# 题目

给你一个字符串 s ，根据下述规则反转字符串：

所有非英文字母保留在原有位置。

所有英文字母（小写或大写）位置反转。

返回反转后的 s 。

示例 1：

输入：s = "ab-cd"

输出："dc-ba"

示例 2：

输入：s = "a-bC-dEf-ghIj"

输出："j-Ih-gfE-dCba"

示例 3：

输入：s = "Test1ng-Leet=code-Q!"

输出："Qedo1ct-eeLg=ntse-T!"

提示

1 <= s.length <= 100

s 仅由 ASCII 值在范围 [33, 122] 的字符组成

s 不含 '\"' 或 '\\'

# 分析

## 方法一：双指针

思路：

代码：

class Solution {

public:

string reverseOnlyLetters(string s) {

int n = s.length(); // 获取字符串长度

int left = 0, right = n - 1; // 初始化左右指针，分别指向字符串的开头和结尾

while (true) { // 循环直到条件不满足时退出

// 左指针向右移动，直到遇到字母为止

while (left < right && !isalpha(s[left])) {

left++;

}

// 右指针向左移动，直到遇到字母为止

while (right > left && !isalpha(s[right])) {

right--;

}

// 如果左指针大于等于右指针，说明已经遍历完所有字母，退出循环

if (left >= right) break;

// 交换左右指针所指向的字母

swap(s[left], s[right]);

// 左右指针分别向中间移动一位

left++;

right--;

}

return s; // 返回反转后的字符串

}

};

复杂度分析

时间复杂度：O(n)，其中n是字符串s的长度。反转过程需要O(n)，C 语言计算字符串长度需要O(n)。

空间复杂度：O(1) 或O(n)。某些语言字符串不可变，需要O(n)的额外空间。