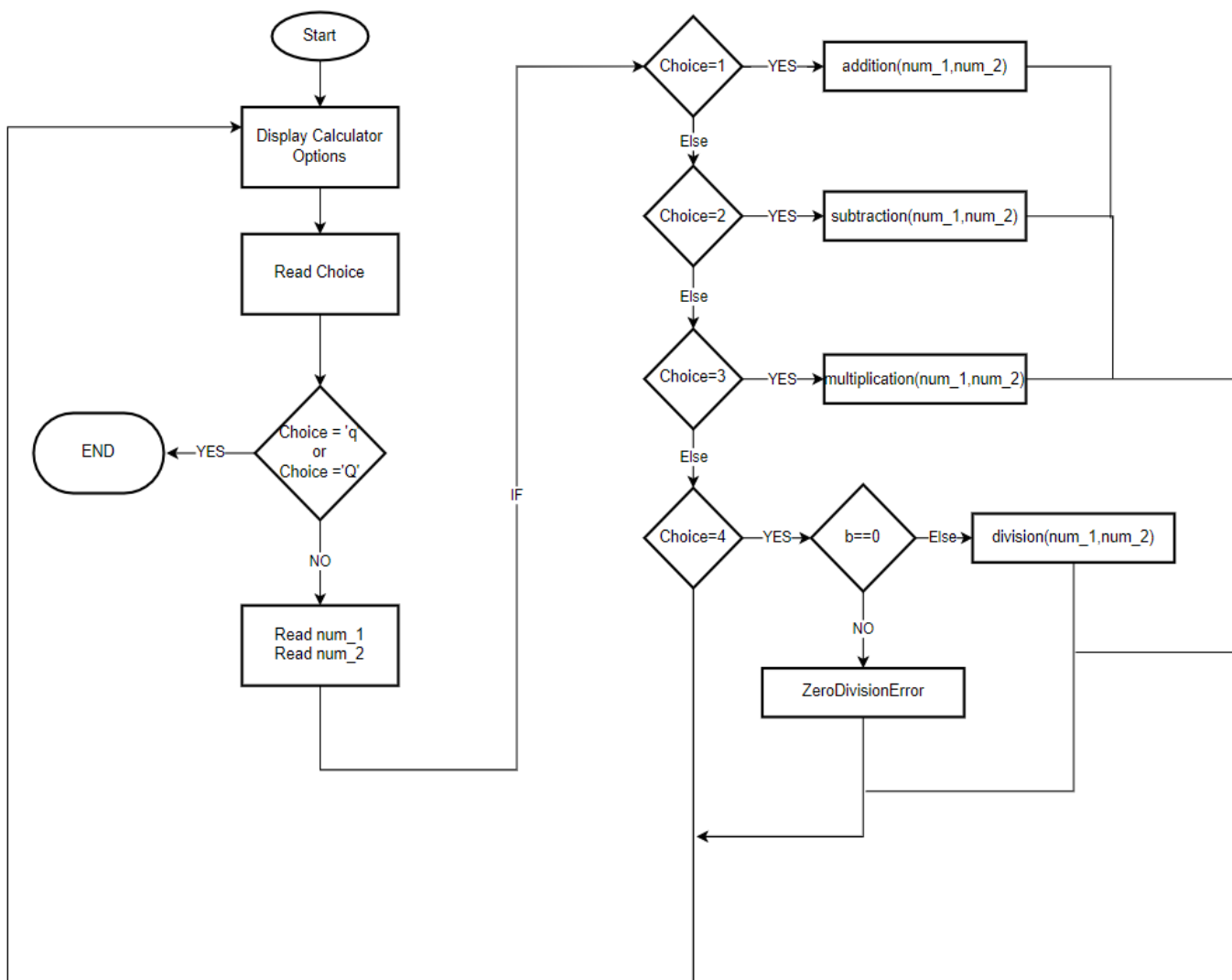
- Processor: Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
- Installed RAM 8.00 GB (7.77 GB usable)
- Device ID 2F0DA166-C34F-44D9-80DC-1264CC873D6C
- System type 64-bit operating system, x64-based processor

Software: Python 3.x, Text Editor (VS Code, Sublime Text).

