71. Simplify Path

Difficulty	medium
☑	☑
Finished	@July 20, 2023
Problem	strings
Previously asked company	Amazon Goldman Sachs
website	leetcode

Question:

Given a string `path`, which is an **absolute path** (starting with a slash `'/'`) to a file or directory in a Unix-style file system, convert it to the simplified **canonical path**.

In a Unix-style file system, a period `'.'` refers to the current directory, a double period `'..'` refers to the directory up a level, and any multiple consecutive slashes (i.e. `'//'`) are treated as a single slash `'/'`. For this problem, any other format of periods such as `'...'` are treated as file/directory names.

The **canonical path** should have the following format:

- The path starts with a single slash `'/'`.
- Any two directories are separated by a single slash `'/'`.
- The path does not end with a trailing `'/'`.
- The path only contains the directories on the path from the root directory to the target file or directory (i.e., no period `'.'` or double period `'..'`)

Return *the simplified* ***canonical path***.

Example 1:

```
Input: path = "/home/"
Output: "/home"
Explanation: Note that there is no trailing slash after the last directory name.
```

Example 2:

```
Input: path = "/../"
Output: "/"
Explanation: Going one level up from the root directory is a no-op, as the root level is t
he highest level you can go.
```

Example 3:

```
Input: path = "/home//foo/"
Output: "/home/foo"
```

```
Explanation: In the canonical path, multiple consecutive slashes are replaced by a single
 one.
```

Optimal solution:

Time complexity: O(n)

Space complexity: O(n)

```
class Solution(object):
    def simplifyPath(self, path):
        stack = []
        curr = ''

        for c in path + '/':
            if c == '/':
                if curr == '..':
                    if stack: stack.pop()
                elif curr != '' and curr != '.':
                    stack.append(curr)
                curr = ''
            else:
                curr += c

        return '/' + '/'.join(stack)
```