Project Report:

Critical Thinking 1 – Basic Calculator

Alexander Ricciardi

Colorado State University Global

CSC500: Principles of Programming

Professor: Dr. Brian Holbert

September 14, 2025

Project Report:

Critical Thinking Assignment 1 – Basic Calculator

This documentation is part of the Critical Thinking Assignment 1 from CSC500: Principles of Programming at Colorado State University Global. This Project Report is an overview of the program's functionality and testing scenarios, including console output screenshots. The program is coded in Python 3.12, and it is called Critical Thinking Assignment 1 – Basic Calculator.

The Assignment Direction:

Creating Python Programs
Part 1:
Write a Python program to find the addition and subtraction of two numbers.

Ask the user to input two numbers (num1 and num2). Given those two numbers, add them together to find the output. Also, subtract the two numbers to find the output.

Part 2:
Write a Python program to find the multiplication and division of two numbers.

Ask the user to input two numbers (num1 and num2). Given those two numbers, multiply them together to find the output. Also, divide num1/num2 to find the output.

Compile and submit your pseudocode, source code, and screenshots of the application executing the code from parts 1 and 2, the results, and GIT repository in a single document (Word is preferred).

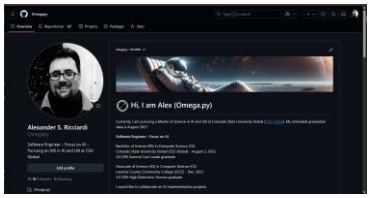
Note: Refer to the Module 1 Overview for resources and help using GIT.

My Program Description:

The program is a simple calculator consisting of two parts.
Part 1 (addition/subtraction) and Part 2 (multiplication/division).
It uses basic input validation, formats console output using box-drawing characters, and implements a menu.

Git Repository:

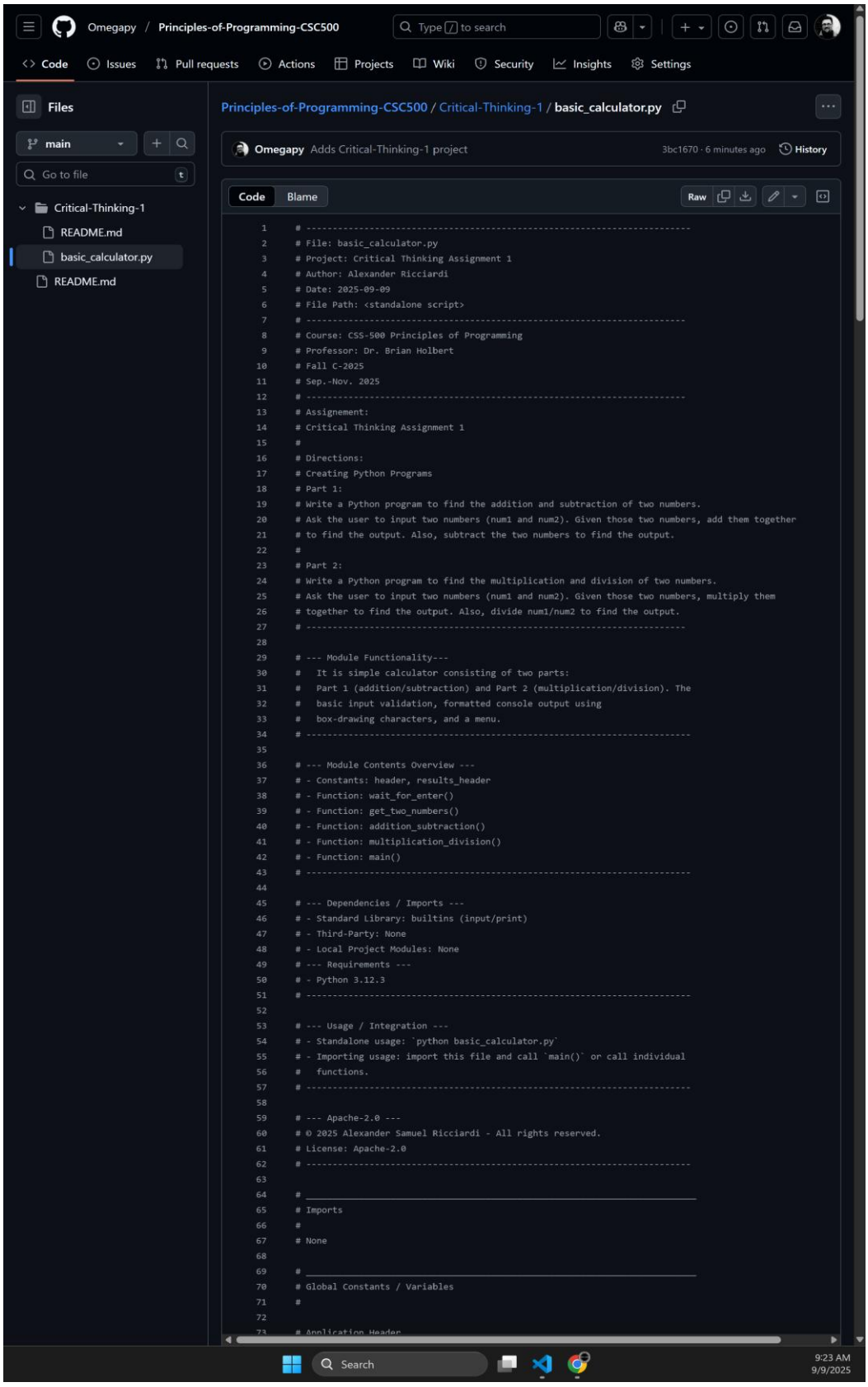
I use GitHub as my Distributed Version Control System (DVCS).
The following is a link to my GitHub profile, Omega.py.