Operating System: Windows.

4. System Design

Flow chat



Function Structure

- add(x, y)
- subtract(x, y)
- multiply(x, y)
- divide(x, y)

5. Code implementation

```
def calc (x,y,op):
```

```
    if op == '1':
```

```
        return x + y
```

```
    if op == '2':
```

```
        return x - y
```

```
    if op == '3':
```

```
        return x * y
```

```
    if op == '4':
```

```
        if y == 0:
```

```
            return "Error! Division by zero.
```

```
        return x / y
```

```
while True:
```

```
    print("Select operation")
```

```
    print("1.Add")
```

```
    print("2.Subtract")
```

```
    print("3.Multiply")
```

```
    print("4.divide")
```

```
    op = input(" Enter choice (1/2/3/4):")
```

```
if op in ('1','2','3','4'):
```

```
    try:
```

```
        a = float(input("Enter first number:"))
```

```
        b = float(input("Enter second number:"))
```

```
        print("result:",calc (a,b,op))
```

```
    except:
```

```
        print("Invalid input")
```

```
if input("Next calculation?(yes/no):") == "no":
```

```
    break
```

```
else:
```

```
    print("invalid choice")
```

6.Testing and Results

6.1 Test Cases

Test No.	Input 1	Input 2	Operation	Expected Output	Actual Output
1	10	5	Add	15	15.0
2	10	5	Subtract	5	5.0
3	10	5	Multiply	50	50.0
4	10	5	Divide	2	2.0
5	10	0	Divide	Error	Error! Division by zero.

6.2 Sample output

=== Simple Calculator ===

Select operation:

- 1. Add
- 2. Subtract
- 3. Multiply
- 4. Divide
- 5. Exit

Enter choice (1/2/3/4/5): 1

Enter first number: 10

Enter second number: 5

Result: 15.0

7. System’s Performance

- **Efficiency:** The calculator provides instant results for all basic operations.
- **Accuracy:** All operations are computed correctly, including error handling for division by zero.
- **Resource Usage:** Minimal, as it is a command-line application.

8. Comparison of Initial Goals and Actual Results

Initial Goals	Actual Results Achieved
Implement basic arithmetic operations	Achieved
Handle invalid input and division by zero	Achieved
User-friendly interface	Achieved (simple command-line prompts)
Modular code using functions	Achieved

9. Challenges Faced

- **Handling Division by Zero:** Initially, the program would crash if the second number was zero during division. This was resolved by adding a conditional check in the divide() function.
- **Input Validation:** Ensuring the user enters valid numbers and operation choices required additional prompts and error messages.

10. Conclusion

This project successfully demonstrates the design and implementation of a simple calculator using Python. All initial objectives were met, and the project enhanced my understanding of Python programming, especially functions and user interaction. The calculator is accurate, efficient, and easy to use

11. Future Scope

- **Advanced Operations:** Add support for exponentiation, square roots, and trigonometric functions.
- **Graphical User Interface:** Develop a GUI
- **Continuous Calculations:** Allow users to perform multiple calculations without restarting the program.
- **Input Validation:** Enhance input validation for better robustness.

