LeetCode 题解 (C Plus Plus 版本)

胡欣毅 (icedomain_hu@qq.com)
https://github.com/Icedomain/LeetCode

最后更新 March 4, 2020

```cpp
/*
 * @lc app=leetcode.cn id=1 lang=cpp
 *
 * [1] 两数之和
 */
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        // 哈希表
        unordered_map<int, int>dict;
        vector<int> res ;
        for(int i = 0 ; i <nums.size();i++ ){
            // 找不到
            if (dict.find(target - nums[i]) == dict.end()){
                dict[nums[i]] = i;
            }
            else{
                res.push_back(dict[target-nums[i]]);
                res.push_back(i);
            }
        }
        return res;
    }
};
```

```cpp
/*
 * @lc app=leetcode.cn id=2 lang=cpp
 *
 * [2] 两数相加
 */
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
```

```
 * };
 */
class Solution {
public:
    ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
        int jinwei = 0;
        // 结果
        ListNode *head , *n ;
        head = n = new ListNode(0);
        int v1 ,v2,sum;
        while (l1 != NULL || l1 != NULL || jinwei)
        {
            v1 = v2 = 0;
            if(l1 != NULL){
                v1 = l1->val;
                l1 = l1->next;
            }
            if(l2 != NULL){
                v2 = l2->val;
                l2 = l2->next;
            }
            // 除数、余数
            jinwei = (v1+v2+jinwei)/10;
            sum = (v1+v2+jinwei)%10 ;
            ListNode *node = new ListNode(sum);
            n->next = node;
            // 指向下一个
            n = n->next;
        }
        return head->next;
    }
};
```

```
/*
 * @lc app=leetcode.cn id=3 lang=cpp
 *
 * [3] 无重复字符的最长子串
 */
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        // 记录表 256个字符 填-1
        vector<int> charmap (256,-1);

        int start = 0;
        int maxlen = 0;
```

```
14          // 遍历 滑动窗 [start,j ] j往右边移动 若遇到重复的 start又移一位
15          for(int  j  = 0;j<s.size();j++){
16              // 如果这个字符出现过了,又移动 最左边那个踢出滑动窗
17              if(charmap[s[j]]  >= start)
18                  start  = charmap[s[j]]  + 1;
19              // 如果这个字符在滑动窗中没出现过,位置给它(出现过也要给它)
20              charmap[s[j]]  = j;
21              maxlen = max(maxlen , j - start +1);
22          }
23          return maxlen;
24      }
25  };
```