[LeetCode] Isomorphic Strings - 字符串操作:数组计数字符个数问题

题目概述：

Given two strings s and t, determine if they are isomorphic.
Two strings are isomorphic if the characters in s can be replaced to get t.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character but a character may map to itself.

For example,
Given "egg", "add", return true.
Given "foo", "bar", return false.
Given "paper", "title", return true.
Note: You may assume both s and t have the same length.

解题方法：

该题意是判断两个字符串s和t是否是同构的。（注意字符不仅是字母）

最简单的方法是通过计算每个字符出现的个数并且相应位置字母对应，但是两层循环肯定TLE。所以需要通过O(n)时间比较，采用的方法是：

eg： "aba" <> "baa" return false

关键代码：nums[s[i]]=t[i] numt[t[i]]=s[i] 再比较是否相同

nums['a']='b' numt['b']='a' (第一次出现)
nums['b']='a' numt['a']='b' (第一次出现)
nums['a']='b' <> t[2]='a' (第二次出现) return false

该方法技巧性比较强，当然如果你使用C++的映射就非常容易实现了。

我的代码：

```
bool isIsomorphic(char* s, char* t) {
    int ls,lt;      //字符串长度
    int i,j;
    int nums[256]={0};
    int numt[256]={0};

    ls = strlen(s);
    lt = strlen(t);
    if(ls!=lt) return false;
    for(i=0; i<ls; i++) {
        //初值为0
        if(nums[s[i]]==0) {
            if(numt[t[i]]==0) {
                nums[s[i]] = t[i];
                numt[t[i]] = s[i];
            }
            else {
                return false;
            }
        }
        else {
            if(nums[s[i]]!=t[i]) {
                return false;
            }
        }
    }
    return true;
}
```

C++推荐代码：

参考：http://www.cnblogs.com/easonliu/p/4465650.html

题目很简单，也很容易想到方法，就是记录遍历s的每一个字母，并且记录s[i]到t[i]的映射，当发现与已有的映射不同时，说明无法同构，直接return false。但是这样只能保证从s到t的映射，不能保证从t到s的映射，所以交换s与t的位置再重来一遍上述的遍历就OK了。

```
class Solution {
public:
    bool isIsomorphic(string s, string t) {
        if (s.length() != t.length()) return false;
        map<char, char> mp;
```

```
        for (int i = 0; i < s.length(); ++i) {
            if (mp.find(s[i]) == mp.end()) mp[s[i]] = t[i];
            else if (mp[s[i]] != t[i]) return false;
                                                                }
        mp.clear();
        for (int i = 0; i < s.length(); ++i) {
            if (mp.find(t[i]) == mp.end()) mp[t[i]] = s[i];
            else if (mp[t[i]] != s[i]) return false;
        }
        return true;
    }
};
```

(By:Eastmount 2015-9-21 凌晨1点半 http://blog.csdn.net/eastmount/)