CS 302 Project 1 Documentation

**Design Process**

At the first look at the project, I assumed I would have to do a lot more work than what was required. The first thing I did was to implement the StackLinked.cpp. As I had done a similar project in CS202 this section was simple to implement; however, I ran into trouble on the assignment operation function. I was trying to create a pointer to the new object rather than just allocate the memory for the new object straightforwardly. I have never truly implemented the throw error/exceptions before on my own. At first I thought that these required embedded amounts of code; however, I was able to implement both with a simple if statement to check if the stack was full or empty. The challenging portion of the exceptions was that my delimiters program relied on the fact that the stack was empty to determine that the expression was invalid. In the end I opted to remove the exceptions from the pop function for the sake of my delimiters program. The rest of the linked list stack implementation was decently easy.

The postfix implementation was challenging only in the sense that I had forgotten the template syntax. I reviewed my notes from previous lectures and the previous semester to finally achieve the basic call of the templated stack. In order to achieve the goals of assignment 1, I created three functions, calculatingPostfix, applyOperator, powerOperation, and strlen. The strlen function was included because I was not sure if we were allowed to use the string library, so I errored on the side of caution. The applyOperator function included a switch statement that detailed the different operations needed for each operator. Because the basic iostream library included most of the operations, it did not take very long. The challenging section was the power operator. Initially I had tried a loop in the switch that had total\*=operand1 for the duration of operand2. For some reason this method did not apply inside the switch because I had initialized the total variable to 0 for the other operations hence, I moved the operation implementation to a separate function to be able to initialize the total to 1. Once the powerOperation function was created, the operation was applied. The calculatingPostfix function was used to then apply the applyOperator function to the input from the user. The most challenging aspect of the program was to recognize what was a digit and what was a delimiter. In order to solve this problem, I found a function in the library which directly identifies if a character is a digit or not. After the digit was identified, it was important to get the correct ascii value of the digit to be pushed into the stack which is why I subtracted ‘0’ from the digit that was found.

I implemented the delimitersOk function using a switch statement to test if the brackets or parentheses matched in an expression. I ran into an error that would not let me push the expression into the stack on the basis that the argument type did not match the parameter type. I then realized that the I had made my stack a string type where I was trying to store characters. Once I made the change to a character type stack, I was able to finish the delimitersOk implementation easily. The major problem I ran into with this assignment was that the EOF condition did not end the program. The only way I could end the program was to enter Ctrl+C. Thinking back on this dilemma now, the condition for ending the program was based on if the input from the keyboard was valid or not. In order to combat this endless loop, I had to find the loop responsible for the EOF condition which I had previously missed. By changing this loop to a for loop with 5 iterations I was able to test my program and export my results to a log file.

**Exercise 1 Test Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| Trial | Infix Expression | Postfix Expression | Result |
| 1 | (4 + 2 - 3) \* 5 | 542+3-\* | 15 |
| 2 | 2^3 | 23^ | 8 |
| 3 | (4 + 2)^2 | 42+2^ | 36 |
| 4 | 4-1 | 41- | 3 |
| 5 | (2+4)/(3\*1) | 24+31\*/ | .5 |

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

**Exercise 2 Test Plan**

|  |  |  |
| --- | --- | --- |
| Trial | Delimiter Expression | Valid or Invalid? |
| 1 | ()()()()() | Valid |
| 2 | []}{}{ | Invalid |
| 3 | {}[]] | Invalid |
| 4 | ([{}]) | Valid |
| 5 | ((((( | Valid |

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated