# 题目

给你一个字符串 S，返回只含 单一字母 的子串个数。

示例 1：

输入： "aaaba"

输出： 8

解释：

只含单一字母的子串分别是 "aaa"， "aa"， "a"， "b"。

"aaa" 出现 1 次。

"aa" 出现 2 次。

"a" 出现 4 次。

"b" 出现 1 次。

所以答案是 1 + 2 + 4 + 1 = 8。

示例 2:

输入： "aaaaaaaaaa"

输出： 55

提示：

1 <= S.length <= 1000

S[i] 仅由小写英文字母组成。

# 分析

## 方法一：分组计算

**思路：**

本题要求只含单一字母的子串个数，首先我们可以计算每一个最长只含单一字母的连续子串。比如 s = "aaabb"，一共有两个符合要求的子串，分别为 aaa 和 bb。

然后考虑对于一个字符串，共有多少不同的子串。我们可以对字符串进行分组计算，比如aaa共有6个子串。a3个，aa2个，aaa1个。转化为通用公式就是 n \* (n + 1) / 2，其中n为连续子串的长度。

**算法：**

遍历字符串s，计算每个最长只含单一字母的连续子串的长度count，每个最长连续子串可以分成count \* (count + 1) / 2个不同的子串，将其累加到结果中。

**代码：**

int countLetters(char\* s) {

int n = strlen(s);

int ans = 0;

int count = 1;

for (int i = 1; i < n; i++) {

if (s[i-1] == s[i])

count++;

else {

ans += (count \* (count + 1) / 2);

count = 1;

}

}

ans += (count \* (count + 1) / 2);

return ans;

}

复杂度分析：

时间复杂度：O(n)，遍历一次字符串。其中n 为字符串 s 的长度。

空间复杂度：O(1)，没有使用额外的空间。

## 方法二：动态规划

解题思路：

1、动态规划

为什么要用动规呢？

首先思考如果出现aaaaaabb这样的情况，是不是a的个数越多，则字符串的组合情况会越多，这是越靠近b的a越和已经出现的a的个数有关，也就是和之前状态有关，所以用递归

2、用两个变量记录状态，节省空间

代码：

class Solution {

public:

int countLetters(string s) {

if(s.empty())

return 0;

int ans=1,pre,cur=1;

for(int i=1;i<s.size();++i)

{

pre=cur;

cur=1;

if(s[i]==s[i-1])

cur+=pre;

ans+=cur;

}

return ans;

}

};