ECE 370    Programming Assignment 2

Fall 2015

Assigned  9/29/15

Due       10/14/15

The following algorithms have been covered in the lecture:

1. INFIX\_TO\_POSTFIX -converts infix expression to postfix using stack and queue
2. EVALUATE\_POSTFIX -evaluates postfix expression using a stack

**Note: all stacks and queues must be implemented using LINKED LISTS.**

You need to write operations such as PUSH, POP, and other procedures for manipulating stacks.  You must implement stack and queue, as defined in the textbook or a style of similar.  You **cannot** just make a function call to the standard stack/queue libraries.  The stack and queue functions must be in your own codes.

The infix expressions to be evaluated are follows.  Your main program reads in an infix expression, calls INFIX\_TO\_POSTFIX to convert it to postfix form, and then calls EVALUATE\_POSTFIX to evaluate it.  For each infix expression, your program should print the original infix expression, its postfix expression, and the result of the evaluation (that is, the value of the expression).  Your program should check for end-of-file and stop when there are no more infix expressions.  After processing all the expressions, your program should print a final line that is the sum of all the values.

The only operators used in the infix expressions are multiplication (\*), division (/), addition (+), subtraction (-), and exponential (^).  Standard C++ precedence rules are observed.  Parentheses are also used.  As is customary, anything within parenthesis is evaluated before anything else is evaluated.  You may assume there will be no unary minus.  All operands are one-digit decimal numbers with no decimal point.  The result of each calculation should be float.

The input data file name should be

"***a2.txt***"

   Sample Test Data:

   3\*2-8

   2\*7+(4-6)\*(8+6)/3

   9\*2+((4-3)\*2\*4)/2^2

Stack and queue examples can be found in the textbook.   You cannot assume the number of infix expressions in a2.txt

Note:  See Lecture Note 03 PPT for the algorithm.