# 题目

给你一个混合了数字和字母的字符串 s，其中的字母均为小写英文字母。

请你将该字符串重新格式化，使得任意两个相邻字符的类型都不同。也就是说，字母后面应该跟着数字，而数字后面应该跟着字母。

请你返回 重新格式化后 的字符串；如果无法按要求重新格式化，则返回一个 空字符串。

示例 1：

输入：s = "a0b1c2"

输出："0a1b2c"

解释："0a1b2c" 中任意两个相邻字符的类型都不同。 "a0b1c2", "0a1b2c", "0c2a1b" 也是满足题目要求的答案。

示例 2：

输入：s = "leetcode"

输出：""

解释："leetcode" 中只有字母，所以无法满足重新格式化的条件。

示例 3：

输入：s = "1229857369"

输出：""

解释："1229857369" 中只有数字，所以无法满足重新格式化的条件。

示例 4：

输入：s = "covid2019"

输出："c2o0v1i9d"

示例 5：

输入：s = "ab123"

输出："1a2b3"

提示：

1 <= s.length <= 500

s 仅由小写英文字母和/或数字组成。

# 分析

## 方法一：双指针

思路：

代码：

class Solution {

public:

string reformat(string s) {

int sum\_digit = 0;

// 统计字符串中数字的数量

for (auto& c : s) {

if (isdigit(c)) {

sum\_digit++;

}

}

int sum\_alpha = s.size() - sum\_digit; // 字符串中字母的数量

// 如果数字和字母的数量相差大于1，则无法重新格式化

if (abs(sum\_digit - sum\_alpha) > 1) {

return "";

}

bool flag = sum\_digit > sum\_alpha; // 标记数字是否多于字母

// 重新格式化字符串

for (int i = 0, j = 1; i < s.size(); i += 2) {

if (isdigit(s[i]) != flag) {

// 找到下一个应该交换的位置

while (isdigit(s[j]) != flag) {

j += 2;

}

// 交换字符

swap(s[i], s[j]);

}

}

return s;

}

};