# 71 simplify path

| | |
|---|---|
| Created | @March 14, 2022 10:53 PM |
| Tags | Medium |
| link | https://leetcode.com/problems/simplify-path/ |
| Problem Number | 71 |
| companies | facebook |

## Description

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***.

```
import re
class Solution:
    def simplifyPath(self, path: str) -> str:
        if path[-1] == '/':
            path = path[:-1]
```

```
        path = path[1:]
        path = path.split('/')

        stacked_path = []
        for i in path:
            if i == '':
                continue
            if i == '.':
                continue
            if i == '..':
                if len(stacked_path) == 0:
                    continue
                stacked_path.pop()
            else:
                stacked_path.append(i)

        path = stacked_path
        print (path)
        if len(path) == 0:
            return '/'
        if len(path) == 1:
            return '/' + path[0]

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