Jared Nielson
Assignment 7 Analysis
CS 2420

1. Have you worked with more than one partner yet? Remember, you are required to switch at least once this semester.

I have not worked with more than one partner this semester. I have a new partner for the next pair programming assignment though.

2. In the LinkedListStack class, the stack data structure is implemented using a doubly-linked list. Would it be better to use a singly-linked list instead? Defend your answer.

One of benefit of a doubly-linked list over a singly-linked list is that there is constant time access to the tail of the list. Adding to and removing from the tail of the list is much more efficient than a singly-linked list. The downside to this is that the implementation is of a doubly-linked list is slightly more complex. There is also slightly more memory overhead as each node requires pointers to two other nodes.

Because of the nature of a stack and the first in first out paradigm access to the front or head of the list is all that is required. The tail does not need to be, nor should it be accessed. This means that the benefits of a doubly-linked list are not needed when implementing a stack. Because of this a singly linked list would be better.

3. Would it be possible to replace the instance of DoublyLinkedList in the LinkedListStack class with an instance of Java's LinkedList? Why or why not?

Yes it would be possible. Java's LinkedList class contains all of the functionality of the DoublyLinkedList. In fact the method names used are identical so there would be minimal refactoring.

4. Comment on the efficiency of your time spent developing the LinkedListStack class.

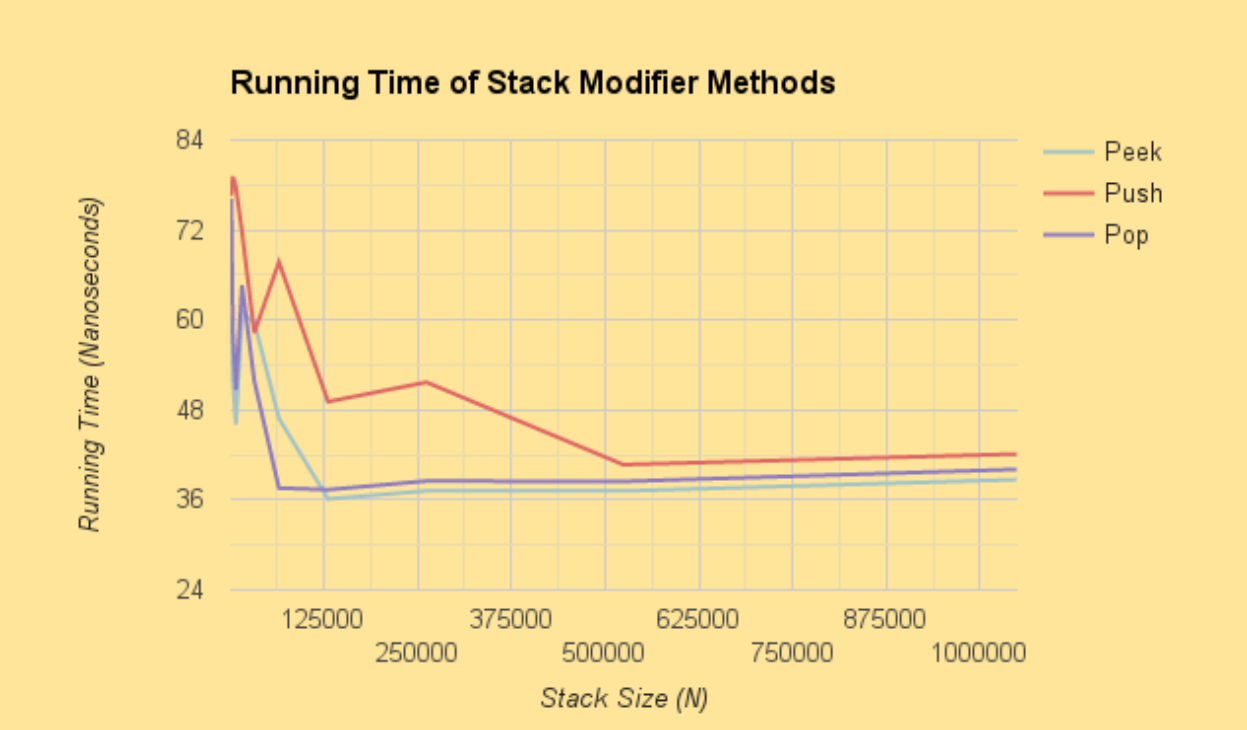
Implementing the LinkedListStack class took less than 15 minutes. The majority of the work was already done in the implementation of the LinkedList class. All that needed to be done in the push, pop and peek methods were calls to addFirst, removeFirst and getFirst respectively. Other implementation details like size and clear made calls to the same methods in the LinkedList class.

5. Note that the line and column number given by BalancedSymbolChecker indicate the location in a file where an unmatched symbol is detected (i.e., where the closing symbol is expected). Explain how you would also keep track of the line and column number of the unmatched opening symbol. For example, in Class1.java, the unmatched symbol is detected at line 6 and column 1, but the original '(' is located at line 2 and column 24.

To keep track of the column number and and line number of unmatched opening symbols two additional stacks could be used. There could be a stack called openingSymbolCol and another called openingSymbolLine. When an opening symbol is encountered and pushed onto the symbol stack it's line number is pushed on to the openingSymbolLine stack and it's column number is pushed on to the openingSymbolCol stack. When a matching closing symbol is encountered and the relevant opening symbol is popped of the symbol stack the top elements of the other two stacks are popped off as well. If an unmatched symbol is encountered the top elements in the row and column stacks are the row and column numbers of the unmatched opening symbol.

6. Collect and plot running times in order to determine if the running times of the LinkedListStack methods push, pop, and peek are O(1) as expected.

As evidenced by the chart below, the run times for push, peek and pop are all O(c) time. There is some noise at the left end of the graph, but overall it shows that regardless of the size of the stack the run time of the method is the same.



7. How many hours did you spend on this assignment?

I spent approximately 8 hours on this assignment.