Lab 9

Goals-
Implement linear data structures using link lists

You will create two data structures a stack and a queue. You will not use any existing containers, such as the STL. You will use a linked linear structure to store the values and provide the necessary functions to interact with that linked structure.

For both you will use a singly linked list to store the data. You may also use pointer variables to the element/node that is at the beginning of the structure, the end of the structure, or both. You will not use a pointer to iterate or go to other individual elements of the structure. In both cases removing is restricted to just one end; either the first or the last.

STACK

The stack is a first in last out structure. You add to the top and can only look at or take off the top. You only need a singly linked list to implement it. (Consider why that is) You only need to use a simple data type, such as `int`, for this lab. You will create these functions, with appropriate parameters:

`void push()`: puts on item onto the structure
`int peek()`: returns the value on top of the structure
`void pop()`: removes the top item in the structure
`isEmpty()`: For this lab you only need this check when you pop a value.

QUEUE

The stack is a first in first out structure. You add to the back and can only take off the front. For the queue sit down with pencil and paper and study the required pointer manipulations. You only have the link(s) in each node, a pointer to the front and a pointer to the back. You can use a singly linked structure. Have your pointer algorithms written out before you start writing your code! You only need to use a simple data type, such as `int`, for this lab. You will create these functions, with appropriate parameters:

`void addBack()`: puts on item at the end of the structure
`int getFront()`: returns the value at the front of the structure
`void removeFront()`: removes the first item in the structure
`isEmpty()`: For this lab you only need this check when you remove a value.

You must also write a program that uses your stack and queue to demonstrate they work correctly. You should prompt the user to enter a series of integers, with some way to indicate they have finished entering values. Just enter the each number into your stack and queue. Then you print out the values as you remove them from your structure. Make sure you label the output as to which is coming from the stack or the queue.

What to submit-

You will submit the following files to TEACH-

Code to implement your stack, both header and source files

Code to implement your queue, both header and source files

Code to demonstrate the operation of your stack and queue.

HINT: Please do the stack first. The pointer manipulation is easier and more obvious. Then use that experience to develop the pointer algorithms for the queue and then write that code.

Modular Grading

We are using modular grading. Each lab will be divided into specific modules. Each module will be graded pass/fail. It either works properly or it does not. 10% of every lab or assignment grade is style/comments or other elements of self-documenting code and clarity. Remember the labs are worth 10 points total.

Programming style- 1 point

Code to implement your stack
- header file- 1 point
- source file- 3 points

Code to implement your queue
- header file- 1 point
- source file- 3 points

Code to demonstrate the operation of your stack and queue- 1 points