CS 0445 Fall 2020 Assignment 2
Part One

Online: Friday, September 18, 2020
Due: All source files for plus any data and write-up files for both Part One and Part Two (see Assignment 2 Part Two for details) plus a completed Assignment Information Sheet zipped into a single .zip file and submitted to the proper directory in the submission site by **11:59PM on Sunday, October 4, 2020**
Late Due Date: 11:59PM on Tuesday, October 6, 2020

Purpose: Now that we have discussed linked lists, it is a good idea to get some practical experience working with them. In this assignment you will implement a commonly used type in a somewhat uncommon way – using a linked list.

Goal: You should be familiar with the **StringBuilder** class in Java[1]. Recall that the String class in Java produces immutable objects, or, in plain terms, strings that are fixed once they have been created. This can be inefficient if operations such as appending to the end or inserting into the middle of a string are necessary. The StringBuilder class, on the other hand, allows for strings that can be modified. The predefined StringBuilder class is implemented with an underlying array of characters. In this project, you will implement your own class, called MyStringBuilder, which is similar in functionality to the standard StringBuilder, but is implemented using a **linked list of characters** rather than an array. You will then test the performance of your class against the Java predefined StringBuilder and String classes in a simple simulation (see Part Two of this assignment).

Details:

I have provided the specifications for the MyStringBuilder class in the file MyStringBuilder.java. **Read these specifications over very carefully, paying particular attention to the comments**. Note that your data must be as specified (you **may not** add any additional instance variables) and you must implement your class using your own linked list with the inner class CNode as specified. **You may not use any predefined linked list within your class, you may not use any predefined Java String-ish class (such as StringBuilder or StringBuffer) anywhere in your code, and you may not copy any code from the Internet**. Furthermore, your methods **must act directly on the linked list** (so, for example, you **may not** convert your MyStringBuilder to an array of chars and then perform the method on that array). Some of the methods will be relatively simple to implement while some will be more complicated. I recommend working on the methods one at a time, only proceeding to the next one when you are sure that the previous one works correctly. This way, any compilation or logic errors that occur will be fairly easy to locate and will be less difficult to fix. For most of the methods, be very careful to **handle special cases**, as we discussed in lecture.

To get you started, **I have implemented one of the constructors and the toString() method for you** (see file A2Help.java). Recitation Exercise 4 will also be very helpful – it is a different class but once you understand how the copy constructor and equals() method for the LList<T> class work, you should better be able to implement some of the MyStringBuilder methods.

After you have completed your MyStringBuilder class, you will test it in two ways:

1) You will compile and run the program **Assig2.java**. This program is provided to you and **must be used as is.** The idea is similar to that of the test files for Assignment 1 – the program is a simple driver that tests most of the functionality of the MyStringBuilder class. Assig2.java will definitely test special cases of your operations, so be sure that these are handled. The output to this program is shown in file A2Out.txt. Your output should be **identical** to the data in this file.
2) You will complete a simple simulation to compare the times required for three operations: `append(char c)`, `insert(int loc, char c)` and `delete(int start, int stop)`. **For more details on this simulation, see the specifications in Assignment 2 Part Two.**

[1] If you are unfamiliar with StringBuilder, I suggest looking it up in the Java API and in your CS 0401 textbook

Make sure you **submit ALL** of the following to get full credit:

1) All of the files that you wrote, well-formatted and reasonably commented:
   a) MyStringBuilder.java
   b) Assig2B.java [your test program for part 2) above]
   c) Any other Java files that you may have also written
2) The test driver program provided to you, unchanged from how you downloaded it:
   a) Assig2.java
3) Your Assig2 Part Two write-up document in one of the specified formats.
4) Your completed Assignment Information Sheet. Be sure to help the TA with grading by clearly indicating here what you did and did not get working for the project.

The above files should be zipped up into a **single zip file** that you will submit for the assignment. As with Assignment 1, the idea from your submission is that your TA can compile and run your programs WITHOUT ANY additional files, so be sure to test them thoroughly before submitting them. If you cannot get the programs working as given, clearly indicate any changes you made and clearly indicate why (ex: "I could not get the indexOf() method to work, so I eliminated code that tested it") on your Assignment Information Sheet [one test the TA may do is to compile your MyStringBuilder.java file with a different copy of Assig2.java from the one you submitted – so if you modified Assig2.java in any way in order to get your program to work, it is essential that you state as much in your Assignment Information Sheet].

Extra Credit Ideas: If you want to get some extra credit, here are two ideas:

1) Implement some additional, non-trivial methods in your MyStringBuilder class. Look up StringBuilder in the Java API, choose one or more interesting methods that are not already required, and add them to your class. Alternatively, you could make up an interesting / useful method that is not already in the StringBuilder API. If you do this option, do not change the Assig2.java driver program. Rather, write a second driver to demonstrate your extra credit methods.
2) Add some additional test cases to Assig2B.java to see how MyStringBuilder compares to StringBuilder and String in those cases. If you do this you should also do the analysis of these additional results.