Programming Assignment #5

Linked List of string

CS 2308.001 and 002 Fall 2016
Instructor: Jill Seaman

Due: Thursday, 11/17/2016: upload electronic copy by 11:55pm

Problem: Implement an interface that manipulates a list of strings. You will be provided with the following files on the class website:

- **StringList.h** containing a class declaration, set up for a linked list representation.
- **Driver.cpp** containing a main function you can use to test your implementation.

You will be responsible for providing the StringList.cpp file, including the implementation of the StringList member functions (described below):

StringList and **~StringList**: creates an empty list, and deallocates all the nodes in the list, respectively.

count: returns the total number of strings (nodes) in the list. Any duplicate strings should be counted.

add(string) Adds a new node containing the string to either the beginning OR the end of the list (your choice).

remove(string) removes a node containing the given string from the linked list. Returns true if successful, otherwise false (if the string was not in the list).

display(): displays the strings in the list to the screen, <u>one string per line</u>.

lookup(string): returns true if the given string is in the list, otherwise false.

sort(): Here is the algorithm you **must** use for implementing the sort function:

1. Define a StringNode * to be the head of a new list (make it the empty list).
2. Repeat for every node in the original list:
   a. Remove the first node from the **original** list (see T9).
   b. Insert this node into the proper position in the **new** list:
      i. Find the node before and after the insertion point in the new list (see T7) —stop at the first node larger than the one you need to insert.
      ii. Modify the new list to insert the old node into the new list (see T11)
3. make the old head pointer (now empty) point to the new list!

Input/Output:

Use the provided Driver.cpp file to test your code. I recommend trying to implement one or two functions at a time, and testing them, rather than implementing all the functions and then trying to debug them all at once.

NOTES:

- This program must be done in a **Linux or Unix** environment, using a command line compiler like g++. Do not use codeblocks, eclipse, or Xcode!
- Put your code in a file named **StringList.cpp.**
- Your StringList.cpp file **must compile** with the (unchanged) provided files, otherwise you may receive a score of 0.
- Your program must pass **Test Case 0** or you will receive a score of 30 or less with no credit for the other grading categories (correctness/constraints/style). This test case is in a driver file called **TC0Driver.cpp** on the class website. It is similar to the Driver.cpp test driver, but it is much shorter. Your code must compile with this driver and produce the expected output (see the comments in the file). Your program must implement a linked list to pass TC0.

Logistics:

For this assignment you need to submit only the **StringList.cpp** file. You do not need a zip file, you do not need a makefile, you do not need to provide your driver.

There are two steps to the turn-in process:

1. Submit an electronic copy using the Assignments tool on the TRACS website for this class.
2. Submit a printout of the source file at the beginning of class, the next class day after the assignment is due. Please **print your name on top of the front page**, and staple if there is more than one page.

Note: Each member of a group must submit their own electronic copy and their own printout!! Make sure your name is written or circled on your printout.

See the assignment turn-in policy on the course website (cs.txstate.edu/~js236/cs2308) for more details.