CS 1410 Introduction to Computer Science - CS2
Assignment #9
Total Points: 50 points

Problem description

Your task is to create a template Linked List base class and a set of specific classes that inherit from it. In addition to the classes, you will implement a set of tests in your main to prove the functionality of the classes.

Task 1 (5 points)

In this task you will implement the template Linked List node class and associated functionality.

- Implement the LinkedListNode class
  - At least the following **private** variables must be included
    - A pointer to a generic data member
    - A next pointer to the next node in the linked list
  - Add any needed accessors and function

Task 2 (15 points)

In this task you will implement the template Linked List class. This class will implement all of the Linked List parsing, inserting, deleting, etc.

- Implement the LinkedList class
  - At least the following private functions must be included
    - A pointer to the head of the list. It will be a LinkedListNode pointer.
    - A count of the total nodes in the list
  - Implement the following protected functions
    - An ‘int insert(T* pItem, int pos = -1)’ function for adding to the list. The function should add ‘pItem’ to the list at position ‘pos’. The position intidicates where in the list the item is. A position of 0 indicates the item is at the head of the list. If a negative position, or a position greater than the current list size, is passed into the function, then the item should be placed at the end of the list. The function will always return the position of the added item.
    - An ‘LinkedListNode* at(int pos)’ function for retrieving a node at a specific position.
    - A ‘T* removeAt(int pos)’ function for removing a node at a specific position. The function will find the node, delete the node , and return the generic data it stored.
    - A ‘LinkedListNode* find(T* pItem)’ function for finding a specific item in the list. It should search the list and return the node if found.
    - a ‘clear()’ function for emptying the list. The function must use recursion. This function must properly delete all memory.
  - Implement the following public functions

        - The ‘getSize()’ function for obtaining the number of nodes in the list
        - The ‘begin()’ function for getting a pointer to the head of the list
        - A T* at(int pos)’ function for retrieving a node at a specific position.
        - A ‘operator++’ function for iteration. This function will update the node pointer to the ‘next’ variable.
    - Implement a destructor that will properly clean up all memory when the
    - structure is destructed.
    - You may add other functions and variables as needed

Task 3 (10 points)

In this task you will implement the template Queue class which inherits from the LinkedList class. A Queue is a First In First Out container. Think of the line a grocery store. The first item in should be the first item removed.

- Implement the Queue class
    - Create the following public functions
        - A ‘T* pop()’ function. It will remove the first item in the list and return it.
        - A virtual ‘void push(T* pItem)’ function. It will add an element to the end of the list
    - You may add other functions and variables as needed

Task 4 (5 points)

In this task you will implement the template OrderedQueue class. This class should inherit from the Queue class. The class is similar to a Queue except that items are inserted in order. and removed in order. For example, if 1, 2, then 3 were inserted, then the removal process would result in 1, 2, then 3. If 1, 3, then 2 was inserted the removal process would result in 1, 2, then 3 because 2 would have been inserted before 3.

- Implement the OrderedQueue class
    - Create the following public functions
        - Implement the ‘void push(T* pItem)’ function. It will push the newly added item into the appropriate location in the queue. It will use the < operator for comparison.
    - You may add any other functions and variable as needed

Task 5 (10 points)

In this task you will implement a template Stack class. This class should inherit from the LinkedList class. You will remember from the previous assignment that a Stack is a First In Last Out container. For example, if 1, 2, then 3 were inserted then the removal process would result in 3, 2, then 1.

- Implement the Stack class
    - Create the following public functions
        - Implement the ‘void push(T* pItem)’ function. It will push an item onto the top of the stack.
        - Implement the ‘T* pop()’ function. It will remove an item from the top of the stack.
        - Implement a ‘T* top()’ function. It will return a pointer to the item at the top of the list, but will NOT remove it from the list

Task 6 (5 points)

In this task you will create an instance of each class type and write code that tests each of the functions.

- Test the Queue class
- Test the Ordered Queue class
- Test the Stack class

Extra Credit (5 points)

For the extra credit you must implement all of the tasks above, add a previous pointer to the node class, and update all classes to support backwards parsing.

- Update the LinkedListNode class
    - Add a private previous pointer
- Update the LinkedList class
    - Add a private tail pointer for reverse iteration
    - Create a ‘end’ function for getting the last node in the list
    - Modify the insert function to use the tail pointer for insertion at the end of the list