CIS211: Data Structures

Delaware Tech, CT Department

Wilmington

(Instructor: Lee Hsu)

**Project #4: Stacks – RPN**

1. **Introduction:**

Refer to chapters 5 and 6 of your textbook for a detailed discussion on Infix, Postfix, and Prefix. The postfix notation is also called Reverse Polish Notation (RPN) in honor of the Polish logician Jan Lukasiewicz who observed that parentheses are not necessary in this form.

Files provided for this project: StackInterface.java, LinkedStack.java, Postfix.java, Driver.java, and p4in.txt.

1. **Discussion:**

As discussed in your textbook that in order to process an algebraic expression, we usually have its Infix form converted to its Postfix form, and then evaluate the Postfix form. However, compare the following two expressions:

* + - 1. A / B + C – D 🡺 AB/C+D-
      2. 20 /15 + 3 – 5 🡺 20 15 / 3 + 5 –

Obviously, the expression that uses the single-letter variables is easier to process. So, in this project, you are going to study the sample programs provided by the author and modify it according to the following specifications.

1. **Assignment:**

Consider the input file **“p4in.txt”** has the following contents (Assuming that all expressions are correct):

2 + 3

(2 + 3) \* 4

2 \* 3 / (4 - 5)

2 / 3 + (4 - 5)

2 / 3 + 4 - 5

2 ^ 3 ^ 4

(2 ^ 3) ^ 4

2 \* (3 / 4 + 5)

(2 + 3) / (4 - 5)

2 / (3 - 4) \* 5

2 - (3 / (4 - 5) \* 6 + 0) ^ 1

(2 - 3 \* 4) / (5 \* 6 ^ 0) \* 1 + 8

Your job is to produce the output file named “**p4out.txt**” similar to the following:

|  |  |
| --- | --- |
| Symbol Table | |
| Variable | Value |
|  |  |
| a | 2 |
| b | 3 |
| c | 4 |
| d | 5 |
| e | 6 |
| f | 0 |
| g | 1 |
| h | 8 |

|  |  |  |  |
| --- | --- | --- | --- |
| Input | Infix | Postfix | Result |
| 2 + 3 | a+b | ab+ | 5 |
| ( 2 + 3) \* 4 | (a+b)\*c | ab+c\* | 20 |
| 2 \* 3 / (4 - 5) | Omitted Here | | |
| 2 / 3 + (4 - 5) |
| 2 / 3 + c - d |
| 2 ^ 3 ^ 4 |
| (2 ^ 3) ^ 4 |
| 2 \* (3 / 4 + 5) |
| (2 + 3) / (4 - 5) |
| 2 / (3 - 4) \* 5 |
| 2 - (3 / (4 - 5) \* 6 + 0) ^ 1 |
| (2 - 3 \* 4) / (5 \* 6 ^ 0) \* 1 + 8 |

**NOTES:**

* 1. The border above is not part of the output.
  2. You must implement ArrayStack.java and modify Positfix.java so that **valueStack** uses ArrayStack and **operatorStack** uses LinkedStack.

1. **Hints:**
   1. Do this program step by step.
   2. It may be easier to use String to split the expression to obtain the array of values.
   3. Build your symbol table using single-letter variable name that starts from ‘a’.