Programming Assignment #2

1. **This program tests the concepts of:**
   - Containers
   - Dynamic Memory
   - Linked Lists
   - Reading chapter 5

2. **Program Objective:**
   Implement and test a custom string class, help can be found in Section 4.5 of the textbook. The class should be implemented by yourself (though you may talk with other students). An non-interactive test program is provided for you (stringexam.cpp). This test program does not completely test your work (for example, it does not test the getline function).

   Submitted files need to be named as follows:

File	Format	Example
StringList implementation	lastname_stringlist.cpp	miller_stringlist.cpp
Additional testing driver (you do not need to hand this in since I will be using my own stringexam)	lastname_stringexam.cpp	miler_stringexam.cpp

   *** Your submission may not compile for me if your files are not named as expected. ***
   Refrain from editing and submitting stringlist.h

3. **Description:**
   In project 1 you created mystring which was based on a dynamic array. With this project you are to base your string implementation on a linked list. You can use either a single or doubly linked list and choose to use a tail pointer or not. However, you want to ensure that your algorithms are efficient. With mystring you had to be sure to end your string with the null character, with a linked list this is no longer important since you will always know when you are at the end when you reach the null pointer. However, this also means that most of the cstring functions will no longer be efficient enough to use.

4. **Output Layout:** Your stringlist container should NOT produce any output.

5. **Other:**

   - For full credit your code should have zero compiler warnings.
   - Your code will be checked for memory leaks. Memory leaks will be verified via a tool called “valgrind” as well as visually.
   - Your linked list must be of nodes, utilizing the provided unedited node1d.h/cpp files. It is recommended you utilize the linked toolkit functions when appropriate, however, this is not required.
   - Usage of any STL containers will affect grade detrimentally.
   - Usage of recursion will result in no credit
   - Avoid all STL items except std::copy (algorithm), std::swap (algorithm or utility), assert (cassert), size_t. In addition, you may use functions from the following libraries: cstring, cctype, cstdlib, and cassert.
   - Only submit the the files listed in #2, submitting object files, IDE project files, and executables will impact grade.