The link to this specific assignment is: z
https://github.com/Omegapy/Principles-of-Programming-CSC500/tree/main/Critical-Thinking-1

Figure-1

Code on GitHub



Code Snippet 1

Project Code in VS Code

```
# ------------------------------------------------------------------------
# File: basic_calculator.py
# Project: Critical Thinking Assignment 1
# Author: Alexander Ricciardi
# Date: 2025-09-09
# File Path: standalone script
# ------------------------------------------------------------------------
# Course: CSS-500 Principles of Programming
# Professor: Dr. Brian Holbert
# Fall C-2025
# Sep.-Nov. 2025
# ------------------------------------------------------------------------
# Assignment:
# Critical Thinking Assignment 1
#
# Directions:
# Creating Python Programs
# Part 1:
# Write a Python program to find the addition and subtraction of two numbers.
# Ask the user to input two numbers (num1 and num2). Given those two numbers, add them together
# to find the output. Also, subtract the two numbers to find the output.
#
# Part 2:
# Write a Python program to find the multiplication and division of two numbers.
# Ask the user to input two numbers (num1 and num2). Given those two numbers, multiply them
# together to find the output. Also, divide num1/num2 to find the output.
# ------------------------------------------------------------------------

# --- Module Functionality---
# The program is a simple calculator consisting of two parts.
# Part 1 (addition/subtraction) and Part 2 (multiplication/division).
# It uses basic input validation, formats console output using box-drawing characters,
# and implements a menu.
# ------------------------------------------------------------------------

# --- Module Contents Overview ---
# - Constants: header, results_header
# - Function: wait_for_enter()
# - Function: get_two_numbers()
# - Function: addition_subtraction()
# - Function: multiplication_division()
# - Function: main()
# ------------------------------------------------------------------------

# --- Dependencies / Imports ---
# - Standard Library: builtins (input/print)
# - Third-Party: None
# - Local Project Modules: None
# --- Requirements ---
# - Python 3.12.3
# ------------------------------------------------------------------------

# --- Usage / Integration ---
# - Standalone usage: `python basic_calculator.py`
# - Importing usage: import this file and call `main()` or call individual
#   functions.
# ------------------------------------------------------------------------

# --- Apache-2.0 ---
# © 2025 Alexander Samuel Ricciardi - All rights reserved.
# License: Apache-2.0
# ------------------------------------------------------------------------
```

```
# ____________________________________________________________________________
# Imports
#
# None

# ____________________________________________________________________________
# Global Constants / Variables
#

# Application Header
header = '''
             ╔═══════════════════════════╗
             ║      Basic Calculator     ║
             ╚═══════════════════════════╝
'''
# Results Header
results_header = '''
        ┌─────────┐
        │ Results │
        └─────────┘'''

# ============================================================================
# User Prompt Utilities Functions
# ============================================================================

# ------------------------------------------------------------------------------ wait_for_enter()
def wait_for_enter() -> None:
    """Pause execution until the user presses Enter.

    This helper function allows the user time to read
    the screen before continuing.

    Args:
        None

    Returns:
        None

    Side Effects:
        Awaiting user input.
    """
