

1. 两数之和

难度 简单 | 6097 | 收藏 | 分享 | 切换为英文

通过次数 521,701 | 提交次数 1,121,354

[题目描述](#)

[评论 \(2.7k\)](#)

[题解\(275\) New](#)

[提交记录](#)

i Python3 ▾

i { } < > ⌂ ⌃ ⌄

给定一个整数数组 `nums` 和一个目标值 `target`，请你在该数组中找出和为目标值的那 **两个** 整数，并返回他们的数组下标。

你可以假设每种输入只会对应一个答案。但是，你不能重复利用这个数组中同样的元素。

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]`

在真实的面试中遇到过这道题?

是

否

```
1 class Solution:
2     def twoSum(self, nums: List[int], target: int) -> List[int]:
```

贡献者



相关企业 



相关标签



相似题目



[题目列表](#)

[随机一题](#)

[上一题](#)

1/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



2. 两数相加

难度 中等 2934 收藏 分享 切换为英文

通过次数 192,050 提交次数 541,836

[题目描述](#)[评论 \(1.9k\)](#)[题解\(290\) New](#)[提交记录](#)

给出两个 **非空** 的链表用来表示两个非负的整数。其中，它们各自的位数是按照 **逆序** 的方式存储的，并且它们的每个节点只能存储一位数字。

如果，我们将这两个数相加起来，则会返回一个新的链表来表示它们的和。

您可以假设除了数字 0 之外，这两个数都不会以 0 开头。

示例：

输入: (2 → 4 → 3) + (5 → 6 → 4)

输出: 7 → 0 → 8

原因: 342 + 465 = 807

在真实的面试中遇到过这道题?

[是](#)[否](#)

贡献者



相关企业 锁



相关标签



相似题目



i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def addTwoNumbers(self, l1: ListNode, l2: ListNode) ->
9         ListNode:

```

[题目列表](#)[随机一题](#)[上一题](#)

2/1158

[下一题](#)[控制台](#)[贡献 i](#)[▶ 执行代码](#)[提交](#)



4. 寻找两个有序数组的中位数

难度 困难 1442 收藏 分享 切换为英文

通过次数 83,751 提交次数 233,571

[题目描述](#)
[评论 \(1.1k\)](#)
[题解 \(120\) New](#)
[提交记录](#)
i Python3


```

1 class Solution:
2     def findMedianSortedArrays(self, nums1: List[int], nums2: List[int]) -> float:
3

```

给定两个大小为 m 和 n 的有序数组 `nums1` 和 `nums2`。

请你找出这两个有序数组的中位数，并且要求算法的时间复杂度为 $O(\log(m + n))$ 。

你可以假设 `nums1` 和 `nums2` 不会同时为空。

示例 1:

```

nums1 = [1, 3]
nums2 = [2]

```

则中位数是 2.0

示例 2:

```

nums1 = [1, 2]
nums2 = [3, 4]

```

则中位数是 $(2 + 3)/2 = 2.5$

在真实的面试中遇到过这道题?

贡献者



相关企业


[题目列表](#)
[随机一题](#)
[上一题](#)

4/1158

[下一题](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



5. 最长回文子串

难度 中等 1189 收藏 分享 切换为英文

通过次数 96,150 提交次数 358,744

题目描述

评论 (975)

题解(110) New

提交记录

i Python3 ▾



```
1 class Solution:
2     def longestPalindrome(self, s: str) -> str:
3         .
```

给定一个字符串 s ，找到 s 中最长的回文子串。你可以假设 s 的最大长度为 1000。

示例 1:

输入: "babad"

输出: "bab"

注意: "aba" 也是一个有效答案。

示例 2:

输入: "cbbd"

输出: "bb"

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业 锁



相关标签



相似题目



显示提示1



题目列表

随机一题

上一题

5/1158

下一题

控制台

贡献 i

执行代码

提交



6. Z 字形变换

难度 中等 367 收藏 分享 切换为英文

[题目描述](#) [评论 \(520\)](#) [题解\(104\) New](#) [提交记录](#)

将一个给定字符串根据给定的行数，以从上往下、从左到右进行 Z 字形排列。

比如输入字符串为 "LEETCODEISHIRING" 行数为 3 时，排列如下：

```

L   C   I   R
E T O E S I I G
E   D   H   N

```

之后，你的输出需要从左往右逐行读取，产生出一个新的字符串，比如："LCIRETOESIIGEDHN"。

请你实现这个将字符串进行指定行数变换的函数：

```
string convert(string s, int numRows);
```

示例 1:

输入：s = "LEETCODEISHIRING", numRows = 3
输出："LCIRETOESIIGEDHN"

示例 2:

输入：s = "LEETCODEISHIRING", numRows = 4
输出："LDREOEIIECIHNTSG"
解释：

i Python3

```

1 class Solution:
2     def convert(self, s: str, numRows: int) -> str:
3

```

[题目列表](#)

[随机一题](#)

[上一题](#)

6/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)

7. 整数反转

难度 简单 | 1259 | [收藏](#) | [分享](#) | [切换为英文](#)

通过次数 171,601 提交次数 521,346

[题目描述](#)

[评论 \(1.5k\)](#)

[题解\(185\) New](#)

[提交记录](#)

i Python3 ▾

i { } < > ⌂ ⌃ ⌁

