Lab 6: Queue & Shunting-yard Algorithm

Data Structures and Algorithms

This lab is to have you implement Queue together with the Shunting-yard algorithm and the algorithm for computing RPN

Your first task: Implement Queue with the following methods

- void enqueue(int d)
- int dequeue()
- int front()
- boolean isFull()
- boolean isEmpty()
- String toString()

You must implement both array and linked list version of the class Queue.

Now, let compute infix expression.

To do so, you have to read from stand input. Then, process the input one token at a time using StringTokenizer. After that, you will have a queue of String as your output.

With the queue of String, you can feed it as an input to your RPN computation program. So, you need to modify your previous RPN program to take a queue as an input rather than StringTokenizer. Other than that, everything should work together nicely.

Your second task: Implement a program that take infix expression and output the result of the calculation. You must implement this using Shunting-yard algorithm and RPN calculation.

Hand in your work in MS Team assignment by summitting MyQueueA.java, MyQueueL.java, and ComputeInfix.java. If your ComputeInfix.java makes use of any other classes from other files, please also include those files in submission too.