题目地址(1737. 满足三条件之一需改变的最少字符数)

https://leetcode-cn.com/problems/change-minimum-characters-to-satisfy-one-of-three-conditions/

题目描述

给你两个字符串 a 和 b ，二者均由小写字母组成。一步操作中，你可以将 a 或 b 中的 任一字符 改变为 任一小写字母 。

操作的最终目标是满足下列三个条件 之一 ：

a 中的 每个字母 在字母表中 严格小于 b 中的 每个字母 。
b 中的 每个字母 在字母表中 严格小于 a 中的 每个字母 。
a 和 b 都 由 同一个 字母组成。
返回达成目标所需的 最少 操作数。

示例 1:

输入: a = "aba", b = "caa"
输出: 2
解释: 满足每个条件的最佳方案分别是:
1) 将 b 变为 "ccc", 2 次操作, 满足 a 中的每个字母都小于 b 中的每个字母;
2) 将 a 变为 "bbb" 并将 b 变为 "aaa", 3 次操作, 满足 b 中的每个字母都小于 a 中的每个字母;
3) 将 a 变为 "aaa" 并将 b 变为 "aaa", 2 次操作, 满足 a 和 b 由同一个字母组成。
最佳的方案只需要 2 次操作（满足条件 1 或者条件 3）。
示例 2:

输入: a = "dabadd", b = "cda"
输出: 3
解释: 满足条件 1 的最佳方案是将 b 变为 "eee" 。

提示:

1 <= a.length, b.length <= 105
a 和 b 只由小写字母组成

前置知识

- 计数
- 枚举

标签

- 枚举

难度

- 中等

入选理由

- 一个很典型的模拟题目

公司

- 暂无

思路

三个条件中，**前两个条件其实是一样的**，因为如果你会了其中一个，那么你只需要将 A 和 B 交换位置就可以解出另外一个了。

对于前两个条件来说，我们可以枚举所有可能。以条件一 `A 中的 每个字母 在字母表中 严格小于 B 中的 每个字母` 为例。我们要做的就是枚举所有可能的 A 的最大字母 和 B 的最小字母（其中 A 的最大字母保证严格小于 B 的最大字母），并计算操作数，最后取最小值即可。

代码上，我们需要先统计 A 和 B 的字符出现次数信息，不妨分别记为 counter_A 和 counter_B。接下来，我们就可以执行核心的枚举逻辑了。

核心代码:

```
# 枚举 A 的最大字母
for i in range(1, 26):
    t = 0
    # A 中大于等于 i 的所有字符都需要进行一次操作
    for j in range(i, 26):
        t += counter_A[j]
    # B 中小于 i 的所有字符都需要进行一次操作
    for j in range(i):
        t += counter_B[j]
    # 枚举的所有情况中取最小的
    ans = min(ans, t)
```

而对于第三个条件，则比较简单，我们只需要将 A 和 B 改为同一个字母，并计算出操作数，取最小值即可。我们可能修改成的字母一共只有 26 种可能，因此直接枚举即可。

核心代码:

```
for i in range(26):
     ans = min(ans, len(A) + len(B) - counter_A[i] - counter_B[i])
```

关键点

- 使用一个长度为 26 的数组计数不仅性能比哈希表好，并且在这里代码书写会更简单

代码

- 语言支持：Python3

Python3 Code:

```
class Solution:
    def minCharacters(self, A: str, B: str) -> int:
        counter_A = [0] * 26
        counter_B = [0] * 26
        for a in A:
            counter_A[ord(a) - ord('a')] += 1
        for b in B:
            counter_B[ord(b) - ord('a')] += 1
        ans = len(A) + len(B)
        for i in range(26):
            ans = min(ans, len(A) + len(B) - counter_A[i] - counter_B[i])
        for i in range(1, 26):
            t = 0
            for j in range(i, 26):
                t += counter_A[j]
            for j in range(i):
                t += counter_B[j]
            ans = min(ans, t)
        for i in range(1, 26):
            t = 0
            for j in range(i, 26):
                t += counter_B[j]
            for j in range(i):
                t += counter_A[j]
            ans = min(ans, t)
        return ans
```

我们也可以将操作封装成函数方便理解。其中：

- greater_cost(a, b) 表示 a 中严格大于 b 的最小操作数。
- equal_cost(a, b) 表示将 a 和 b 转化为同一字符的最小操作数。

Python3 Code:

```python
class Solution:
    def minCharacters(self, A: str, B: str) -> int:
        ca = collections.Counter(A)
        cb = collections.Counter(B)
        # ca 中严格大于 cb 的最小操作数
        def greater_cost(ca, cb):
            ans = float("inf")
            # 枚举 ca 中的最小值
            for i in range(1, 26):
                count = 0
                # 将 ca 中小于最小值的都进行一次操作
                for j in range(i):
                    count += ca[chr(97 + j)]
                # 将 cb 中大于等于最小值的都进行一次操作（注意这里的等号）
                for j in range(i, 26):
                    count += cb[chr(97 + j)]
                ans = min(ans, count)
            return ans

        def equal_cost(ca, cb):
            ans = float("inf")
            for i in range(26):
                ans = min(ans, len(A) + len(B) - ca[chr(97 + i)] - cb[chr(97 + i)])
            return ans

        return min(greater_cost(ca, cb), greater_cost(cb, ca), equal_cost(ca, cb))
```

复杂度分析

令 m, n 分别为数组 A 和数组 B 的长度。

- 时间复杂度：$O(m + n)$
- 空间复杂度：$O(26)$

更多题解可以访问我的 LeetCode 题解仓库：https://github.com/azl397985856/leetcode 。 目前已经 40K star 啦。

关注公众号力扣加加，努力用清晰直白的语言还原解题思路，并且有大量图解，手把手教你识别套路，高效刷题。

欢迎长按关注

上一页 下一页