1003. Check If Word Is Valid After Substitutions

Given a string `s`, determine if it is **valid**.

A string `s` is **valid** if, starting with an empty string `t = ""`, you can **transform** `t` **into** `s` after performing the following operation **any number of times**:

- Insert string `"abc"` into any position in `t`. More formally, `t` becomes `t<sub>left</sub> + "abc" + t<sub>right</sub>`, where `t == t<sub>left</sub> + t<sub>right</sub>`. Note that `t<sub>left</sub>` and `t<sub>right</sub>` may be **empty**.

Return `true` *if* `s` *is a **valid** string, otherwise, return* `false`.

Example 1:

```
Input: s = "aabcbc"
Output: true
Explanation:
"" -> "abc" -> "aabcbc"
Thus, "aabcbc" is valid.
```

Example 2:

```
Input: s = "abcabcababcc"
Output: true
Explanation:
"" -> "abc" -> "abcabc" -> "abcabcabc" -> "abcabcababcc"
Thus, "abcabcababcc" is valid.
```

Example 3:

```
Input: s = "abccba"
Output: false
Explanation: It is impossible to get "abccba" using the operation.
```

Example 4:

```
Input: s = "cababc"
Output: false
Explanation: It is impossible to get "cababc" using the operation.
```

Constraints:

- `1 <= s.length <= 2 * 10<sup>4</sup>`

- `s` consists of letters `'a'`, `'b'`, and `'c'`

```python
class Solution:
    def isValid(self, s: str) -> bool:
        stack = []

        for i in range(len(s)):
            ch = s[i]

            if ch=='c':
                if len(stack)>=2 and stack.pop()=='b' and stack.pop()=='a':
                    continue
                else:
                    return False
            else:
                stack.append(ch)

        return True if len(stack)==0 else False
```