```
1 class Solution:
2     def reverse(self, x: int) -> int:
3         .
```

给出一个 32 位的有符号整数，你需要将这个整数中每位上的数字进行反转。

示例 1:

输入：123

输出：321

示例 2:

输入：-123

输出：-321

示例 3:

输入：120

输出：21

注意:

假设我们的环境只能存储得下 32 位的有符号整数，则其数值范围为 $[-2^{31}, 2^{31} - 1]$ 。请根据这个假设，如果反转后整数溢出那么就返回 0。

在真实的面试中遇到过这道题？

是

否

二三五

[题目列表](#)

[随机一题](#)

[上一题](#)

7/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



8. 字符串转换整数 (atoi)

难度 中等 402 收藏 分享 切换为英文

通过次数 63,514 提交次数 355,587

[题目描述](#)
[评论 \(918\)](#)
[题解\(154\) New](#)
[提交记录](#)
i Python3


```
1 class Solution:
2     def myAtoi(self, str: str) -> int:
```

请你来实现一个 `atoi` 函数，使其能将字符串转换成整数。

首先，该函数会根据需要丢弃无用的开头空格字符，直到寻找到第一个非空格的字符为止。

当我们寻找到的第一个非空字符为正或者负号时，则将该符号与之后面尽可能多的连续数字组合起来，作为该整数的正负号；假如第一个非空字符是数字，则直接将其与之后连续的数字字符组合起来，形成整数。

该字符串除了有效的整数部分之后也可能存在多余的字符，这些字符可以被忽略，它们对于函数不应该造成影响。

注意：假如该字符串中的第一个非空格字符不是一个有效整数字符、字符串为空或字符串仅包含空白字符时，则你的函数不需要进行转换。

在任何情况下，若函数不能进行有效的转换时，请返回 0。

说明：

假设我们的环境只能存储 32 位大小的有符号整数，那么其数值范围为 $[-2^{31}, 2^{31} - 1]$ 。如果数值超过这个范围，请返回 `INT_MAX` ($2^{31} - 1$) 或 `INT_MIN` (-2^{31})。

示例 1：

输入： "42"

输出： 42

示例 2：

[题目列表](#)
[随机一题](#)
[上一题](#)

8/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



9. 回文数

难度 简单 | 山 711 | ⚡ | ❤ 收藏 | ⚡ 分享 | ⚡ 切换为英文

通过次数 151,912 提交次数 268,759

[题目描述](#)

[评论 \(1.2k\)](#)

[题解\(163\) New](#)

[提交记录](#)

i Python3

```
1 class Solution:
2     def isPalindrome(self, x: int) -> bool:
3         .
```

判断一个整数是否是回文数。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。

示例 1:

输入: 121

输出: true

示例 2:

输入: -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: 10

输出: false

解释: 从右向左读, 为 01 。因此它不是一个回文数。

进阶:

你能不将整数转为字符串来解决这个问题吗?

在真实的面试中遇到过这道题?

是

否

[题目列表](#)

[随机一题](#)

[上一题](#)

9/1158

[下一题](#) >

控制台

贡献 i

▶ 执行代码

提交



10. 正则表达式匹配

难度 困难 612 收藏 分享 切换为英文

通过次数 26,571 提交次数 108,397

[题目描述](#)
[评论 \(510\)](#)
[题解\(43\) New](#)
[提交记录](#)
i Python3

i { } < > ⌂ ⌃ ⌄

给你一个字符串 s 和一个字符规律 p ，请你来实现一个支持
'.' 和 '*' 的正则表达式匹配。

'.' 匹配任意单个字符

'*' 匹配零个或多个前面的那一个元素

所谓匹配，是要涵盖 整个 字符串 s 的，而不是部分字符串。

说明:

- s 可能为空，且只包含从 $a-z$ 的小写字母。
- p 可能为空，且只包含从 $a-z$ 的小写字母，以及字符 $.$ 和 $*$ 。

示例 1:

输入：

```
s = "aa"
```

```
p = "a"
```

输出：false

解释："a" 无法匹配 "aa" 整个字符串。

```
1 class Solution:
2     def isMatch(self, s: str, p: str) -> bool:
3         .
```

示例 2:

输入：

