CS2383 – Fall 2024
Assignment 2 – Review of linked lists, stacks, and queues

Due: Tuesday Sept. 24, 10am (class time)

IMPORTANT: individual work please!

Tasks:

Part 1:

Let's assume that we have the following code inside the SimpleLinkedList class, and that we are calling it on the head of the list. What would this code do? Show what would happen (trace the execution) on an example. The list you are using as an example should contain at least 4 elements.

```
void printMystery(SimpleNode n)
{
        if (n == null) return;
        printMystery(n.getNext());
        System.out.println(n.getStr());
}
```

Part 2:

Do the Exercise 1.3.4 in the textbook. You can use the LinkedList class and its methods "push" and "pop" as your stack. Note: you program should read the whole string as input, and then loop through it one character at a time. Convert the string into an array of characters first, so as to be able to easily access each character using an index (see the String class…it does have a method for doing this). The input string can contain any character (e.g., "(agr[9 q]%%)we"), but your code should only look at balancing "(" and ")", "[" and "]", and "{" and "}". As soon as an imbalance is found, stop going through the input string and print an error message.

<see next page for the last part of this assignment>

Part 3:

Download the set of classes related to the implementation of a Doubly-Linked list of Strings including Iterators. Add a method with the following signature:

void multiDelete(String str, int n)

This method should first search for this string (you can call existing methods for this). You should remove this element from the list as well as the next n-1 elements after it. For example:

Assuming a list representing the alphabet (i.e., [“A”, “B”, “C”, …, “Z”]),
Calling multiDelete(“D”, 21) would result in the list [“A”, “B”, “C”, “Y”, “Z”]

Important:

- Your code should not remove each element one at a time. If you visualize the deleted portion as its own list, you should only break the links at its head and at its tail. No marks given if you are not doing so.
- Make sure you cover all cases (i.e., deleting at the head of at the tail). If deleting n strings gets you passed the last element in the list (e.g., multiDelete(“X”, 6) in the example above), simply delete all elements that you can and indicate the problem using a printed error message.
- If the string is not even part of the original list, print an error message saying so.

What to submit: a paper copy of your work (printed code for both Part 2 and Part 3, hand-written diagram for the program trace in Part 1). Also submit your Java files through D2L (marking will be involve running your code).