**Programming Project Report**

Name: Edgar Alcocer

Date: 10/21/2022

**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 is to demonstrate my abilities in implementing a using a stack in a file processing application. The program inputs were the user’s choice to decide which file they want to interact with. There are 5 files to choose from. Each option will read an XML to be then parsed correctly and put into the stack. The programs output was to show if the program could read in the XML file onto a stack which hold the information in proper format. The error handling oversaw catching errors such as incorrect file input, this was done to check if the XML file could be formatted with the program. In addition, my menu interface has error handling for the users selection. The is possible with a do-while statement.

**Design:**

First, I needed to decide how to read the XML file. To do this I have set up a while loop which iterates until reaching the end of the file. At the same time, it is looping I have a while loop that reads until the less than symbol. Once the ‘<’ character is reached we getline() the rest of the line until we reach <. The first tasks was to create a string stack class. The second task was to implement the XML syntax checker. The third tasks will test the XML syntax checker. I will do this by modifying two of the XML example and introduce errors to locate them. The data structure used for this project was the stack abstract data type. The algorithm I used was to do nested if statements in my while loop

Stacks are ideal for enforcing sequential rules of access such as LIFO, the only con is that comparing the bottom input with the top was confusing but was manageable.

**Implementation:**

The implementation of this project was broken into three files, stringStack.h, stringStack.cpp, main.cpp. the main file will provide the initialization of the stack. The sample code I used was the stack files that were provided by Dr.Gauch, I used the skeleton of the stack and implemented code towards the solution of the project. I extended this code by adding a readfile() function which reads the .txt file to compare it to a successful XML file. The development of this project was around five days. Going tasks by tasks helped speed up the development process of the project.

**Testing:**

I tested the program by using a menu input when the program starts. I gave the user options on which file they would lie to read in. two of the files have several errors manual put into them, this was done to check the syntax error of the XML file. The normal inputs I used were implemented into the menu options so which eve option the user inputs the correct file will be read in. What were the special cases you tested?

The special case I tested was the iteration which had to read a file incorrectly. This was to demonstrate and locate the errors.

<><><><><><><><><><><><><><><><><><><>

Welcome to Project 4!!!

<><><><><><><><><><><><><><><><><><><>

1. Create stack for sample 1

2. Create stack for sample 2

3. Create stack for sample 3

4. Create stack for sample 4

5. Create stack for sample 5

6. Quit The Program

Please Select an option: 1

valid input breakfast\_menu

valid input food

valid input name

input before /name

input after

popping input name

valid input price

input before /price

input after

popping input price

valid input description

input before /description

input after

popping input description

valid input calories

input before /calories

input after

popping input calories

input before /food

input after

popping input food

valid input food

valid input name

valid input description

input before /description

input after

popping input description

valid input calories

input before /calories

input after

popping input calories

input before /food

input after

valid input food

valid input name

input before /name

input after

popping input name

valid input price

input before /price

input after

popping input price

valid input description

input before /description

input after

popping input description

valid input calories

input before /calories

input after

popping input calories

input before /food

input after

popping input food

valid input food

valid input name

valid input description

input before /description

input after

popping input description

valid input calories

input before /calories

input after

popping input calories

input before /food

input after

valid input food

valid input name

input before /name

input after

popping input name

valid input price

input before /price

input after

popping input price

valid input description

input before /description

input after

popping input description

valid input calories

input before /calories

input after

popping input calories

input before /food

input after

popping input food

input before /breakfast\_menu

input after

5

<><><><><><><><><><><><><><><><><><><>

Welcome to Project 4!!!

<><><><><><><><><><><><><><><><><><><>

1. Create stack for sample 1

2. Create stack for sample 2

3. Create stack for sample 3

4. Create stack for sample 4

5. Create stack for sample 5

6. Quit The Program

Please Select an option:

**Conclusions:**

The result of this assignment was demonstrated by the output of the different selections the user inputs. If the user selects 1-3 the stack will be correctly inputted. If the user selects 4-5 the stack will be continuously reading the incorrect format of the XML file. This project took around five days to complete. What I would do differently to improve upon this project would be to reverse this project and instead of getting the <tags> I would go for the information that is needed, there is not much to do with only the tags.