CSE 330 Data Structures
Lab 3: Infix to postfix

October 8, 2014

Due date: Monday (for Monday section) or Wednesday (for Wednesday section) May 11 or 13, 2015, before lab begins.

Write a program in C++ with the following specifications.[1]

```
// An algorithm for infix to postfix expression conversion.
// For example,   a + b - c     translates to   a b + c -
//                a + b * c     translates to   a b c * +
//                (a + 2) / (5 + d)   goes to   a 2 + 5 d + /
// Valid operands are single digits and characters: 0-9 a-z A-Z
// Valid operators are: + - * / ( )
// Highest precedence:  * /
// Lowest precedence:   + -
// ( has lowest precedence on the stack and highest precedence outside of stack.
// ) never goes on stack.
// Bottom of the stack has the lowest precedence than any operator.
// Use a prec() function to compare the precedence of the operators based on the above rules.
// Note there is little error checking in the algorithm!

while there is more input
        if input is an operand
                print input
        else
                if input is '('  // '(' has lowest precedence in the stack, highest outside
                        push input on stack
                else if input is ')'
                        while stack is not empty and top of stack is not '('
                                print top of stack
                                pop stack
                        if stack is not empty
                                pop stack
                        else error  // no matching '('
                else if stack is empty or prec(top of stack) < prec(input) // bottom of stack has lower
                                                                           // precedence than everything
                        push input on stack
                else if prec(top of stack) >= prec(input)
                        while stack is not empty and prec(top of stack) >= prec(input)
                                print top of stack
                                pop stack
                        push input on stack
                else check for errors
while stack is not empty
        print top of stack
        pop stack
```

[1] This lab specification is by Dr. Kay Zemoudeh.

- Header for the code (standard for all labs, with appropriate adjustments of course like for the lab title, etc.):

```
// Class: CSE 330
// Term:
// Instructor: George M. Georgiou
// Name(s): ...
// Lab 3
// Title: Infix to postfix expression conversion
```

- For all labs, code **must be commented.** You will loose points if there are not not sufficient comments to explain code.
- What to turn in:

1. A single document in **pdf or text** format that contains (1) the source code with above header at the top and (2) the run (typescript) that shows the output.
2. The name of the file should be in this format: `CSE_330_Lab1_Lastname1_Lastname2.pdf`. Upload on Blackboard. That would be the official turn-in time.
3. A hardcopy of the above at the beginning of the lab session on the due date.