BLG 233E DATA STRUCTURES AND LABORATORY

EXPERIMENT 6 – STACKS

IMPORTANT REMINDERS

1. It is not allowed to use USB sticks during the lab sessions.
2. You should unplug your ethernet cables during the lab sessions.
3. Any reference book or help material (C++) is allowed.

In this experiment, you are required to write an infix to postfix converter and an evaluator for the postfix expressions.

As an input, user will enter an infix expression and your program will express this infix expression in postfix expression format (like in our last recitation). After this phase, your program should calculate the value of this expression using stack data structure.

Pseudo code of the infix to postfix conversion part is given below.

```
Initialize an empty stack.
Scan the Infix string from left to right. For all the characters,
    If the scanned character is an operand, add it to the Postfix string.
    If the scanned character is an operator and
        If the stack is empty or the character is ‘(‘, push the character to the stack.
        If the stack is not empty, compare the precedence of the character on top of the stack (topStack). If topStack has higher precedence over the scanned character pop it from the stack to the Postfix string else push the scanned character to stack. Repeat this step as long as the stack is not empty and topStack has precedence over the character.
        If the character is ‘)’ pop all the characters to the Postfix string until reaching ‘(‘ in the stack.
After all the characters are scanned, add any remaining character in the stack to the Postfix string.
```

For example, assume that user enters 2*3-6/(2+1), outputs should be like this,

```
Postfix expression: 23*621+/-                    Value: 4
```