Contents

1 数组

1.1 两数之和

题目：

给定一个整数数组和一个目标值，找出数组中和为目标值的两个数。

你可以假设每个输入只对应一种答案，且同样的元素不能被重复利用。

示例:

给定 nums = [2, 7, 11, 15], target = 9

因为 nums[0] + nums[1] = 2 + 7 = 9 所以返回 [0, 1]

解法：

用一个 map，key 是元素值，value 是 idx 看新来的这个元素的目标值（tgt - nums[i]）在不在 map 里，在的话把它的 value 拿出来就行了。。

```
class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        vector<int> res;
        unordered_map<int, int> map;
        for (int i = 0; i < nums.size(); ++i) {
            const int& tgt_val = target - nums[i];
            if (map.find(tgt_val) != map.end()) {
                res.push_back(map[tgt_val]);
                res.push_back(i);
                return res;
            } else {
                map.insert(std::make_pair(nums[i], i));
            }
        }
    }
};
```

2 链表

2.1 两数相加

题目：给定两个非空链表来表示两个非负整数。位数按照逆序方式存储，它们的每个节点只存储单个数字。将两数相加返回一个新的链表。

你可以假设除了数字 0 之外，这两个数字都不会以零开头。

示例：

输入：(2 -> 4 -> 3) + (5 -> 6 -> 4)
输出：7 -> 0 -> 8
原因：342 + 465 = 807

解答：

- 搞一个 dummyhead，然后每一次 while 的时候，填充他的 next，最后返回出是 dummyhead 的 next
- 要 x->next 之前先判断 x!=NULL(不是判断 x->next!=NULL)
- while 的条件是或，处理两个链表不等长的情况
- while 外面，如果还有 carry，要再 new 一个

```
/**
 * 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 carry = 0;
        ListNode* dummy_head = new ListNode(0);
        ListNode* tmp = dummy_head;
        ListNode* ptr1 = l1;
        ListNode* ptr2 = l2;
        while (ptr1 != NULL || ptr2 != NULL) {
            int val1 = ptr1 != NULL? ptr1->val: 0;
            int val2 = ptr2 != NULL? ptr2->val: 0;
            int sum = val1 + val2 + carry;
            //cout << sum << " " << carry << " " << val1 << " " << val2 << endl;
            carry = sum / 10;
            int remain = sum % 10;
            tmp->next = new ListNode(remain);
            ptr1 = (NULL == ptr1? NULL: ptr1->next);
            ptr2 = (NULL == ptr2? NULL: ptr2->next);
            tmp = tmp->next;
        }
        if (carry > 0) {
            tmp->next = new ListNode(carry);
        }
        return dummy_head->next;
    }
};
```

3 树

4 图

5 动态规划