CSE 3302: Programming Languages

Lab - Nesting depth of curly braces using Python

100 points

INSTRUCTIONS

1. **Do NOT plagiarize.**
2. **No group work. All work should be your own.**
3. **Do not discuss your work with other students in the class.**
4. **You CANNOT borrow code from online sources.**
5. **Turn in your program using Canvas. Do not email your program to the TA or the instructor.**
6. **Name your document as netid_PA4.py where netid is your UTA netid. If you do not know your netid, check what it is using NetID Self Service. Your 1000 number is NOT your netid. If your file name is wrong, your assignment will not be graded.**
7. **All code should be your own. You may not copy code from the slides, book, others, or the internet unless specified.**
8. **Display your results as in the sample output.**
9. **Write an explanation of your code using comments. If the explanation is not clear, you will NOT receive full credit.**
10. **The code should have your name, 1000 number, the due date of the assignment, and OS and Python version used as the first 4 lines in order.**
11. **Use Python; it can be downloaded for free at python.org.**
12. **There are two opportunities for extra credit described at the end and will turn them in as a separate source file by containing the base code and the extra credit code into a ZIP file using the same naming convention.**

Description

Write a program that will take an input file called input.txt which will contain a Java program and parse it and output an annotated version.

Your program will track the nesting depth of the braces of the input file and will output an annotated version of the file.

Your final program will:

1. List the nesting depth of curly braces
2. Ignore braces inside quotes or comments
3. Test for unmatched braces (expected '}' but found EOF); output an error message
4. Allow for multiple braces on the same line, for example "}}" ending a loop and a function
5. Handle block comments that cross multiple lines of the input file.

example:
/* comment with
ignored brace { */

Use Python

Assume that braces can be anywhere
Assume that all quoted strings begin and end on the same line.

Sample INPUT:

```
import blah;
class Foo
{
    void Foo()
   {
      System.out.println("braces are fun! {{{{{");    // ignored
      if (condition)
      {
        // also ignored: {
        int a = 1;
        // as is this: }
        if (condition){   int b = 1; } a = 42;

      } // what if this were on one line?  }}}
   }
}
//end of program
```

Sample annotated OUTPUT:

```
-------------
0 import blah;
0 class Foo
1 {
1    void Foo()
2    {
2        System.out.println("braces are fun! {{{{{"); // ignored
2        if (condition)
3        {
3            // also ignored: {
3            int a = 1;
3            // as is this: }
3        }
2    }
1 }
0 // end of program
```

Extra credit A (10 points)
Output the source code to be properly indented, even if the input was all one line. This is also known as 'pretty print'.

INPUT:

```
if (condition){   int a = 1; }
```

BECOMES:

```
if (condition)
{
  int a = 1;
}
```

Extra Credit B (10 points)
Handle input in which characters can be escaped using the backslash character "\"
Example:

```
System.out.println("""");  // how do you print a single quote character? This is not it.
System.out.println("{\"}");  // you have to escape it, does the close brace get ignored?
```

If you work on the extra credit you will turn in TWO source files, the base file named <netID>_PA4.py and your extra credit code file named <netid>_PA4_EC.py and your input file will be input_EC.txt. These files will be contained in a ZIP file named <netID>_PA4_EC.zip

Note :- The sample input and output will not be the only test case used to test the working of your code. Do NOT contact the TA or instructor regarding the test cases.