Assignment 8: Linked List

EE312 – The University of Texas at Austin – Fall 2016

Assigned: Tuesday, Nov 15th
Due: Tuesday, Nov 22nd, before 23:59

Deliverables

This assignment focuses on Classes. You will be implementing a LinkedList class.
You are provided with two starter files: main.cpp and linkedlist.cpp
The main.cpp files contains test cases for your LinkedList class implementation.
The linkedlist.cpp file should contain your LinkedList implementation. Initially, the linkedlist.cpp file is empty. That should give you flexibility in your design.

Once you have completed the LinkedList class implementation, you can run main to make sure you pass all test cases given to you.
Submit your main.cpp and linkedlist.cpp files to Canvas.
Note: The main.cpp file should only include test cases. Feel free to add test cases to main.cpp to further test your implementation.
The linkedlist.cpp file will be uniquely tested, so make sure the LinkedList class implementation goes into that file.

Note: If you are using a linkedlist.hh you need to post that to Canvas too.

Implementation

Your LinkedList class implementation needs to support the following functions. Each node in the Linked List has a int value and a pointer to the next node. The Linked List is 0 indexed, meaning that the first node is at position (index) 0.

1. `addToListAtPosition` (int position, int value)
   The addToList add the value to the Linked List at the specified position. If the position is greater than the length of the string, the function does nothing to the Linked List. For example, when adding at position 3, the node that was previously at position 3 is pushed to position 4.

2. `addToHead` (int value)
   The addToHead function adds the value to the head of the list.

3. `appendToList` (int value)
   The appendToList function adds the given value to the end of the Linked List.

4. `removeFromListAtPosition` (int position)
   The removeFromListAtPosition removes the value from the Linked List at the specified position. If the position is greater than the length of the list, the function does nothing to the Linked List.

5. `removeFromHead` ()

The removeFromHead function removes the value at the head of the Linked List. If the list is empty, the function does nothing to the Linked List.

6. `removeFromEnd` ()
   The removeFromEnd function removes the value at the end of the Linked List. If the list is empty, the function does nothing to the Linked List.

7. `find` (int value)
   The find function returns true if value is inside the LinkedList and false otherwise.

8. `reverseList` ()
   The function reverses the Linked List. For example, given the Linked List 1->2->3, the list becomes 3->2->1

9. `getListLength` ()
   The function returns the length of the LinkedList.

You are provided with 11 test cases in the main.cpp files. Use these test cases to test your LinkedList implementation. Feel free to add more test cases to further test your implementation. Use the same function names given in main.cpp for your implementation.