# 题目

给你两个下标从0开始的字符串s和target。你可以从s取出一些字符并将其重排，得到若干新的字符串。

从 s 中取出字符并重新排列，返回可以形成target的最大副本数。

示例 1：

输入：s = "ilovecodingonleetcode", target = "code"

输出：2

解释：

对于 "code" 的第 1 个副本，选取下标为 4 、5 、6 和 7 的字符。

对于 "code" 的第 2 个副本，选取下标为 17 、18 、19 和 20 的字符。

形成的字符串分别是 "ecod" 和 "code" ，都可以重排为 "code" 。

可以形成最多 2 个 "code" 的副本，所以返回 2 。

示例 2：

输入：s = "abcba", target = "abc"

输出：1

解释：

选取下标为 0 、1 和 2 的字符，可以形成 "abc" 的 1 个副本。

可以形成最多 1 个 "abc" 的副本，所以返回 1 。

注意，尽管下标 3 和 4 分别有额外的 'a' 和 'b' ，但不能重用下标 2 处的 'c' ，所以无法形成 "abc" 的第 2 个副本。

示例 3：

输入：s = "abbaccaddaeea", target = "aaaaa"

输出：1

解释：

选取下标为 0 、3 、6 、9 和 12 的字符，可以形成 "aaaaa" 的 1 个副本。

可以形成最多 1 个 "aaaaa" 的副本，所以返回 1 。

提示：

1 <= s.length <= 100

1 <= target.length <= 10

s 和 target 由小写英文字母组成

# 分析

## 方法一：贪心算法

思路：

可以使用贪心算法来解决。首先，我们统计字符串 s 和 target 中各个字符的出现次数。然后，对于 target 中的每个字符，我们计算在 s 中出现的次数与 target 中出现的次数的比值，取最小值作为可形成副本的最大数量。

具体步骤如下：

1、统计字符串 s 和 target 中各个字符的出现次数，分别存储在两个数组 sCount 和 targetCount 中。

2、遍历 target 中的每个字符 c，计算 sCount[c] / targetCount[c] 的最小值，即为可形成副本的最大数量。

3、返回所有字符的最小值作为结果。

代码：

class Solution {

public:

int rearrangeCharacters(string s, string target) {

// 统计字符串 s 和 target 中各个字符的出现次数

vector<int> sCount(26, 0);

vector<int> targetCount(26, 0);

for (char c : s) {

sCount[c - 'a']++;

}

for (char c : target) {

targetCount[c - 'a']++;

}

// 计算可形成副本的最大数量

int maxCopies = INT\_MAX;

for (char c : target) {

if (targetCount[c - 'a'] == 0) {

continue; // 如果 target 中的某个字符在 s 中不存在，则跳过

}

maxCopies = min(maxCopies, sCount[c - 'a'] / targetCount[c - 'a']);

}

return maxCopies;

}

};