71. Simplify Path

Initial thoughts

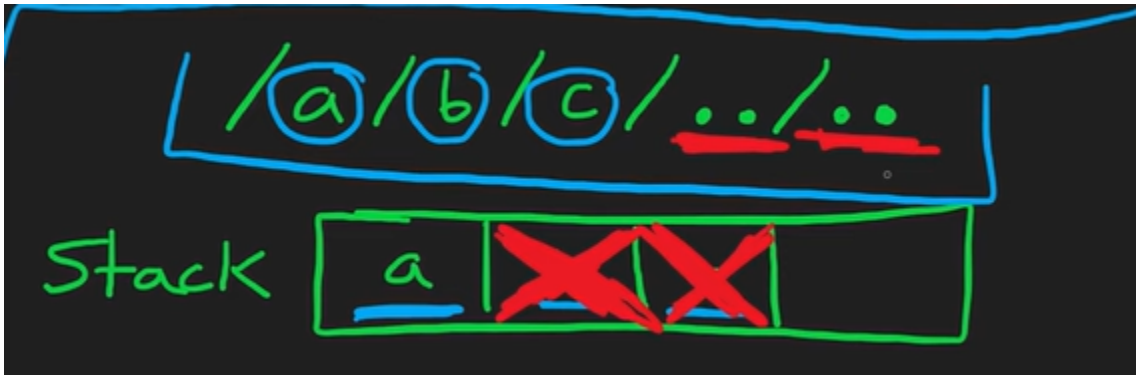
1. Thinking to solve this question using list as a stack in python
2. There will be multiple if and while loops

Actual Solution

1. Lets take an example and understand the soln

2. Here we will traverse the string from left to right
3. First we will encounter **"/"** and add it in a list
4. Next we encounter **".."** it says to go back to previous directory..but as we are starting with root directory..there will be no previous directory..so we just ignore these **".."**
5. Next we encounter **"/"** as we already have one slash and we do not need consecutive slashes..so we just ignore it
6. Next we encounter **"abc/"** which is an actual directory..so we'll include that in output
7. Now we encounter **"/"** which will be consecutive slash..so we ignore it
8. Next we get **"."** which we need to simply ignore it
9. Now we encounter **"/"** adding it makes a consecutive slash..so we just ignore it
10. Next we get an actual directory **"def"** so we just include that in our list
11. Next we encounter **"/"** as we dont want to include **"/"** at the end of our simplified path ..so we just ignore it
12. **Output :** 

13. So why we need to use stack? Because when we encounter **".."** then we need to backtrack to previous directory



14. So if we encounter a **“..”** we’ll use pop() function and remove the directory

Python code :

1.
```
class Solution:
    def simplifyPath(self, path: str) -> str:
        stack = []
        cur = ""

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

        return "/" + "/".join(stack)
```
2. Here in this code ..we have initialized stack and an empty string (used to store the directories)
3. Now we iterate each char from left to right

4. Now if we encounter