0125. 验证回文串

ITCharge 大约 1 分钟

- 标签：双指针、字符串
- 难度：简单

题目链接

- 0125. 验证回文串 - 力扣

题目大意

描述：给定一个字符串 `s` 。

要求：判断是否为回文串（只考虑字符串中的字母和数字字符，并且忽略字母的大小写）。

说明：

- 回文串：正着读和反着读都一样的 串。
- $1 \leq s.length \leq 2 * 10^5$。
- `s` 仅由可打印的 ASCII 字符组成。

示例：

- 示例 1：

```py
输入："A man, a plan, a canal: Panama"
输出：true
解释："amanaplanacanalpanama" 是回文串。
```

- 示例 2：

```py
输入："race a car"
输出：false
解释："raceacar" 不是回文串。
```

解题思路

思路 1：对撞指针

1. 使用两个指针 `left` ， `right` 。 `left` 指向字符串开始位置， `right` 指向字符串结束位置。
2. 判断两个指针对应字符是否是字母或数字。 通过 `left` 右移、 `right` 左移的方式过滤掉字母和数字以外的字符。
3. 然后判断 `s[left]` 是否和 `s[right]` 相等（注意大小写）。
    1. 如果相等，则将 `left` 右移、 `right` 左移，继续进行下一次过滤和判断。
    2. 如果不相等，则说明不是回文串，直接返回 `False` 。
4. 如果遇到 `left == right` ，跳出循环，则说明该字符串是回文串，返回 `True` 。

思路 1：代码

```py
class Solution:
    def isPalindrome(self, s: st    > bool:
        left = 0
        right = len(s) - 1

        while left < right:
            if not s[left].isalnum():
                left += 1
                continue
            if not s[right].isalnum():
                right -= 1
                continue

            if s[left].lower() == s[right].lower():
                left += 1
                right -= 1
            else:
                return False
        return True
```

思路 1：复杂度分析

- **时间复杂度**：$O(len(s))$。
- **空间复杂度**：$O(len(s))$。