12. References/Bibliography

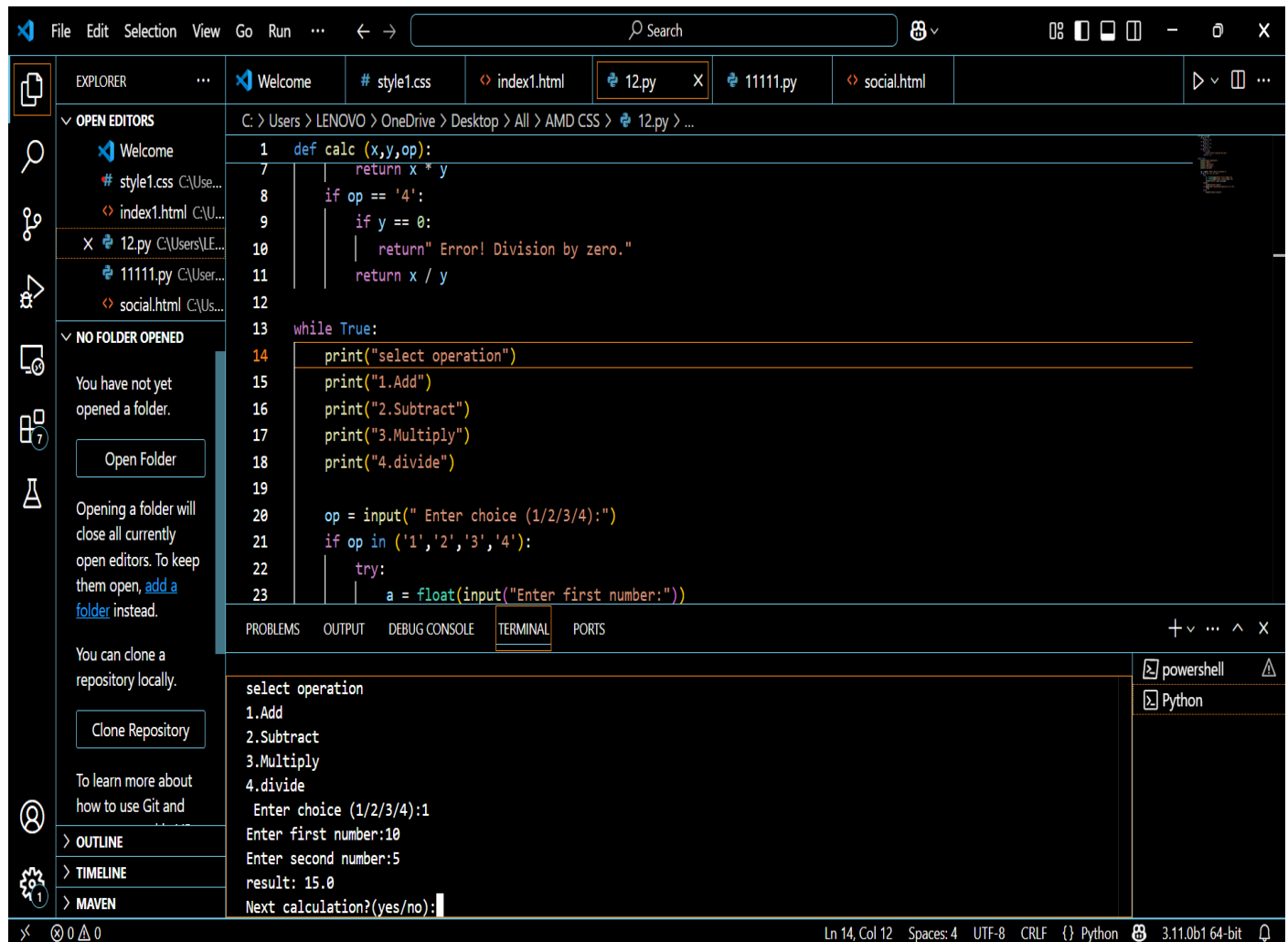
- Codingal. (2023). [How to build a basic calculator in Python](#)
- Shiksha. (2024). [Calculator Program in Python](#)
- Built In. (2024). [How to Make a Python Calculator](#)
- Python.org Documentation. (2024). [Python Input and Output](#)

13. Appendices

A. Full Source Code

(See Section 5: Implementation)

B. Additional Screenshots



14. Acknowledgments

I would like to thank my project guide A. Mohammad Fizan, and my faculty members for their guidance and support throughout this project. I also acknowledge the resources and documentation that aided my learning.

Formatting Guidelines

- Font: Times New Roman or Arial, 12 pt (Headings: 14–16 pt)
- Spacing: 1.5 line spacing
- Margins: 1-inch on all sides
- Alignment: Justify text
- Header/Footer: Include project title or chapter name

Submitted by: Rudrappa gari vaishnavi

Contact details: vaishnavi25045@gmail.com

Phone: 7032588746