CS272 Lab Assignment #6.

Due: Thursday, 3/29 by 11:30pm.

0.1: Read Chapter 6 from the textbook.
0.2. In this assignment you must use linked list implementation for generic stack class presented in Chapter 6 of the textbook. The code for linked list version of generic stack (LinkedStack< E >) and generic node (Node< E >) classes is provided:

- LinkedStack.java
- Node.java

You should not make any changes to these classes (Node class and LinkedStack class).

1: Write a program (with the name *Palindrome.java*) that will use stacks to recognize palindromic sequences of integers. A sequence of integers is palindromic if it reads the same backward and forward. For example, sequence 5 12 32 4 32 12 5 is palindromic. Sequence 3 4 5 4 2 3 is not palindromic. Your program should prompt the user to enter a sequence of integers in a line and output whether the sequence is palindromic or not. Dialog with the user may look like the following:

```
How many integers are in the sequence? 8
Please enter a sequence of 8 integers:
5 20 101 2 2 101 20 5
This sequence is palindromic
```

You will need three stacks to implement the program. You may only use stacks to store the numbers. You are not allowed to store numbers in an array.

2: Do programming project 9 from Chapter 6 of the textbook. Name your program *ComputeInfix.java*. Your program should prompt the user to enter an arithmetic expression and output the value of the expression. Assume that the input contains integers (no variables), operations +, -, *, /, and parenthesis (and). No other characters should be in the input string. The integers are all non-negative. You may assume that input expression always has correct syntax (balanced parentheses, etc.) Your program does not have to detect syntactic errors in the expression. Integers may have more than one digit in them, e.g. 25, 143, etc. To extract integers from a strings you may use the following method:
How to extract an integer from a string
Dialog with the user may look like the following:

```
Please enter an expression: 6+(2+3*4)/2
The expression evaluates to 13
```

Pseudocode to evaluate an infix expression without full parentheses is the following:

```
_____________________________________
1. Initialize two stacks: a numbers stack and an operators stack.
2. do
     if (the next input is a left parenthesis)
       Read the left parenthesis and push it onto the operators stack
     else if (the next input is a number)
       Read the number and push it onto the numbers stack
     else if (the next input is one of the operation symbols)
```

```
     {
        while (1) the operators stack is not empty, and
               (2) the top symbol on the operators stack is NOT a left
parenthesis, and
               (3) the top symbol on the operators stack is NOT an operation with
                   lower precedence than the next input
           do the following:
              pop an operation off the operators stack and
              use that operation by popping two numbers off the numbers stack,
                        applying the operation to the two numbers, and
                        pushing the answer back on the numbers stack
        After finishing the while loop read the next input (operation) and
        push it onto the operators stack
     }
     else
     {
        Read and discard the next input symbol (which should be a right
parenthesis).
        while the top symbol on operators stack is NOT a left parenthesis and
              the operators stack is not empty
           do the following:
              pop operation off the operators stack and
              use that operation by popping two numbers off the numbers stack,
                        applying the operation to the two numbers, and
                        pushing the answer back on the numbers stack
        If no left parenthesis is encountered in the operators stack, then
              print an error message indicating unbalanced parentheses and halt.
        Pop and discard the left parentheses from the operators stack.
     }
   while (there is more of the expression to read)
3. while the operators stack is not empty do the following:
              pop operation off the operators stack and
              use that operation by popping two numbers off the numbers stack,
                        applying the operation to the two numbers, and
                        pushing the answer back on the numbers stack
              (There should be no remaining left parentheses; otherwise,
               the input expression did not have balanced parentheses.)
```

_____

If the infix expression has correct syntax (balanced parentheses, etc.), then after execution of the pseudocode, the operators stack will be empty and the numbers stack will contain one number which is the value of the expression.

What to submit:

- Submit all your source code (Palindrome.java and ComputeInfix.java files) electronically on Canvas.