Lab6 - CIT315

Goals

The primary goal of this project is to use linked lists to build stack and queue data structures. The previous lab tasked you with the construction of the single linked list with all of its structure and functions. The second part is the use of the linked list to construct *stacks* and *queues*. The stacks and queues will then be used to manipulate data.

Stack

Develop a *stack* data structure, using your Lab 5 linked list, with the following functions (keep in mind that these structures are a subset of the linked list structure)

- Push - pushes a data node onto the top of the stack. This function will need a parameter of the data type stored in the stack.
- Pop - Takes the top node off the stack and returns the data to the program.
- Size - returns the size of the stack in terms of the number of nodes.
- Empty - returns a boolean, true if the stack has no nodes and false if the stack has greater than 0 nodes.

Queue

Develop a *queue* data structure using the Lab 5 linked list, with the following functions:

- Enqueue - adds a data node onto the back of the queue. This function will need a parameter of the data type stored in the queue.
- Dequeue - Takes the front node from the queue and returns the data to the program.
- Size - returns the size of the queue in terms of the number of nodes.
- Empty - returns a boolean, true if the queue has no nodes and false if the queue has greater than 0 nodes.

The description for each of the stack and queue functions are given in the lecture. There must be a Node Struct containing data and a pointer. In this case, the data will be a Struct with the following data: *First Name*, *Last Name*, *PUID* and *Age*. You will need a *start* pointer to maintain the head of the list and a *current* pointer when traversing the list. Hint - think about the order to write the functions, as some are dependent on others!

To use the stack and queue, you will develop a simple text-based interface. The interface will ask a user to input data for a Struct First Name, Last Name, PUID, Age. Once you get the 4 data elements, you add them to a Struct. Once they are added to a Struct, you will add the Struct to both a stack and a queue. During each iteration, the user will have the option to:

- Pop - delete a node from the Stack
- Push - add a node to the Stack
- Enqueue - add a node to the Queue
- Dequeue - delete a node from the queue
- Empty Queue - remove all of the nodes from the queue
- Empty Stack - remove all of the nodes from the stack
- Print Queue - print the nodes of the Queue in the order of arrival into the queue
- Print Stack - print the nodes of the Stack, which should be the reverse of the Queue.
- Reverse Queue - Use the Stack to reverse the order of the elements in the Queue, by pushing them on and then popping them in reverse order. This will be the test if your stack and queue structures work.
- Exit - exit the program.

Scoring

- Correct implementation of interface - 2 points
- Correct implementation of Stack functions - 10 points
- Correct implementation of Queue functions - 10 points
- Reverse of Queue using Stack
- Documentation - 1 points

Submission

Complete the lab and submit the project file(s) to Brightspace by the required date and time.