605.202: Introduction to Data Structures

Omar Ismail

Lab2 Analysis

Due Date: March 23nd, 2021

Dated Turned In: March 23nd, 2021

Lab2 Analysis

Recursively converting prefix expressions to postfix expressions was quite a challenge. My program has two main components: user input/output through .txt file system and the conversion of prefix to postfix expression, which was done recursively.

The conversion of the expression was much more difficult to code than I expected. In our last lab, we used stacks which are fairly simple to understand. This time, utilizing recursion allowed us to convert the expression without storing it anywhere, except for the output file. I would say that this method allows us to use the program in more instances. It is also way more efficient in terms of space and run-time.

Looking back at lab1, I think that a recursive solution was harder to manifest but it's a lot more efficient in terms of memory and run-time. The stack solution in lab1 worked really well and it was quite easy to code; however, I would say that a recursive solution just makes more sense. I definitely feel different than I did in lab1. After creating my own stack and utilizing it in lab1, I thought that stacks was the preferred method because it was easy to understand. The recursive solution is harder to understand but it works just as well.

Due to the requirements of this project, we could not use the built in ArrayList library (java.util.ArrayList) which is essentially a resizable array list with built in commands that remove data points and add other data points very easily. Because I could not use this library, I found that the trick was to have return statements inside the if statements and to utilize substring to change the original prefix expression.

What I Learned

This project was quite challenging in terms of understanding how to filter through the expression. It was different in that we were not storing data in certain variables; rather, we were converting as we read. It took a lot of trial and error and I even considered making my own ArrayList library to use to store the data in, but thankfully I got the code working before I had to resort to that.

What I might do differently Next Time

Something that I didn't do at the start of the project that I wish I had done was fully understand how to recursively solve this problem, with a pencil and paper. I struggled early on because I was not clear on how to solve the problem, so I found myself wasting a lot of time trying things that were not going to work. Once I had solved it out on paper, I really understood what steps I need to take in order to find the recursive solution.

Justification for Design Decisions

There are two classes in my program. The main entry point outlines the solution of converting prefix to postfix expressions without showing the details of the conversion. This allows for a cleaner code.

The PrefixToPostfixRec class is the conversion algorithm that I created in order to convert from prefix to postfix recursively. If you compare it to lab1, it is so much less code in terms of lines.