PYTHON #02
CSCI-410 Spring 2013

In this assignment you will write a "scanner" that reads an input file and processes it based on knowledge of what has been seen previously (the state) and the value of the next character in the file. The result is the type of machine, known as a finite automaton (FA), upon which most lexical analyzers (or, "lexers") are based.

Your program will read a Jack source code file (Jack is the name of the language you will be working with in the latter part of this course) and strips all of the comments from it. The output from your program will consist of two parts: The first is an output file that contains the stripped code (i.e., the Jack source code with the comments removed), the second is a summary, printed to the console, of the input and output files.

The comments in Jack follow the same rules as comments in C and many other languages. There are two types of comments, block comments and end-of-line comments. Block comments may start or stop anywhere on a line and may span multiple lines. A block comment starts with the character sequence "/*" (a forward slash followed by an asterisk, sometimes referred to as "slash splat") and end with the character sequence "*/" (splat slash). Once in a block comment, all other characters are part of the comment until the end of the comment block is reached. In particular, this means that comment blocks cannot be nested because the start of the second (and subsequent) comment blocks have no relevance while the end of the inner most comment block will be seen as the end-of-comment sequence for the outermost comment block. The other kind of comment is the end-of-line comment, which starts with a "//" (double forward slash) sequence and ends at the end of the line. Note that the new line character that marks the end of an end-of-line comment is NOT part of the comment itself.

Name your main file (the one containing your top level code) JackLex.py. You should be able to run your Python program (script) from the command line, supplying two arguments with the first being the input file to read from and the second being the output file to write to, similar to the following (the exact details depend on how you have configured your system):

```
C:> python JackLex.py input.jck output.jck
```

The summary that you will print out will have the following basic format (shown via example). The details of how wide your columns are and how you align them are up to you.

```
                  INPUT          OUTPUT
Filename          input.jck      output.jck
Lines               123              97
Characters         5752            3612
Block comments       12               0
  Characters       1793               0
EOL comments         42               0
  Characters        347               0
```

PYTHON #02
CSCI-410 Spring 2013

Restrictions

Python supports what are known as "regular expressions", which you may or may not be familiar with. You are NOT to use them. Part of the objective of the course is for you to gain an appreciation for the mechanics involved in implementing the tools you are using and using regular expression utilities would defeat that goal.

You are free to read the input file an entire line at a time, but you are to process the contents one character at a time. Any information you need about what you have seen previously is to be captured in a variable named "state". For instance, if the value of state is 3, this might indicate that you are within an end-of-line comment while a state equal to 7 might indicate that the last character seen was an asterisk that might be the first character in the sequence that will end the current block comment. You should document what your states are and what they mean in your comments.

Your files should contain comments at the top giving your name, the course and assignment identification, and the due date.

Submission

In a manner similar to the ECS project, place all files needed for this assignment into a directory named PY02 and zip up the entire directory into a zip file of the form:

CS410_UserID_PY_02.zip

GRADING RUBRIC – 40 pts

10 pts	**Effort**
10 pts	**Strips Block comments correctly**
10 pts	**Strips EOL comments correctly**
10 pts	**Does not strip any non-comment content**
5 pts	**Output summary in an acceptable format.**
5 pts	**Correct count on number of lines and characters.**
-5 pts	**Inadequate header comments**
-10 pts	**Using Python's regular expression capabilities.**
-10 pts	**Not processing the input file one character at a time.**
-2pts	**Incorrect submission (filename, etc).**