Assignment 1: Infix to Postfix Expression using Stack
CSE 225: Data Structures and Algorithms
Due Date: 10 November 2017

Instructions: Please zip your project folder containing all codes and send it to sifat.momen@northsouth.edu by the due date. Write your name, id and section number in the driver file (which should be commented out). Any evidence of plagiarism will lead to zero mark in this assessment.

Total points in this assignment = 15

1 Introduction

In the class, we discussed how to convert arithmetic expressions from the infix to postfix form. *Postfix*, also known as the *Reverse Polish Notation (RPN)*, is a notational system where the operator follows the operands. In this assignment your job is to write a program in *C/C++* that will convert a correct infix arithmetic expression to its corresponding postfix expression and also evaluate the expression.

2 Algorithms

The first part of the assignment requires you to convert an arithmetic expression from the infix form to the postfix form. The following is the algorithm for this.

1. Examine the next element in the input.
2. If it is operand, output it.
3. If it is opening parenthesis, push it on to the stack.
4. If it is an operator, then
    (a) Pop until the top of the stack has an element of lower precedence.

   (b) Then push it.

5. If it is a closing parenthesis, pop operators from stack and output them until an opening parenthesis is encountered. Pop and discard the opening parenthesis.

6. If there is more input go to *step 1*

7. If there is no more input, pop the remaining operators to output.

The second part of the assignment needs you to determine the value of the arithmetic expression. The following is the algorithm to evaluate the expression in the postfix form.

1. Initialise an empty stack

2. Read next item in the expression

   (a) If item is an operand, push it into the stack

   (b) Else, if item is an operator, pop top two items off the stack, apply the operator, and push the answer back into the stack

3. If there are more items to process, go to step 2

4. Pop the answer off the stack.

3 Sample Input

(4+8)*(6-5)/((3-2)*(2+2))

4 Sample Output

```
4 8 + 6 5 - * 3 2  2 2 + * /
3
```