Overview

A program that utilizes stack operations to provide a usable calculator for basic operations with a calculator class.

Introduction

This document holds the technical design of the CSCI312\_A5\_Schmidt program and serves as a pseudo lessons learned for the changes made to the initial design. The program holds a calculator class that provides basic operations for the user.

## Scope

Holds a calculator program that performs basic operations: +, -, \*, and /.

### PROCESSING

The program creates an instance of a Calculator class and allows the user to enter basic operations in the format “digit.decimal operator digit.decimal”. When the user enters a legitimate operation, the Calculator class adds a new Momento to the CareTaker instance. A Momento holds the Momento before it so when a user UNDOs an operation, the previous Momento is pulled forward and the current Momento is thrown. CLEAR gets rid of all the Mementos the Caretaker has.

### DATA

The logical and physical data structure of files should be defined in detail.

Calculator

Holds CareTaker instance.

Holds methods to break apart the user input string and perform the operations to add to the momento.

CareTaker

Holds the Momento and handles it.

Momento

Holds the currentSolution, the runningTotal, and the previous Momento

### COMPONENTS

Calculator

Holds CareTaker instance.

Holds methods to break apart the user input string and perform the operations to add to the momento.

CareTaker

Holds the Momento and handles it.

Momento

Holds the currentSolution, the runningTotal, and the previous Momento

|  |
| --- |
| **Calculator** |
| -CareTaker theUndertaker |
| +newOperation(string): string  +operatorOperation(string):string  +wrongInput(): string  +undoOperation(decimal): string |

|  |
| --- |
| **Caretaker** |
| -Momento |
| +getMomento(): Momento  +addMomento(decimal): Momento  +undoMomento(): Momento  +clearMomento(): Momento |

|  |
| --- |
| **Momento** |
| -decimal m\_runningTotal  -decimal m\_currentSolution  -Momento m\_lastMomento |
| +Momento(decimal): Constructor  +Momento(Momento, decimal): Constructor  +getCurrentSolution(): decimal  +getLastMomento(): Momento  +getRunningTotal(): decimal  +setRunningTotal(decimal): void |

### TESTING

Present one or more named scenarios that will be utilized to test the application.

The testing plan should be repeatable.

Describe the scenario in detail, the steps required to execute the test, the input data, the output data, and the success criteria.

Present a summary of the testing scenarios before the details of each scenario.

Scenario #1- Calculator Test

|  |  |  |
| --- | --- | --- |
| Step | Input | Output |
| 1. | “4 + 6” | “Sum: 10; Running Total: 10” |
| 2. | “77 - 5” | “Difference: 72; Running Total: 82” |
| 3. | “5 \* 6” | “Product: 30; Running Total: 112” |
| 4. | “9 / 3” | “Quotient: 3; Running Total 115” |
| 5. | “4/2” | “Invalid input.” |
| 6. | “Invalid Input” | “Invalid input.” |
| 7. | UNDO | “Running Total: 112” |
| 8. | CLEAR | “Running Total: 0” |
| 9. | “99 + 9” | “Sum: 108; Running Total: 108” |
| 10. | EXIT | Program closes |
| EXPECTED OUTPUT | | To perform the operations correctly. |
| ACTUAL OUTPUT | | The program worked perfectly. |
| RESULTS – The desired output of a .txt file with the findings was produced. | | Pass |

##### 