给定一个字符串，找到它的第一个不重复的字符，并返回它的索引。如果不存在，则返回 -1。

案例:

s = "leetcode"

返回 0.

s = "loveleetcode",

返回 2.

注意事项：您可以假定该字符串只包含小写字母。

===========================================================

class Solution {

public:

int firstUniqChar(string s) {

if (s.length() == 0)

{

return -1;

}

if (s.length() == 1)

{

return 0;

}

int flag = 0;

for (int i = 0; i < s.length(); ++i)

{

for (int j = 0; j < s.length(); ++j)

{

if (i != j && s[i] == s[j] && flag >= i)

{

flag = i + 1;

}

}

}

if (flag >= s.length())

{

return -1;

}

return flag;

}

};

27 / 104 个通过测试用例

状态：超出时间限制

最后执行的输入：

""

===========================================================

class Solution {

public:

int firstUniqChar(string s) {

vector<int> cnt(26,0); // 存放26个字母的个数

for(char c:s){

cnt[c-'a']++; // 字母出现的个数+1

}

for(int i=0; i<s.size(); ++i){

if(cnt[s[i]-'a']==1) // 遍历字符串的每一个字母，判断个数是否为1

return i;

}

return -1;

}

};

104 / 104 个通过测试用例

状态：通过

执行用时：32 ms

<https://blog.csdn.net/ajiangfan/article/details/52620847> First Unique Character in a String（字符串中的第一个唯一字符）

===========================================================

给出一个字符串，找出第一个只出现一次的字符。

class Solution {

public:

char firstUniqChar(string &str) {

unordered\_map<char, int> counts;

int pos = 0;

for(int i = 0; i < str.length(); ++i) {

counts[str[i]]++;

while(counts[str[pos]] > 1) {

pos++;

}

}

return str[pos];

}

};

c++实现的基于hashmap的实现方式

设立一个指针指向目前认为的只出现一次的字符，

每次增加一个counts都重新挪动pos到最靠近开头的只出现一次的字符的位置上