Programming Project Report

Name: Cameron Eddy
Date: 6/28/2024

Academic Integrity Statement: I pledge that I have neither given nor received unauthorized help on this programming assignment.

Problem Statement:
The goal of this programming assignment was to implement and utilize stack and queue data structures in C++. The program includes a bracket checker using a stack and a tag-system algorithm using a queue. The inputs were strings of brackets or tag sequences, along with inputting which option the user originally wanted. The program outputs whether the brackets are balanced or the results of the tag-system algorithm. For error checking I made sure that when the user input the option of what they originally wanted the number they input was a 1, 2, or 3. I also error-checked on the tag system if they typed a character that wasn't one of the characters allowed in the tag system.

Design:
For the design of the stack, I used a linked list. With the bracket checker, I started with a for loop that would check all the characters in the string to see which ones were brackets. For the ones that were opening brackets, they were pushed onto the stack. If the character was a closing bracket, I had a switch statement to see which closing bracket matched the bracket on the top of the stack. If it didn't match then the brackets weren't balanced and I statement stating such. For the design of the queue, I used a circular array. For the tag checker, I had one for loop that would insert elements into the queue if they were supposed to be there. If not, the function would return false and they would have to retype their input. Then I had a second for loop to iterate 100 times which would iterate through the queue and insert the corresponding tag onto the back of the queue.

Implementation:
For implementation, I first developed the stack and queue classes, including their constructors, destructors, and methods. I then implemented the bracketChecker and tagChecker functions using these data structures. The main function contains a do-while loop to repeatedly present the menu and execute the chosen operation.

Testing:
The main thing needed for testing was to make sure the circular array was large enough. If not enough space was allocated for it, strange characters would be output when printing the queue. To avoid this, I changed the queue's print function to be a Boolean and check to make sure it would only output a value from the queue if it was an a, b, c, or # character. If not, the user would be told there was an error and the function would return false.

Conclusions:
The result of this assignment was a success. Implementing the circular array was hard, but all in all it probably took me 7 hours to complete the whole assignment.