```
s = "aa"
```

```
p = "a*"
```

[题目列表](#)
[随机一题](#)
[上一题](#)

10/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



11. 盛最多水的容器

难度 中等 748 收藏 分享 切换为英文

[题目描述](#)

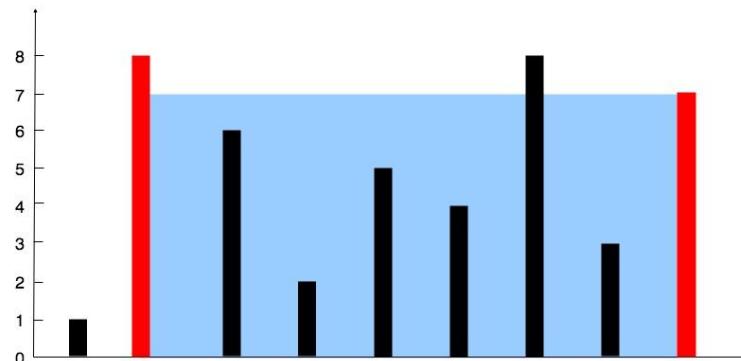
[评论 \(400\)](#)

[题解\(72\) New](#)

[提交记录](#)

给定 n 个非负整数 a_1, a_2, \dots, a_n , 每个数代表坐标中的一个点 (i, a_i) 。在坐标内画 n 条垂直线，垂直线 i 的两个端点分别为 (i, a_i) 和 $(i, 0)$ 。找出其中的两条线，使得它们与 x 轴共同构成的容器可以容纳最多的水。

说明：你不能倾斜容器，且 n 的值至少为 2。



图中垂直线代表输入数组 [1,8,6,2,5,4,8,3,7]。在此情况下，容器能够容纳水（表示为蓝色部分）的最大值为 49。

示例:

输入：[1,8,6,2,5,4,8,3,7]

输出：49

在真实的面试中遇到过这道题？

是

否

Python3

```
1 class Solution:
2     def maxArea(self, height: List[int]) -> int:
3         pass
```

[题目列表](#)

[随机一题](#)

[上一题](#)

11/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



12. 整数转罗马数字

难度 中等 188 收藏 分享 切换为英文

通过次数 35,776 提交次数 59,025

[题目描述](#) [评论 \(387\)](#) [题解\(68\) New](#) [提交记录](#)

i Python3

```
1 class Solution:
2     def intToRoman(self, num: int) -> str:
3         ...
```

罗马数字包含以下七种字符： I , V , X , L , C , D 和 M 。

字符	数值
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

例如，罗马数字 2 写做 II ，即为两个并列的 1。12 写做 XII ，即为 X + II 。27 写做 XXVII ，即为 XX + V + II 。

通常情况下，罗马数字中小的数字在大的数字的右边。但也存在特例，例如 4 不写做 IIII ，而是 IV 。数字 1 在数字 5 的左边，所表示的数等于大数 5 减小数 1 得到的数值 4 。同样地，数字 9 表示为 IX 。这个特殊的规则只适用于以下六种情况：

- I 可以放在 V (5) 和 X (10) 的左边，来表示 4 和 9。
- X 可以放在 L (50) 和 C (100) 的左边，来表示 40 和 90。
- C 可以放在 D (500) 和 M (1000) 的左边，来表示 400 和 900。

给定一个整数，将其转为罗马数字。输入确保在 1 到 3999 的范围

控制台 贡献 i

[题目列表](#)

[随机一题](#)

[上一题](#)

12/1158

[下一题](#)

[执行代码](#)

[提交](#)



13. 罗马数字转整数

难度 简单 570 收藏 分享 切换为英文

通过次数 88,294 提交次数 149,921

[题目描述](#)[评论 \(1.1k\)](#)[题解 \(192\) Ne](#)[提交记录](#)

Python3

```
1 class Solution:
2     def romanToInt(self, s: str) -> int:
3
```

罗马数字包含以下七种字符: I , V , X , L , C , D 和 M 。

字符	数值
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

例如，罗马数字 2 写做 II ，即为两个并列的 1。12 写做 XIII ，即为 X + III 。27 写做 XXVII ，即为 XX + V + II 。

通常情况下，罗马数字中小的数字在大的数字的右边。但也存在特例，例如 4 不写做 IIII ，而是 IV 。数字 1 在数字 5 的左边，所表示的数等于大数 5 减小数 1 得到的数值 4。同样地，数字 9 表示为 IX 。这个特殊的规则只适用于以下六种情况：

- I 可以放在 V (5) 和 X (10) 的左边，来表示 4 和 9。
- X 可以放在 L (50) 和 C (100) 的左边，来表示 40 和 90。
- C 可以放在 D (500) 和 M (1000) 的左边，来表示 400 和 900。

给定一个罗马数字，将其转换成整数。输入确保在 1 到 3999 的范围内。

控制台 ▾ 贡献 i

[题目列表](#)[随机一题](#)[上一题](#)

13/1158

[下一题](#) >[▶ 执行代码](#)[提交](#)



14. 最长公共前缀

难度 简单 673 收藏 分享 切换为英文

通过次数 113,292 提交次数 327,813

[题目描述](#)

[评论 \(914\)](#)

[题解\(116\) New](#)

[提交记录](#)

i Python3



```
1 class Solution:
2     def longestCommonPrefix(self, strs: List[str]) -> str:
3         pass
```

编写一个函数来查找字符串数组中的最长公共前缀。

如果不存在公共前缀，返回空字符串 ""。

示例 1:

输入: ["flower", "flow", "flight"]

输出: "fl"

示例 2:

输入: ["dog", "racecar", "car"]

输出: ""

解释: 输入不存在公共前缀。

说明:

所有输入只包含小写字母 a-z。

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



[题目列表](#)

[随机一题](#)

[上一题](#)

14/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交



15. 三数之和

难度 中等 1281 收藏 分享 切换为英文

通过次数 85,675 提交次数 361,333

[题目描述](#)[评论 \(787\)](#)[题解\(96\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def threeSum(self, nums: List[int]) -> List[List[int]]:
3

