**CSC220 Assignment07**

**Stacks**

The goal of this week’s assignment is:

1. Practice using stacks

2. Learn about the importance of debugging

**Things you must do:**

1. There are many details in this assignment. Make sure you read the whole thing carefully before writing any code and closely follow this instruction.

2. You must complete your assignment **individually**.

3. Always remember Java is case sensitive.

4. Your file names, class names, and package name must match exactly as they are specified here.

**5.** **Your project must include the methods you implemented in the lab.**

**Things you must not do:**

1. You must not change the file names, class names, package names.

2. You must not change the signature of any of these methods (name, parameters,

…).

**Part 0**

• You first must make sure that you have already finished the lab successfully and have all methods in the lab instruction working properly.

• **DO NOT** start your assignment unless you have all the features in the labworking.

**Part 1 – The problem description**

For this assignment, you are asked to write a method that accepts a fully parenthesized infix expression and returns a string representing an equivalent postfix expression. As we saw during the lecture, postfix expression are ones where each operator follows its two. For example, 12- is postfix representation of 1-2.

**Do not continue unless you have finished the lab.**

**Part 2 – toPostfix method**

Your method should accept a string (a fully parenthesized infix expression) and have the following signature:

**public static** String toPostfix(Stringline)

toPostfix uses the StringSplitter you worked with during the lab to tokenize the input String that contains the infix expression, and then follow the algorithm below to convert it into a postfix representation. In order to accomplish this task, we are still going to use stack. In this case, we only need one stack to store the operators (you no longer need a stack to store the operands).

You already should have learned how to use StringSplitter and how to declare your operator stack in the lab instruction. The helper methods you implemented during lab will prove very helpful!

Here is the pseudo-code of the algorithm you have been asked to implement for your assignment:

1. Scan the input string (infix notation) from left to right (one pass)

2. If the current token is an operand

2.1. append it to the postfix string.

3. If the current token is an operator (call it *current operator*).

3.1. Pop from the operator stack and append to the postfix string every operator on the stack that

3.1.1. is above the most recently scanned left parenthesis, and

3.1.2. has precedence higher than to that of the current operator

3.2. Push the current operator onto the stack.

4. If the current token is a left parenthesis 4.1. push it onto the stack.

5. If the current token is a right parenthesis

5.1. Pop all operators down to the most recently scanned left parenthesis and append them to the postfix string.

5.2. Pop the corresponding left parenthesis and discard this pair of parentheses

Before start writing your code, grab a piece of paper and apply the algorithm on the following expression: "((4+5)\*6)". The output should be: "45+6\*". Follow the algorithm ***exactly!***

**Part 3 – Test your code**

After you are done implementing your toPostfix, you need to test it. We have provided different test scenarios for you in the main method of the class. You should uncomment these tests.

Run your code, if you see any red text that says “test failed”, you need to debug your code. How to debug your code?

1. Use the Eclipse debugger you learned about during the first lab

2. Think about writing helper method that help you inspect the status of your stacks, etc.

3. Go back to your pen/paper example and follow your code to see if indeed it does what it is supposed to.

4. There are other ways to debug your code as well.

**Remarks**

* Make sure to submit your assignment by (re-)uploading your **Lab07** folder into your **csc220-cXXXX** folder by the deadline **(Tuesday @ 11:59pm)**
* **For all your assignments, please start early and seek help early (either from the instructor or the TAs).**