



Menu-Driven Calculator Project Documentation

1. Project Overview

This project is a simple **Menu-Driven Calculator** implemented in Python. Its primary function is to perform basic arithmetic operations—addition, subtraction, multiplication, and division—based on user input. The code is structured using **functions** for each operation and a main **loop** to present the menu and handle user interaction.

1.1 Key Learning Objectives

- **Function Definition and Usage:** Defining and calling reusable blocks of code.
- **User Input Handling:** Accepting and processing input from the console.
- **Conditional Logic (if/elif/else):** Implementing decision-making for menu choices and error handling.
- **Looping (while True):** Creating a continuous, interactive user experience (menu).
- **Type Conversion:** Converting string input to numeric types (`float`).
- **Error Handling:** Implementing basic error checking (e.g., division by zero).

2. Source Code Analysis

The calculator is divided into two main parts: **Arithmetic Functions** and the **Main Program Loop**.

2.1 Arithmetic Functions

Four distinct functions are defined, each responsible for a single arithmetic operation.

Function Name	Parameters	Description
<code>add(a, b)</code>	<code>a, b</code>	Returns the sum of <code>a</code> and <code>b</code> .
<code>subtract(a, b)</code>	<code>a, b</code>	Returns the difference of <code>a</code> and <code>b</code> .

<code>multiply(a, b)</code>	a, b	Returns the product of a and b.
<code>divide(a, b)</code>	a, b	Returns the quotient of a and b, or an error message if b is zero.

Division by Zero Error Handling

The `divide` function includes a crucial conditional check to prevent the program from crashing if the user attempts to divide by zero.

Python

```
def divide(a, b):
    if b != 0:
        return a / b
    else:
        return "Division by zero is not allowed" # Returns an error message
```

2.2 Main Program Loop

The program uses a `while True:` loop to keep the calculator running indefinitely until the user explicitly chooses to exit.

A. Menu Display

The loop starts by printing the **Menu**, clearly listing the available operations and the option to exit.

Python

```
print("\nMenu:")
print("1. Addition")
# ... other options
print("5. Exit")
```

B. Exit Condition

The program checks the user's `choice`. If the choice is "5", the `break` statement is executed, which immediately terminates the `while` loop, thus ending the program gracefully.

C. Input and Type Conversion

If the user selects an operation (1-4), the program prompts for two numbers. The `input()` function returns a string, so it must be converted to a floating-point number using `float()` to allow for decimal calculations.

Python

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
```

D. Operation Execution

A series of `if-elif` statements maps the user's choice to the corresponding function call, and the result is printed.

Python

```
if choice == "1":
    print("Result: ", add(num1, num2))
elif choice == "2":
    print("Result: ", subtract(num1, num2))
# ... and so on
```

E. Invalid Choice Handling

An final `else` block catches any choice that is not 1, 2, 3, 4, or 5, providing feedback to the user and restarting the loop.

3. How to Run the Code

1. **Save:** Save the provided code in a file named `calculator.py`.
2. **Execute:** Open your terminal or command prompt and run the file using the Python interpreter:
3. Bash

python calculator.py

- 4.
- 5.
6. **Interact:** The menu will be displayed. Enter the number corresponding to the desired operation (1-4) or enter `5` to exit.