```

给定一个包含 n 个整数的数组 nums ，判断 nums 中是否存在三个元素 a, b, c ，使得 $a + b + c = 0$ ？找出所有满足条件且不重复的三元组。

注意：答案中不可以包含重复的三元组。

例如，给定数组 $\text{nums} = [-1, 0, 1, 2, -1, -4]$ ，

满足要求的三元组集合为：

```

[
    [-1, 0, 1],
    [-1, -1, 2]
]

```

在真实的面试中遇到过这道题？

[是](#)[否](#)

贡献者



相关企业



相关标签



相似题目

[题目列表](#)[随机一题](#)[上一题](#)

15/1158

[下一题](#)

控制台

贡献 i [执行代码](#)[提交](#)



16. 最接近的三数之和

难度 中等 236 收藏 分享 切换为英文

通过次数 38,803 提交次数 93,482

[题目描述](#) [评论 \(226\)](#) [题解\(48\) New](#) [提交记录](#)

给定一个包括 n 个整数的数组 nums 和一个目标值 target 。找出 nums 中的三个整数，使得它们的和与 target 最接近。返回这三个数的和。假定每组输入只存在唯一答案。

例如，给定数组 $\text{nums} = [-1, 2, 1, -4]$ ，和 $\text{target} = 1$ 。

与 target 最接近的三个数的和为 2. ($-1 + 2 + 1 = 2$)。

在真实的面试中遇到过这道题？

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```
1 class Solution:
2     def threeSumClosest(self, nums: List[int], target: int) ->
3         int:
```

[题目列表](#)

[随机一题](#)

[上一题](#)

16/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



17. 电话号码的字母组合

难度 中等 429 收藏 分享 切换为英文

通过次数 40,109 提交次数 79,062

[题目描述](#)[评论 \(516\)](#)[题解\(87\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def letterCombinations(self, digits: str) -> List[str]:
3
.
```

给定一个仅包含数字 2-9 的字符串，返回所有它能表示的字母组合。

给出数字到字母的映射如下（与电话按键相同）。注意 1 不对应任何字母。



示例:

输入: "23"
输出: ["ad", "ae", "af", "bd", "be", "bf", "cd", "ce", "cf"].

说明:

尽管上面的答案是按字典序排列的，但是你可以任意选择答案输出的顺序。

在真实的面试中遇到过这道题?

[是](#)[否](#)[题目列表](#)[随机一题](#)[上一题](#)

17/1158

[下一题](#) >

控制台 ▾

贡献 i

[▶ 执行代码](#)[提交](#)



18. 四数之和

难度 中等 270 收藏 分享 切换为英文

通过次数 提交次数

28,499 79,159

[题目描述](#)

[评论 \(226\)](#)

[题解\(51\) New](#)

[提交记录](#)

i Python3



```

1 class Solution:
2     def fourSum(self, nums: List[int], target: int) ->
3         List[List[int]]:

```

给定一个包含 n 个整数的数组 nums 和一个目标值 target ，判断 nums 中是否存在四个元素 a, b, c 和 d ，使得 $a + b + c + d$ 的值与 target 相等？找出所有满足条件且不重复的四元组。

注意：

答案中不可以包含重复的四元组。

示例：

给定数组 $\text{nums} = [1, 0, -1, 0, -2, 2]$ ，和 $\text{target} = 0$ 。

满足要求的四元组集合为：

```
[
    [-1, 0, 0, 1],
    [-2, -1, 1, 2],
    [-2, 0, 0, 2]
]
```

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



[题目列表](#)

[随机一题](#)

< 上一题

18/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



19. 删除链表的倒数第N个节点

难度 中等 496 收藏 分享 切换为英文

通过次数 66,713 提交次数 189,312

[题目描述](#)

[评论 \(598\)](#)

[题解\(105\) New](#)

[提交记录](#)

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def removeNthFromEnd(self, head: ListNode, n: int) ->
9         ListNode:

```

给定一个链表，删除链表的倒数第 n 个节点，并且返回链表的头结点。

示例：

给定一个链表：1->2->3->4->5，和 $n = 2$.

当删除了倒数第二个节点后，链表变为 1->2->3->5.

说明：

给定的 n 保证是有效的。

进阶：

你能尝试使用一趟扫描实现吗？

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



显示提示1



[题目列表](#)

[随机一题](#)

[上一题](#)

19/1158

[下一题](#)

控制台

贡献 *i*

▶ 执行代码

提交



20. 有效的括号

难度 简单 1007 收藏 分享 切换为英文

通过次数 113,150 提交次数 286,938

[题目描述](#)[评论 \(1.0k\)](#)[题解\(165\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def isValid(self, s: str) -> bool:
3