    input("\n        Press Enter to continue...")
# ------------------------------------------------------------------------------ end wait_for_enter()

# ------------------------------------------------------------------------------ get_two_numbers()
def get_two_numbers() -> tuple[float, float]:
    """Prompts the user to enter 2 floating-point numbers and captures them.

    The function loops until valid numeric input are entered.
    It uses `ValueError` to check inputs.

    Args:
        None

    Returns:
        tuple[float, float]: A pair (num1, num2) entered by the user.

    Notes:
        The inputs are converted using `float(...)`.
    """

    # Get first number
```

```
    while True:
        try:
            num1 = float(input("        Enter the first number: "))
            break
        except ValueError:
            print("        Error: Please enter a valid number for the first number!\n")

    # Get second number
    while True:
        try:
            num2 = float(input("        Enter the second number: "))
            break
        except ValueError:
            print("        Error: Please enter a valid number for the second number!\n")

    return num1, num2
# ------------------------------------------------------------------------------ end get_two_numbers()


# ================================================================================
# Part 1 Operations (Addition & Subtraction)
# ================================================================================

# ------------------------------------------------------------------------------ addition_subtraction()

def addition_subtraction() -> None:
    """Part 1: Compute and display results of the addition and subtraction operations

    Workflow:
      1) Displays section header
      2) Captures two numbers using `get_two_numbers()`
      3) Computes addition and subtraction
      4) Displays results

    Args:
        None
    Returns:
        None
    Side Effects:
        Prints results to stdout and waits for user input to continue
    """
    header_part1 = '''
        ┌──────────────────────────────────────┐
        │ PART 1: Addition and Subtraction     │
        └──────────────────────────────────────┘'''
    print(header_part1)

    num1, num2 = get_two_numbers()
```

```
    # Calculate results
    addition = num1 + num2
    subtraction = num1 - num2

    # Display results
    print(results_header)
    print("        Addition:")
    print(f"         {num1} + {num2} = {addition}")
    print("        Subtraction:")
    print(f"         {num1} - {num2} = {subtraction}")

    wait_for_enter()
# ------------------------------------------------------------------------ end addition_subtraction()

# ========================================================================
# Part 2 Operations (Multiplication & Division)
# ========================================================================

# ------------------------------------------------------------------------ multiplication_division()
def multiplication_division() -> None:
    """Part 2: Computes and displays multiplication and division operations for two numbers

    Division by zero is handled by showing am 'error' descriptive message.

    Workflow:
      1) Displays section header
      2) Captures two numbers using `get_two_numbers()`
      3) Computes multiplication and division operations
      4) Displays results with a formatted results header

    Args:
        None

    Returns:
        None

    Side Effects:
        Prints results to stdout and waits for user input to continue.
    """
    header_part2 = '''
        ┌──────────────────────────────────────┐
        │ PART 2: Multiplication and Division  │
        └──────────────────────────────────────┘'''
    print(header_part2)
```

