LeetCode 题解 (C Plus Plus 版本)

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

最后更新 April 23, 2020

本文档一共统计了 8 道题

```cpp
/*
 * @lc app=leetcode.cn id=1 lang=cpp
 *
 * [1] 两数之和
 */
#include <iostream>
#include <vector>
#include <unordered_map>
using namespace std;

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;
    }
};
/*
int main(){
    Solution s;

    std::vector<int> v1{2, 7, 11, 15};
    for(const int& k : s.twoSum(v1, 9))
```

```
        cout << k << " "; // 0, 1
    cout << endl;

    std::vector<int> v2{0, 4, 3, 0};
    for(const int& k : s.twoSum(v2, 0))
        cout << k << " "; // 0, 3
    cout << endl;

    std::vector<int> v3{-3, 4, 3, 90};
    for(const int& k : s.twoSum(v3, 0))
        cout << k << " "; // 0, 2
    cout << endl;

    return 0;
}
*/
```

```
/*
 * @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 dummy(0) ,*tail = &dummy;
        int v1 ,v2,sum;
        while (l1 || l2 || jinwei)
        {
            v1 = v2 = 0;
            if(l1){v1 = l1->val;l1 = l1->next; }
            if(l2){v2 = l2->val;l2 = l2->next; }
            // 除数、余数
            sum = (v1+v2+jinwei)%10 ;
            jinwei = (v1+v2+jinwei)/10;
            tail->next = new ListNode(sum);
```

```
            // 指向下一个
            tail = tail->next;
        }
        return dummy.next;
    }
};
```

```
/*
 * @lc app=leetcode.cn id=3 lang=cpp
 *
 * [3] 无重复字符的最长子串
 */
#include <iostream>
#include <unordered_map>
#include <algorithm>
#include <string>
using namespace std;

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;
        // 遍历 滑动窗 [start,j ] j往右边移动 若遇到重复的 start又移一位
        for(int j = 0;j<s.size();j++){
            // 如果这个字符出现过了,又移动 最左边那个踢出滑动窗
            if(charmap[s[j]] >= start)
                start = charmap[s[j]] + 1;
            // 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)
            charmap[s[j]] = j;
            maxlen = max(maxlen , j - start +1);
        }
        return maxlen;
    }
};
/*
int main(){
    Solution s ;
    cout << s.lengthOfLongestSubstring("abcabcbb"); // 3

}
*/
```

```cpp
/*
 * @lc app=leetcode.cn id=7 lang=cpp
 *
 * [7] 整数反转
 */
#include <climits>
#include <iostream>

class Solution {
public:
    int reverse(int x) {
        long result = 0;
        while(x != 0)
        {
            result = result*10 + x % 10;
            std::cout << result;
            x /= 10;
        }
        return (result > INT_MAX || result < INT_MIN)? 0 : result;
        }
};
/*
int main()
{
    Solution s;
    std::cout << s.reverse(123) << std::endl;
    std::cout << s.reverse(-123) << std::endl;
    std::cout << s.reverse(10100) << std::endl;
    std::cout << s.reverse(1000000003) << std::endl;

    return 0;
}
*/
```

```cpp
/*
 * @lc app=leetcode.cn id=9 lang=cpp
 *
 * [9] 回文数
 */
#include<iostream>
using namespace std;

class Solution {
public:
    bool isPalindrome(int x) {
        if(x<0) return false;
```

```
        // 最高位的位数
        int d = 1;
        while (x / d >= 10){
            d *= 10;
        }
        int p , q;
        while (x > 0){
            // p q 对应最高位和最低位
            p = x / d ;
            q = x % 10;
            if (p != q) return false ;
            // x 去掉最高位,去掉最低位
            x = x % d / 10 ;
            // x 去掉了两位,d也减两位
            d /= 100;
        }
        return true;
    }

    bool isPalindrome2(int x) {
        if (x < 0) return false ;
        long rev = 0, origin = x;
        while (x > 0){
            rev = rev * 10 + x % 10;
            x /= 10;
        }
        return rev == origin;
    }
};
/*
int main(){
    Solution s;
    cout << s.isPalindrome(2002) << endl;
    cout << s.isPalindrome2(2102) << endl;
    return 0;
}
*/
```

```
/*
 * @lc app=leetcode.cn id=69 lang=cpp
 *
 * [69] x 的平方根
 */
#include <iostream>
#include <cmath>
using namespace std;
```

```cpp

class Solution {
public:
    int mySqrt(int x) {
        /*
        long r = x;
        while (r*r > x)
            r = (r + x/r) / 2;
        return (int) r;
        */

        int l = 0, r = x ;
        long mid ,ret ;
        while (l <= r){
            mid = (l + r) >> 1 ;
            if ( mid * mid <= x & x < (mid+1)*(mid+1) )
            {
                ret = mid;
                break ;
            }
            else if (x < mid*mid )
                r = mid ;
            else
                l = mid + 1;
        }
        return (int) mid;
    }
};
/*
int main() {
    Solution s;
    std::cout << s.mySqrt(8) << std::endl;
    std::cout << s.mySqrt(19) << std::endl;
        return 0;
}
*/
```

```cpp
/*
 * @lc app=leetcode.cn id=70 lang=cpp
 *
 * [70] 爬楼梯
 */
#include <iostream>
#include <vector>
#include <functional>
using namespace std;
```

```cpp

class Solution {
public:
    int climbStairs2(int n) {
        if (n < 0)
            return 0;
        else if (n < 3)
            return n;
        int res = 0;
        int a = 1, b = 2;
        for (int i = 2; i < n; i++ )
        {
            res = a + b ;
            a = b ;
            b = res ;
        }
        return b;
    }

    int climbStairs(int n) {
        if (n < 0) return 0;
        vector<int> vec = {1,1};
        if (n > 1) vec.resize(n+1, -1);
        int res = fib(vec , n) ;
        return res;
    }
    int fib(vector<int>& vec, int n){
            if (vec[n] == -1)
                vec[n] = fib(vec ,n-1) + fib(vec ,n-2);
            return vec[n];
        }
};

/*
int main() {
    Solution s;
    std::cout << s.climbStairs(4) << std::endl;
        return 0;
}
*/
```

```cpp
/*
 * @lc app=leetcode.cn id=100 lang=cpp
 *
 * [100] 相同的树
 */
```

```
6
7  /**
8   * Definition for a binary tree node.
9   * struct TreeNode {
10  *     int val;
11  *     TreeNode *left;
12  *     TreeNode *right;
13  *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
14  * };
15  */
16 class Solution {
17 public:
18     bool isSameTree(TreeNode* p, TreeNode* q) {
19         if (!p && !q ) return true;
20         else if (p && q && p->val == q->val)
21             return isSameTree(p->left , q->left) && isSameTree(p->right , q->right);
22         else return false ;
23     }
24 };
```