(Matching Parentheses with a Stack) – a mini-compiler

(0) **Download the Lab7 Starter Kit**.
(1) Write a C++ (client) program that uses the `MyCharStack` class to read lines of a file from a text file, checks parenthesis matching, and reports if each line is OK or contains mismatched parentheses.

(2) Work on the assignment in parts:

a) In your client (`main.cpp`), open a file (see the two examples provided). Does it open ok? *Just get this much of your program to compile and work. Do not proceed until you have this working!*
b) Read lines of the file one line at a time into a string. For this step, print (echo) the lines to the console so you are assured you are dealing with the file ok.
c) Note: we will assume that all statements *must* be on one line; that is, no statements are allowed to wrap-around.
d) Look over the code for `MyCharStack`, both the `.h` and `.cpp`. You should notice that I have left some work for you. Fix the .cpp so that all your stack operations work.
e) Declare an object stack, `S`, of the class `MyCharStack`. Practice pushing and popping some characters on and off your stack. Don't worry about the actual characters in the file for now, just make sure your stack, `S`, is working. That is, can you do this?

```
MyCharStack   S;    // calls the CTOR

if ( !S.IsFull() )
        S.Push('x');
char c;
c = S.Top();
cout << "I just popped a" << c << endl;
S.Pop()
```

f) Iterate over each character in your `line` (string). Note: strings can be indexed, `[i]` since they are really arrays of characters. For example, `line[2]` is the third character on a line. Use a `switch` statement to handle the cases of `'('` and `')'`. You should push onto the stack when encountering a left parenthesis and pop the stack when you read a right paren.

Your program should produce "syntax error messages", well, at least those associated with mismatched parentheses.

`(( x+2 );`	"missing RIGHT PAREN" on line *n*
`(y-8));`	"missing LEFT PAREN" on line *n*
`(((((…`	"Stack is full on line *n*" (*exit the program if this occurs*)

You program shold report an error it finds on a line, but also continue with that line.
Each line should be treated individually, that is, you should flush your stack after each line and start again.

Below is an example input file (on the left) and a sample of the output that your program should produced if given this input file. **Note: also try running *your program* as input to (well) your program ☺**

```
int main()
{
    short x=0, y=8;
    x=((y+x)*3);
    y = (y*3));

    y = (x*3)) - (x*x));


    cout << ((x+y);

    return 0;
}
```

```
Line #1 is OK.
Line #2 is OK.
Line #3 is OK.
Line #4 is OK.
Stack is Empty: Missing LEFT PAREN '(' on line 5
Line #6 is OK.
Stack is Empty: Missing LEFT PAREN '(' on line 7
Stack is Empty: Missing LEFT PAREN '(' on line 7
Line #8 is OK.
Missing RIGHT PAREN ')' on line 9
Line #10 is OK.
Line #11 is OK.
Line #12 is OK.
```