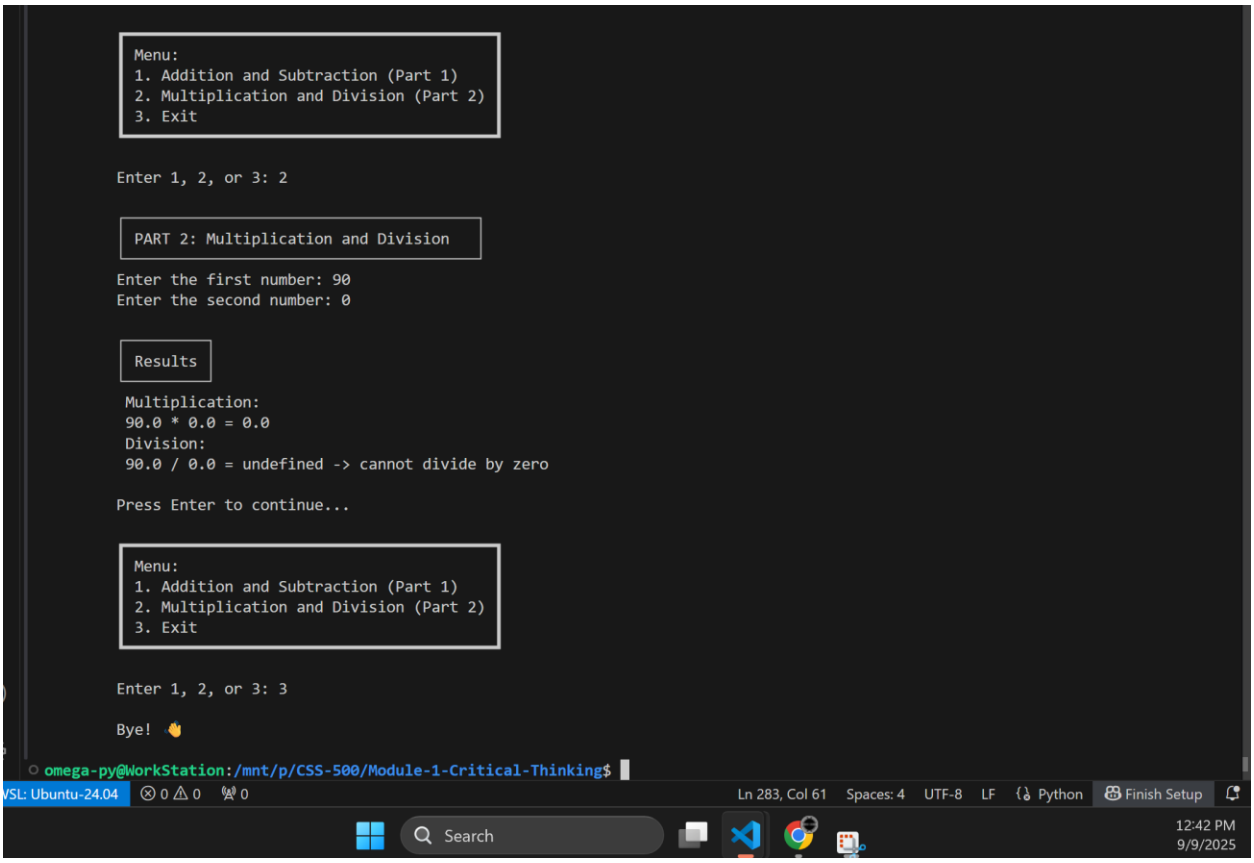
```
    num1, num2 = get_two_numbers()

    # Calculate results
    multiplication = num1 * num2

    # Handle division by zero
    if num2 != 0:
        division = num1 / num2
        division_text = str(division)
    else:
        division_text = "undefined -> cannot divide by zero"

    # Display results
    print(results_header)
    print("         Multiplication:")
    print(f"          {num1} * {num2} = {multiplication}")
    print("         Division:")
    print(f"          {num1} / {num2} = {division_text}")

    wait_for_enter()
# ------------------------------------------------------------------------------ end
multiplication_division()

# ================================================================================
# Program Entry / Menu Loop
# ================================================================================

# ------------------------------------------------------------------------------ main()
def main() -> None:
    """Entry point for the calculator's menu.

    Presents a looped menu with three choices:
      1) Part 1: Addition and Subtraction
      2) Part 2: Multiplication and Division
      3) Exit

    Args:
        None

    Returns:
        None

    Side Effects:
        Prints a menu, captures user input, and runs the operations,
```

```
        and loops until the user exits
    """
    print(header)

    while True:
        menu = '''
        ┌──────────────────────────────────────┐
        │ Menu:                                │
        │ 1. Addition and Subtraction (Part 1) │
        │ 2. Multiplication and Division (Part 2) │
        │ 3. Exit                              │
        └──────────────────────────────────────┘
        '''
        print(menu)
        choice = input("        Enter 1, 2, or 3: ").strip()

        if choice == '1':
            addition_subtraction()
        elif choice == '2':
            multiplication_division()
        elif choice == '3':
            print("\n        Bye! 👌\n")
            break
        else:
            print("\n        Error: Please enter 1, 2, or 3.")
            wait_for_enter()
# ------------------------------------------------------------------------------ end main()

# ____________________________________________________________________________
# Module Initialization / Main Execution Guard
# This codes runs only when the file is executed
# ------------------------------------------------------------------------------ main_guard
# Run the program
if __name__ == "__main__":
    main()
# ------------------------------------------------------------------------------ end main_guard
```

Continue next page

Figure 2

Project Outputs in the VS Code Terminal



As shown in Figures 2 and Code Snippet 1, the program runs without any issues, displaying the correct outputs as expected.