Lab Stack and Queue

Data Structures CSCI 2320

Lab Objective

Learn how to implement and use a Stack and Queue ADT in C++.

Lab Overview

You will receive a partial implementation of a Stack and Queue in C++. Your mission is to implement some of the methods for the class as well as understand all the other methods.

Lab Requirement

Verify that you are using the ISO std 20+ compiler. See Blackboard for more information.

Lab Tasks

Review the code provided. Notice that the Stack and Queue use an underlying data type of `std::list`, rather than using `Nodes` directly. This code will not compile and run because you need to implement the required methods described below.

Task 1

Implement the following methods for the Stack class in `Stack.h`:

- `int size()` This method returns the number of items in the stack.
- `bool empty()` This method returns `true` if the stack is empty and `false` if it is not empty.
- `void push(const T& value)` This method places an item at the top of the stack. **IMPORTANT:** Use the methods in the `std::list` that manipulate the front of the list (not the back) so that your iterators works correctly for a Stack.
- `void pop()` This method removes the top item in the stack. This method should throw an `out_of_range` exception. **Hint:** Use the methods in the `std::list` that manipulate the front of the list so that your iterators works correctly for a Stack.
- `T& top()` This method returns the top value in the stack. This method should throw an `out_of_range` exception.

Task 2

Implement the following methods for the Queue class in `Queue.h`:

- `int size()` This method returns the number of items in the queue.

- `bool empty()` This method returns `true` if the queue is empty and `false` if it is not empty.
- `void enqueue(const T& value)` This method places an item at the back of the queue.
- `void dequeue()` This method removes the front item in the queue. This method should throw an `out_of_range` exception.
- `T& front()` This method returns the front value in the queue. This method should throw an `out_of_range` exception.

Task 3

Implement the following in your main driver `main.cpp`.

- Print this message at the beginning of your main before any output.
  `Welcome to the Stack and Queue test program.`
- Print this message at the end of your main after all other output.
  `Goodbye!`

Final output

Your output can vary, so feel free to experiment.

Rubric

Name	Description	Points
AutoTest Setup	Install test code, copy student source, build student and test code	0
Main Output	Run student main and compare output to test	10
Coding Style	Run cpplint on student code	4
Stack Empty		1
Stack Size		1
Stack Top		5
Stack Top Empty Stack		5
Stack Push		10
Stack Pop		10
Stack Pop Empty Stack		5
Stack ToString		1
Stack Print		1

Stack Copy Constructor		1
Stack SaveRestore		3
Queue Empty		1
Queue Size		1
Queue Front		5
Queue Front Empty Queue		5
Queue Enqueue		10
Queue Dequeue		10
Queue Dequeue Empty		5
Queue ToString		1
Queue Print		1
Queue Copy Constructor		1
Queue SaveRestore		3
Total Points		100

Due Dates and Honor

The due date is specified on Blackboard.

This is an *independent* programming project, and it is very important that you understand and abide by the *academic integrity policy* concerning programming projects. Remember, your personal honor and integrity is far more important than your grade on the project.

Grading

This lab is available in GitHub Classroom. Accept the URL on Blackboard and then clone your repository to your machine for development. Your lab will be graded automatically via GitHub. Please check the grading results each time you check in your code. Your final grade will be based upon your last sync to GitHub before the deadline.

Project Artifacts

The following should be completed by the due date/time specified on Blackboard.

- Check in all source code changes to your GitHub repository. Please check your URL using a web browser to verify that your changes have been synced.

- Submit the URL for your repository to Blackboard.