[**1. 两数之和**](https://leetcode.cn/problems/two-sum/)

给定一个整数数组 nums 和一个整数目标值 target，请你在该数组中找出 **和为目标值***target*  的那 **两个** 整数，并返回它们的数组下标。

你可以假设每种输入只会对应一个答案。但是，数组中同一个元素在答案里不能重复出现。

你可以按任意顺序返回答案。

**示例 1：**

**输入：**nums = [2,7,11,15], target = 9

**输出：**[0,1]

**解释：**因为 nums[0] + nums[1] == 9 ，返回 [0, 1] 。

**示例 2：**

**输入：**nums = [3,2,4], target = 6

**输出：**[1,2]

**示例 3：**

**输入：**nums = [3,3], target = 6

**输出：**[0,1]

代码：

1.遍历法

class Solution {

    public int[] twoSum(int[] nums, int target) {

        for(int i=0;i<nums.length-1;i++){

            for(int j=i+1;j<nums.length;j++){

                if(nums[i]+nums[j]==target){

                    return new int[]{i,j};

                }

            }

        }

        return null;

    }

}

2. 记录表法

class Solution {

    public int[] twoSum(int[] nums, int target) {

        HashMap<Integer,Integer>h1=new HashMap<Integer,Integer>();

        for(int i=0;i<nums.length;i++){

            if(h1.containsKey(target-nums[i])){

                return new int[]{h1.get(target-nums[i]),i};

            }

             h1.put(nums[i],i);

        }

        return new int[0];

    }

}

[**9. 回文数**](https://leetcode.cn/problems/palindrome-number/)

已解答

简单

相关标签

相关企业

提示

给你一个整数 x ，如果 x 是一个回文整数，返回 true ；否则，返回 false 。

回文数

是指正序（从左向右）和倒序（从右向左）读都是一样的整数。

* 例如，121 是回文，而 123 不是。

**示例 1：**

**输入：**x = 121

**输出：**true

**示例 2：**

**输入：**x = -121

**输出：**false

**解释：**从左向右读, 为 -121 。 从右向左读, 为 121- 。因此它不是一个回文数。

**示例 3：**

**输入：**x = 10

**输出：**false

**解释：**从右向左读, 为 01 。因此它不是一个回文数。

**提示：**

* -231 <= x <= 231 - 1

代码：

class Solution {

    public boolean isPalindrome(int x) {

        if (x < 0) {

            return false;

        } else if (x == 0) {

            return true;

        } else {

            String s = Integer.toString(x);

            char[] a1 = s.toCharArray();

            int ilength = a1.length - 1;

            for (int i = 0; i <= ilength/2; i++) {

                if (a1[i] != a1[ilength - i]) {

                    return false;

                }

            }

            return true;

        }

    }

}

[**70. 爬楼梯**](https://leetcode.cn/problems/climbing-stairs/)

假设你正在爬楼梯。需要 n 阶你才能到达楼顶。

每次你可以爬 1 或 2 个台阶。你有多少种不同的方法可以爬到楼顶呢？

**示例 1：**

**输入：**n = 2

**输出：**2

**解释：**有两种方法可以爬到楼顶。

1. 1 阶 + 1 阶

2. 2 阶

**示例 2：**

**输入：**n = 3

**输出：**3

**解释：**有三种方法可以爬到楼顶。

1. 1 阶 + 1 阶 + 1 阶

2. 1 阶 + 2 阶

3. 2 阶 + 1 阶

**提示：**

* 1 <= n <= 45

代码：

class Solution {

    public int climbStairs(int n) {

         int p=0;

         int q=0;

         int r=1;

         for(int i=1;i<=n;i++){

            p=q;

            q=r;

            r=p+q;

         }

         return r;

    }

}

[**3. 无重复字符的最长子串**](https://leetcode.cn/problems/longest-substring-without-repeating-characters/)

已解答

中等

相关标签

相关企业

给定一个字符串 s ，请你找出其中不含有重复字符的 **最长**

**子串**

的长度。

**示例 1:**

**输入:** s = "abcabcbb"

**输出:** 3

**解释:** 因为无重复字符的最长子串是 "abc"，所以其长度为 3。

**示例 2:**

**输入:** s = "bbbbb"

**输出:** 1

**解释:** 因为无重复字符的最长子串是 "b"，所以其长度为 1。

**示例 3:**

**输入:** s = "pwwkew"

**输出:** 3

**解释:** 因为无重复字符的最长子串是 "wke"，所以其长度为 3。

  请注意，你的答案必须是 **子串** 的长度，"pwke" 是一个子序列，不是子串。

**提示：**

* 0 <= s.length <= 5 \* 104
* s 由英文字母、数字、符号和空格组成

代码：

**class** Solution\_4{

**public** **int** lengthOfLongestSubstring(String s) {

**if** (s == **null** || s.length() == 0) **return** 0;

//字符串为空

//非空

HashMap<Character, Integer>h1= **new** HashMap<>();

**int** max = 0;

**int** start = 0;

**for** (**int** i= 0;i< s.length();i++) {

**char** currentChar = s.charAt(i);

**if** (h1.containsKey(currentChar)) {

start = Math.*max*(start, h1.get(currentChar) + 1);

}

h1.put(currentChar,i);

max = Math.*max*(max, i-start+1);

}

**return** max;

}

}

[**7. 整数反转**](https://leetcode.cn/problems/reverse-integer/)

已解答

中等

相关标签

相关企业

给你一个 32 位的有符号整数 x ，返回将 x 中的数字部分反转后的结果。

如果反转后整数超过 32 位的有符号整数的范围 [−231,  231− 1] ，就返回 0。

**假设环境不允许存储 64 位整数（有符号或无符号）。**

**示例 1：**

**输入：**x = 123

**输出：**321

**示例 2：**

**输入：**x = -123

**输出：**-321

**示例 3：**

**输入：**x = 120

**输出：**21

**示例 4：**

**输入：**x = 0

**输出：**0

**提示：**

* -231 <= x <= 231 - 1

代码：

class Solution {

    public int reverse(int x) {

     int reversed = 0;

    while (x != 0) {

        int digit = x % 10;

        x /= 10;

        if (reversed > Integer.MAX\_VALUE / 10 || (reversed == Integer.MAX\_VALUE / 10 && digit > 7)) return 0;

        if (reversed < Integer.MIN\_VALUE / 10 || (reversed == Integer.MIN\_VALUE / 10 && digit < -8)) return 0;

        reversed = reversed \* 10 + digit;

    }

    return reversed;

    }

}

[**94. 二叉树的中序遍历**](https://leetcode.cn/problems/binary-tree-inorder-traversal/)

已解答

简单

相关标签

相关企业

给定一个二叉树的根节点 root ，返回 它的 ***中序*** 遍历 。

**示例 1：**



**输入：**root = [1,null,2,3]

**输出：**[1,3,2]

**示例 2：**

**输入：**root = []

**输出：**[]

**示例 3：**

**输入：**root = [1]

**输出：**[1]

**提示：**

* 树中节点数目在范围 [0, 100] 内
* -100 <= Node.val <= 100

代码：# Definition for a binary tree node.

# class TreeNode(object):

#     def \_\_init\_\_(self, val=0, left=None, right=None):

#         self.val = val

#         self.left = left

#         self.right = right

class Solution(object):

    def inorderTraversal(self, root):

        """

        :type root: TreeNode

        :rtype: List[int]

        """

        if root is None:

            return []

        return self.inorderTraversal(root.left)+[root.val]+self.inorderTraversal(root.right);