

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

```
1  /*
2  * @lc app=leetcode.cn id=1 lang=cpp
3  *
4  * [1] 两数之和
5  */
6  #include <iostream>
7  #include <vector>
8  #include <unordered_map>
9  using namespace std;
10
11 class Solution {
12 public:
13     vector<int> twoSum(vector<int>& nums, int target) {
14         // 哈希表
15         unordered_map<int, int> dict;
16         vector<int> res ;
17         for(int i = 0 ; i < nums.size(); i++) {
18             // 找不到
19             if (dict.find(target - nums[i]) == dict.end()){
20                 dict[nums[i]] = i;
21             }
22             else {
23                 res.push_back(dict[target - nums[i]]);
24                 res.push_back(i);
25             }
26         }
27         return res;
28     }
29 };
30 /*
31 int main(){
32     Solution s;
33
34     std::vector<int> v1{2, 7, 11, 15};
35     for(const int& k : s.twoSum(v1, 9))
```

```

36     cout << k << " "; // 0, 1
37     cout << endl;
38
39     std::vector<int> v2{0, 4, 3, 0};
40     for(const int& k : s.twoSum(v2, 0))
41         cout << k << " "; // 0, 3
42     cout << endl;
43
44     std::vector<int> v3{-3, 4, 3, 90};
45     for(const int& k : s.twoSum(v3, 0))
46         cout << k << " "; // 0, 2
47     cout << endl;
48
49     return 0;
50 }
51 */

```

```

1  /*
2   * @lc app=leetcode.cn id=2 lang=c++
3   *
4   * [2] 两数相加
5   */
6  /**
7   * Definition for singly-linked list.
8   * struct ListNode {
9   *     int val;
10    *     ListNode *next;
11    *     ListNode(int x) : val(x), next(NULL) {}
12    * };
13    */
14    class Solution {
15    public:
16        ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
17            int jinwei = 0;
18            // 结果
19            ListNode dummy(0), *tail = &dummy;
20            int v1, v2, sum;
21            while (l1 || l2 || jinwei)
22            {
23                v1 = v2 = 0;
24                if (l1) {v1 = l1->val; l1 = l1->next;}
25                if (l2) {v2 = l2->val; l2 = l2->next;}
26                // 除数、余数
27                sum = (v1+v2+jinwei)%10;
28                jinwei = (v1+v2+jinwei)/10;
29                tail->next = new ListNode(sum);

```

```

30         // 指向下一个
31         tail = tail->next;
32     }
33     return dummy.next;
34 }
35 };

```

```

1  /*
2  * @lc app=leetcode.cn id=3 lang=c++
3  *
4  * [3] 无重复字符的最长子串
5  */
6  #include <iostream>
7  #include <unordered_map>
8  #include <algorithm>
9  #include <string>
10 using namespace std;
11
12 class Solution {
13 public:
14     int lengthOfLongestSubstring(string s) {
15         // 记录表 256个字符 填-1
16         vector<int> charmap (256,-1);
17
18         int start = 0;
19         int maxlen = 0;
20         // 遍历 滑动窗 [start,j ] j往右边移动 若遇到重复的 start又移一位
21         for(int j = 0;j<s.size();j++){
22             // 如果这个字符出现过了,又移动 最左边那个踢出滑动窗
23             if(charmap[s[j]] >= start)
24                 start = charmap[s[j]] + 1;
25             // 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)
26             charmap[s[j]] = j;
27             maxlen = max(maxlen , j - start +1);
28         }
29         return maxlen;
30     }
31 };
32
33 int main(){
34     Solution s ;
35     cout << s.lengthOfLongestSubstring("abcabcbb"); // 3
36
37 }
38 */

```

```

1  /*
2   * @lc app=leetcode.cn id=7 lang=cpp
3   *
4   * [7] 整数反转
5   */
6  #include <climits>
7  #include <iostream>
8
9  class Solution {
10 public:
11     int reverse(int x) {
12         long result = 0;
13         while(x != 0)
14         {
15             result = result*10 + x % 10;
16             std::cout << result;
17             x /= 10;
18         }
19         return (result > INT_MAX || result < INT_MIN)? 0 : result;
20     }
21 };
22 /*
23 int main()
24 {
25     Solution s;
26     std::cout << s.reverse(123) << std::endl;
27     std::cout << s.reverse(-123) << std::endl;
28     std::cout << s.reverse(10100) << std::endl;
29     std::cout << s.reverse(1000000003) << std::endl;
30
31     return 0;
32 }
33 */