```

给定一个只包括 '(', ')', '{', '}', '[', ']' 的字符串，判断字符串是否有效。

有效字符串需满足：

1. 左括号必须用相同类型的右括号闭合。
2. 左括号必须以正确的顺序闭合。

注意空字符串可被认为是有效字符串。

示例 1:

输入: "()"
输出: true

⋮
⋮
⋮

示例 2:

输入: "()"[]{}"
输出: true

示例 3:

输入: "[]"
输出: false

示例 4:

输入: "...()"

[题目列表](#)

[随机一题](#)

[上一题](#)

20/1158

[下一题](#)

[控制台](#)

贡献 *i*

[▶ 执行代码](#)

[提交](#)



21. 合并两个有序链表

难度 简单 592 收藏 分享 切换为英文

通过次数 102,153 提交次数 179,925

[题目描述](#)

[评论 \(587\)](#)

[题解 \(97\) ^{New}](#)

[提交记录](#)

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def mergeTwoLists(self, l1: ListNode, l2: ListNode) ->
9         ListNode:

```

将两个有序链表合并为一个新的有序链表并返回。新链表是通过拼接给定的两个链表的所有节点组成的。

示例:

输入: 1->2->4, 1->3->4

输出: 1->1->2->3->4->4

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



相似题目



[题目列表](#)

[随机一题](#)

[上一题](#)

21/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交



22. 括号生成

难度 中等 491 收藏 分享 切换为英文

通过次数 37,373 提交次数 52,223

[题目描述](#)[评论 \(436\)](#)[题解\(87\) New](#)[提交记录](#)

Python3

给出 n 代表生成括号的对数，请你写出一个函数，使其能够生成所有可能的并且有效的括号组合。

例如，给出 $n = 3$ ，生成结果为：

```
[  
    "((()))",  
    "(()())",  
    "((())()",  
    "()(())",  
    "()()()"  
]
```

在真实的面试中遇到过这道题？

[是](#)[否](#)

贡献者



相关企业



相关标签



相似题目



```
1 class Solution:  
2     def generateParenthesis(self, n: int) -> List[str]:  
3         ...  
...  
...
```

[题目列表](#)[随机一题](#)[上一题](#)

22/1158

[下一题](#)[控制台](#)[贡献 i](#)[▶ 执行代码](#)[提交](#)



23. 合并K个排序链表

难度 困难 319 收藏 分享 切换为英文

通过次数 38,407 提交次数 81,006

[题目描述](#) [评论 \(387\)](#) [题解\(65\) New](#) [提交记录](#)

合并 k 个排序链表，返回合并后的排序链表。请分析和描述算法的复杂度。

示例:

输入：

```
[1->4->5,
1->3->4,
2->6]
```

输出：1->1->2->3->4->4->5->6

在真实的面试中遇到过这道题？

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def mergeKLists(self, lists: List[ListNode]) -> ListNode:
9

```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)

[随机一题](#)

[上一题](#)

23/1158

[下一题](#)

控制台

贡献 i

[执行代码](#)

[提交](#)



24. 两两交换链表中的节点

难度 中等 268 收藏 分享 切换为英文

通过次数 提交次数

36,280 58,666

[题目描述](#)

[评论 \(421\)](#)

[题解\(70\) New](#)

[提交记录](#)

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def swapPairs(self, head: ListNode) -> ListNode:
9

```

给定一个链表，两两交换其中相邻的节点，并返回交换后的链表。

你不能只是单纯的改变节点内部的值，而是需要实际的进行节点交换。

示例:

给定 $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ ，你应该返回 $2 \rightarrow 1 \rightarrow 4 \rightarrow 3$.

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



相似题目



[题目列表](#)

[随机一题](#)

[上一题](#)

24/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交



26. 删除排序数组中的重复项

难度 简单 1037 收藏 分享 切换为英文

通过次数 155,026 提交次数 337,301

[题目描述](#)

[评论 \(1.1k\)](#)

[题解\(102\) New](#)

[提交记录](#)

i Python3

```
1 class Solution:
2     def removeDuplicates(self, nums: List[int]) -> int:
3         .
```

给定一个排序数组，你需要在原地删除重复出现的元素，使得每个元素只出现一次，返回移除后数组的新长度。

不要使用额外的数组空间，你必须在原地修改输入数组并在使用 O(1) 额外空间的条件下完成。

示例 1:

给定数组 `nums = [1,1,2]`,

函数应该返回新的长度 **2**，并且原数组 `nums` 的前两个元素被修改为 **1, 2**。

你不需要考虑数组中超出新长度后面的元素。

示例 2:

给定 `nums = [0,0,1,1,1,2,2,3,3,4]`,

函数应该返回新的长度 **5**，并且原数组 `nums` 的前五个元素被修改为 **0, 1, 2, 3, 4**。

你不需要考虑数组中超出新长度后面的元素。

说明:

为什么返回数值是整数，但输出的答案是数组呢？

[题目列表](#)

[随机一题](#)

[上一题](#)

26/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交



28. 实现 strStr()

难度 简单 251 收藏 分享 切换为英文

通过次数 76.093 提交次数 196,201

[题目描述](#)

[评论 \(804\)](#)

[题解\(120\) New](#)

[提交记录](#)

i Python3

