722. Remove Comments

Approach #1: Parsing [Accepted]

Intuition and Algorithm

We need to parse the `source` line by line. Our state is that we either are in a block comment or not.

- If we start a block comment and we aren't in a block, then we will skip over the next two characters and change our state to be in a block.
- If we end a block comment and we are in a block, then we will skip over the next two characters and change our state to be *not* in a block.
- If we start a line comment and we aren't in a block, then we will ignore the rest of the line.
- If we aren't in a block comment (and it wasn't the start of a comment), we will record the character we are at.
- At the end of each line, if we aren't in a block, we will record the line.

Python

```
class Solution(object):
    def removeComments(self, source):
        in_block = False
        ans = []
        for line in source:
            i = 0
            if not in_block:
                newline = []
            while i < len(line):
                if line[i:i+2] == '/*' and not in_block:
                    in_block = True
                    i += 1
                elif line[i:i+2] == '*/' and in_block:
                    in_block = False
                    i += 1
                elif not in_block and line[i:i+2] == '//':
                    break
                elif not in_block:
                    newline.append(line[i])
                i += 1
            if newline and not in_block:
                ans.append("".join(newline))

        return ans
```

Java

```java
class Solution {
    public List<String> removeComments(String[] source) {
        boolean inBlock = false;
        StringBuilder newline = new StringBuilder();
        List<String> ans = new ArrayList();
        for (String line: source) {
            int i = 0;
            char[] chars = line.toCharArray();
            if (!inBlock) newline = new StringBuilder();
            while (i < line.length()) {
                if (!inBlock && i+1 < line.length() && chars[i] == '/' && chars[i+1] == '*') {
                    inBlock = true;
                    i++;
                } else if (inBlock && i+1 < line.length() && chars[i] == '*' && chars[i+1] == '/') {
                    inBlock = false;
                    i++;
                } else if (!inBlock && i+1 < line.length() && chars[i] == '/' && chars[i+1] == '/') {
                    break;
                } else if (!inBlock) {
                    newline.append(chars[i]);
                }
                i++;
            }
            if (!inBlock && newline.length() > 0) {
                ans.add(new String(newline));
            }
        }
        return ans;
    }
}
```

Complexity Analysis

- Time Complexity: $O(S)$, where S is the total length of the source code.
- Space Complexity: $O(S)$, the space used by recording the source code into `ans`.

Analysis written by: @awice (https://leetcode.com/awice).

Comments: 2

Sort By

Type comment here... (Markdown is supported)

Preview | Post

(/nishtha_2)

Nishtha_2 (/nishtha_2) ★ 20 April 1, 2018 1:25 PM

I think it is possible to do it in place

0 Share Reply

(/apeforest)

apeforest (/apeforest) ★ 32 November 20, 2017 1:59 AM

Why don't you need to check for array bound line[i:i+2] in Python, but you did check that in Java

0 Share Reply

SHOW 1 REPLY