Coding challenge

Problem definition

Scripted calculator

Write a piece of code to read a script file, check the syntax of an expression in each line as per the following rules, do the calculation and display its output. The code should be well documented and compilable (preferably no warnings). Programming language can be Python or C.

Syntax of the script:

- This hypothetical calculator has 4 keywords - ADD, SUB, MUL and DIV. These 4 keywords indicate basic mathematical operations as their names suggest. They are binary operators; meaning they can take only two operands at a time.
- Brackets are used to provide the arguments. There should be a closing bracket ')' corresponding to an opening bracket '('.
- '#' is used to write comments in single line. Assume that comments always start in a newline. No need to support inline comments.
- User should be able to give nested expressions too.

The program should take the script file as command line input.

Example: *calc C:\test\Level1.calc*

Challenge Levels

Level 1 - Easy

At least one keyword must be implemented satisfying the following requirements:

1. Code should be compilable. Preferably 0 warnings.
2. Code should be well documented. Use neat and consistent naming conventions.
3. The executable should accept a script file as command line argument as specified above. (In the case of python, no need to make an executable. Just run it as an argument to *python*)
4. The executable should parse the script, execute the implemented keyword and display the output.

Level 1 example

```
ADD (1,2)
```

Level 2 - Medium

Implement parser for single line comment, and add support for more than one keyword. **Attempt this only after achieving Level 1**.

Level 2 example

```
# This is a sample script
ADD (1,2)
SUB (4,3)
MUL (10,4)
DIV (12,4)
```

Level 3 - Hard

Implement nested expression. **Attempt this only after achieving Level 2** with all the 4 keywords implemented and fully functional.

Level 3 example

```
# (1 + ((2*3) - (5+3)))/10
DIV(ADD(1,SUB(MUL(2,3), ADD(5,3))), 10)
```

Note:

- Aim to fulfil Level 1 requirements first and then build upon that.
- You will get bonus points as you go up the challenge levels 2 and 3.
- Ensure that code is compilable before sending it back.
- Do not use any specialized library calls other than for basic string/file/IO operations.
- You are free to assume anything which is not explicitly stated here. Would appreciate if the assumptions are documented in the code.
- This is just a way for us as interviewers to know your programming skill. Be cool and enjoy the process. We trust you. All the best!