```
1 class Solution:
2     def strStr(self, haystack: str, needle: str) -> int:
3         pass
```

实现 strStr() 函数。

给定一个 haystack 字符串和一个 needle 字符串，在 haystack 字符串中找出 needle 字符串出现的第一个位置 (从0开始)。如果不存在，则返回 -1。

示例 1:

输入: haystack = "hello", needle = "ll"

输出: 2

示例 2:

输入: haystack = "aaaaa", needle = "bba"

输出: -1

说明:

当 needle 是空字符串时，我们应当返回什么值呢？这是一个在面试中很好的问题。

对于本题而言，当 needle 是空字符串时我们应当返回 0 。这与C语言的 strstr() 以及 Java的 indexOf() 定义相符。

在真实的面试中遇到过这道题？

是

否

贡献者



控制台 贡献 i

题目列表

随机一题

< 上一题

28/1158

下一题 >

▶ 执行代码

提交



29. 两数相除

难度 中等 168 收藏 分享 切换为英文

[题目描述](#) [评论 \(251\)](#) [题解\(53\) New](#) [提交记录](#)

给定两个整数，被除数 `dividend` 和除数 `divisor`。将两数相除，要求不使用乘法、除法和 mod 运算符。

返回被除数 `dividend` 除以除数 `divisor` 得到的商。

示例 1:

输入: `dividend = 10, divisor = 3`

输出: 3

示例 2:

输入: `dividend = 7, divisor = -3`

输出: -2

说明:

- 被除数和除数均为 32 位有符号整数。
- 除数不为 0。
- 假设我们的环境只能存储 32 位有符号整数，其数值范围是 $[-2^{31}, 2^{31} - 1]$ 。本题中，如果除法结果溢出，则返回 $2^{31} - 1$ 。

在真实的面试中遇到过这道题?

是

否

贡献者



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

29/1158

[下一题](#)

控制台

贡献 *i*

[执行代码](#)

[提交](#)



31. 下一个排列

难度 中等 259 收藏 分享 切换为英文

通过次数 20,708 提交次数 65,105

[题目描述](#)
[评论 \(285\)](#)
[题解\(43\) ^{New}](#)
[提交记录](#)
i Python3

```

1 class Solution:
2     def nextPermutation(self, nums: List[int]) -> None:
3         """
4             Do not return anything, modify nums in-place instead.
5         """
6

```

实现获取下一个排列的函数，算法需要将给定数字序列重新排列成字典序中下一个更大的排列。

如果不存在下一个更大的排列，则将数字重新排列成最小的排列（即升序排列）。

必须原地修改，只允许使用额外常数空间。

以下是一些例子，输入位于左侧列，其相应输出位于右侧列。

1, 2, 3 → 1, 3, 2

3, 2, 1 → 1, 2, 3

1, 1, 5 → 1, 5, 1

在真实的面试中遇到过这道题？

[是](#)
[否](#)

贡献者



相关企业



相关标签



相似题目



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)


[题目列表](#)
[随机一题](#)
[上一题](#)

31/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



32. 最长有效括号

难度 困难 311 收藏 分享 切换为英文

通过次数 18,364 提交次数 65,350

题目描述

评论 (164)

题解(35) New

提交记录

i Python3

```
1 class Solution:
2     def longestValidParentheses(self, s: str) -> int:
3         pass
```

给定一个只包含 '(' 和 ')' 的字符串，找出最长的包含有效括号的子串的长度。

示例 1:

输入: "(()"

输出: 2

解释: 最长有效括号子串为 "()"

示例 2:

输入: ")()())"

输出: 4

解释: 最长有效括号子串为 "()()"

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业

相关标签

相似题目

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

题目列表

随机一题

< 上一题

32/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



33. 搜索旋转排序数组

难度 中等 337 收藏 分享 切换为英文

[题目描述](#)
[评论 \(397\)](#)
[题解\(69\) New](#)
[提交记录](#)
i Python3

```
1 class Solution:
2     def search(self, nums: List[int], target: int) -> int:
3         pass
```

假设按照升序排序的数组在预先未知的某个点上进行了旋转。

(例如，数组 `[0, 1, 2, 4, 5, 6, 7]` 可能变为 `[4, 5, 6, 7, 0, 1, 2]`)。

搜索一个给定的目标值，如果数组中存在这个目标值，则返回它的索引，否则返回 `-1`。

你可以假设数组中不存在重复的元素。

你的算法时间复杂度必须是 $O(\log n)$ 级别。

示例 1:

输入: `nums = [4,5,6,7,0,1,2]`, `target = 0`

输出: 4

示例 2:

输入: `nums = [4,5,6,7,0,1,2]`, `target = 3`

输出: -1

在真实的面试中遇到过这道题?

贡献者



相关企业



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)


[题目列表](#)
[随机一题](#)
[上一题](#)

33/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



35. 搜索插入位置

难度 简单 305 收藏 分享 切换为英文

通过次数 提交次数

67,030 151,650

[题目描述](#)
[评论 \(670\)](#)
[题解\(93\) ^{New}](#)
[提交记录](#)
i Python3