```

```

1  /*
2   * @lc app=leetcode.cn id=9 lang=cpp
3   *
4   * [9] 回文数
5   */
6  #include<iostream>
7  using namespace std;
8
9  class Solution {
10 public:
11     bool isPalindrome(int x) {
12         if(x<0) return false;

```

```

13 // 最高位的位数
14 int d = 1;
15 while (x / d >= 10){
16     d *= 10;
17 }
18 int p , q;
19 while (x > 0){
20     // p q 对应最高位和最低位
21     p = x / d ;
22     q = x % 10;
23     if (p != q) return false ;
24     // x 去掉最高位,去掉最低位
25     x = x % d / 10 ;
26     // x 去掉了两位,d也減两位
27     d /= 100;
28 }
29 return true;
30 }
31
32 bool isPalindrome2(int x) {
33     if (x < 0) return false ;
34     long rev = 0, origin = x;
35     while (x > 0){
36         rev = rev * 10 + x % 10;
37         x /= 10;
38     }
39     return rev == origin;
40 }
41 };
42 /*
43 int main(){
44     Solution s;
45     cout << s.isPalindrome(2002) << endl;
46     cout << s.isPalindrome2(2102) << endl;
47     return 0;
48 }
49 */

```

```

1  /*
2   * @lc app=leetcode.cn id=69 lang=cpp
3   *
4   * [69] x 的平方根
5   */
6  #include <iostream>
7  #include <cmath>
8  using namespace std;

```

```

9
10 class Solution {
11 public:
12     int mySqrt(int x) {
13         /*
14         long r = x;
15         while (r*r > x)
16             r = (r + x/r) / 2;
17         return (int) r;
18         */
19
20         int l = 0, r = x ;
21         long mid ,ret ;
22         while (l <= r){
23             mid = (l + r) >> 1 ;
24             if ( mid * mid <= x & x < (mid+1)*(mid+1) )
25             {
26                 ret = mid;
27                 break ;
28             }
29             else if (x < mid*mid )
30                 r = mid ;
31             else
32                 l = mid + 1;
33         }
34         return (int) mid;
35     }
36 };
37 /*
38 int main() {
39     Solution s;
40     std::cout << s.mySqrt(8) << std::endl;
41     std::cout << s.mySqrt(19) << std::endl;
42     return 0;
43 }
44 */

```

```

1  /*
2   * @lc app=leetcode.cn id=70 lang=cpp
3   *
4   * [70] 爬楼梯
5   */
6  #include <iostream>
7  #include <vector>
8  #include <functional>
9  using namespace std;

```

```

10
11 class Solution {
12 public:
13     int climbStairs2(int n) {
14         if (n < 0)
15             return 0;
16         else if (n < 3)
17             return n;
18         int res = 0;
19         int a = 1, b = 2;
20         for (int i = 2; i < n; i++)
21         {
22             res = a + b ;
23             a = b ;
24             b = res ;
25         }
26         return b;
27     }
28
29     int climbStairs(int n) {
30         if (n < 0) return 0;
31         vector<int> vec = {1,1};
32         if (n > 1) vec.resize (n+1, -1);
33         int res = fib(vec , n) ;
34         return res;
35     }
36     int fib(vector<int>& vec, int n){
37         if (vec[n] == -1)
38             vec[n] = fib(vec ,n-1) + fib(vec ,n-2);
39         return vec[n];
40     }
41 };
42
43 /*
44 int main() {
45     Solution s;
46     std::cout << s.climbStairs(4) << std::endl;
47     return 0;
48 }
49 */

```

```

1  /*
2  * @lc app=leetcode.cn id=100 lang=cpp
3  *
4  * [100] 相同的树
5  */

```

```

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 };

```