```
1 class Solution:
2     def searchInsert(self, nums: List[int], target: int) -> int:
3         pass
```

给定一个排序数组和一个目标值，在数组中找到目标值，并返回其索引。如果目标值不存在于数组中，返回它将会被按顺序插入的位置。

你可以假设数组中无重复元素。

示例 1:

输入: [1,3,5,6], 5

输出: 2

示例 2:

输入: [1,3,5,6], 2

输出: 1

示例 3:

输入: [1,3,5,6], 7

输出: 4

示例 4:

输入: [1,3,5,6], 0

输出: 0

[题目列表](#)
[随机一题](#)
[上一题](#)

35/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)

36. 有效的数独

难度 中等 180 收藏 分享 切换为英文

通过次数 提交次数

34,171 61,666

[题目描述](#)

[评论 \(259\)](#)

[题解\(34\) New](#)

[提交记录](#)

i Python3

i { } < > ⌂ ⌃ ⌄

判断一个 9x9 的数独是否有效。只需要根据以下规则，验证已经填入的数字是否有效即可。

1. 数字 1-9 在每一行只能出现一次。
2. 数字 1-9 在每一列只能出现一次。
3. 数字 1-9 在每一个以粗实线分隔的 3x3 宫内只能出现一次。

5	3	.	.	7
6	.	.	1	9	5	.	.	.
.	9	8	6
8	.	.	6	3
4	.	.	8	3	.	.	.	1
7	.	.	2	6
.	6	.	.	.	2	8	.	.
.	.	4	1	9	.	.	.	5
.	.	8	.	.	7	9	.	.

上图是一个部分填充的有效的数独。

数独部分空格内已填入了数字，空白格用 ‘.’ 表示。

示例 1:

输入：

```
1 class Solution:
2     def isValidSudoku(self, board: List[List[str]]) -> bool:
3         pass
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)

[随机一题](#)

[上一题](#)

36/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



37. 解数独

难度 困难 205 收藏 分享 切换为英文

[题目描述](#)
[评论 \(131\)](#)
[题解\(27\) New](#)
[提交记录](#)
i Python3

```

1 class Solution:
2     def solveSudoku(self, board: List[List[str]]) -> None:
3         """
4             Do not return anything, modify board in-place instead.
5         """
6

```

编写一个程序，通过已填充的空格来解决数独问题。

一个数独的解法需遵循如下规则：

1. 数字 1-9 在每一行只能出现一次。
2. 数字 1-9 在每一列只能出现一次。
3. 数字 1-9 在每一个以粗实线分隔的 3x3 宫内只能出现一次。

空白格用 '.' 表示。

5	3	.	.	7
6	.	.	1	9	5	.	.	.
9	8	6	.	.
8	.	.	6	3
4	.	8	.	3	.	.	.	1
7	.	.	2	6
6	2	8	.	.
.	.	4	1	9	.	.	.	5
.	.	8	.	.	7	9	.	.

一个数独。

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)
[随机一题](#)
[上一题](#)

37/1158

[下一题](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



38. 报数

难度 简单 | 291 | [收藏](#) | [分享](#) | [切换为英文](#)

[题目描述](#)

[评论 \(983\)](#)

[题解 \(92\) New](#)

[提交记录](#)

报数序列是一个整数序列，按照其中的整数的顺序进行报数，得到下一个数。其前五项如下：

1. 1
2. 11
3. 21
4. 1211
5. 111221

1 被读作 "one 1" ("一个一")，即 11。

11 被读作 "two 1s" ("两个一")，即 21。

21 被读作 "one 2", "one 1" ("一个二", "一个一"), 即 1211。

给定一个正整数 n ($1 \leq n \leq 30$)，输出报数序列的第 n 项。

注意：整数顺序将表示为一个字符串。

示例 1:

输入： 1

输出： "1"

示例 2:

输入： 4

i Python3

```
1 class Solution:
2     def countAndSay(self, n: int) -> str:
3         :
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

38/1158

[下一题](#) >

[控制台](#)

[贡献](#)

[▶ 执行代码](#)

[提交](#)



39. 组合总和

难度 中等 351 收藏 分享 切换为英文

[题目描述](#)
[评论 \(256\)](#)
[题解\(44\) New](#)
[提交记录](#)

给定一个无重复元素的数组 `candidates` 和一个目标数 `target`，找出 `candidates` 中所有可以使数字和为 `target` 的组合。

`candidates` 中的数字可以无限制重复被选取。

说明:

- 所有数字（包括 `target`）都是正整数。
- 解集不能包含重复的组合。

示例 1:

输入: `candidates = [2,3,6,7]`, `target = 7`,
所求解集为：

```
[  
    [7],  
    [2,2,3]  
]
```

示例 2:

输入: `candidates = [2,3,5]`, `target = 8`,
所求解集为：

```
[  
    [2,2,2,2],  
    [2,3,3],  
    [3,5]
```

i Python3 ▾

```
1 class Solution:  
2     def combinationSum(self, candidates: List[int], target: int)  
3         -> List[List[int]]:  
4             .  
5             .  
6             .
```



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#) ×

[题目列表](#)
[随机一题](#)
[上一题](#)

39/1158

[下一题](#)
[控制台](#) ▾ 贡献 *i*
[▶ 执行代码](#)
[提交](#)



40. 组合总和 II

难度 中等 139 收藏 分享 切换为英文

通过次数 提交次数

20,352 36,067

[题目描述](#)
[评论 \(168\)](#)
[题解\(34\) New](#)
[提交记录](#)
i Python3


给定一个数组 candidates 和一个目标数 target，找出 candidates 中所有可以使数字和为 target 的组合。

candidates 中的每个数字在每个组合中只能使用一次。

说明:

- 所有数字（包括目标数）都是正整数。
- 解集不能包含重复的组合。

示例 1:

输入: candidates = [10,1,2,7,6,1,5], target = 8,
所求解集为：

```
[  
    [1, 7],  
    [1, 2, 5],  
    [2, 6],  
    [1, 1, 6]  
]
```

示例 2:

输入: candidates = [2,5,2,1,2], target = 5,
所求解集为：

```
[  
    [1,2,2],  
    [5]
```

```
1 class Solution:  
2     def combinationSum2(self, candidates: List[int], target: int) -> List[List[int]]:  
3         pass
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)
[随机一题](#)
[上一题](#)

40/1158

[下一题](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



41. 缺失的第一个正数

难度 困难 240 收藏 分享 切换为英文

通过次数 20,206 提交次数 55,899

[题目描述](#)
[评论 \(358\)](#)
[题解\(46\) New](#)
[提交记录](#)
i Python3


给定一个未排序的整数数组，找出其中没有出现的最小的正整数。

示例 1:

输入: [1,2,0]

输出: 3

示例 2:

输入: [3,4,-1,1]

输出: 2

示例 3:

输入: [7,8,9,11,12]

输出: 1

说明:

你的算法的时间复杂度应为 $O(n)$ ，并且只能使用常数级别的空间。

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业



```
1 class Solution:
2     def firstMissingPositive(self, nums: List[int]) -> int:
3         ...
```

[题目列表](#)
[随机一题](#)
[上一题](#)

41/1158

[下一题](#)
[控制台](#)

 贡献 *i*
[▶ 执行代码](#)
[提交](#)

42. 接雨水

难度 **困难** | 山 543 | 手收藏 | 分享 | 切换为英文

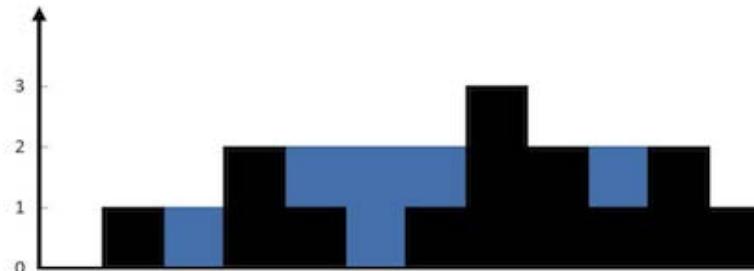
通过次数 23,870 | 提交次数 51,837

[题目描述](#)[评论 \(266\)](#)[题解\(44\) New](#)[提交记录](#)

i Python3

```
1 class Solution:
2     def trap(self, height: List[int]) -> int:
3         pass
```

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。



上面是由数组 `[0,1,0,2,1,0,1,3,2,1,2,1]` 表示的高度图，在这种情况下，可以接 6 个单位的雨水（蓝色部分表示雨水）。感谢 Marcos 贡献此图。

示例:

输入： `[0,1,0,2,1,0,1,3,2,1,2,1]`

输出： 6

在真实的面试中遇到过这道题？

[是](#)[否](#)

贡献者



相关企业



相关标签

[题目列表](#)[随机一题](#)[上一题](#)

42/1158

[下一题](#)

控制台

贡献 *i*[▶ 执行代码](#)[提交](#)



44. 通配符匹配

难度 困难 177 收藏 分享 切换为英文

[题目描述](#) [评论 \(91\)](#) [题解\(20\) New](#) [提交记录](#)

给定一个字符串 (`s`) 和一个字符模式 (`p`)，实现一个支持 `'?'` 和 `'*'` 的通配符匹配。

`'?'` 可以匹配任何单个字符。

`'*'` 可以匹配任意字符串（包括空字符串）。

两个字符串完全匹配才算匹配成功。

说明:

- `s` 可能为空，且只包含从 `a-z` 的小写字母。
- `p` 可能为空，且只包含从 `a-z` 的小写字母，以及字符 `?` 和 `*`。

示例 1:

输入：

```
s = "aa"
```

```
p = "a"
```

输出：false

解释："a" 无法匹配 "aa" 整个字符串。

示例 2:

输入：

```
s = "aa"
```

```
p = "*"
```

i Python3 ▾

```
1 class Solution:
2     def isMatch(self, s: str, p: str) -> bool:
3         .
```



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#) X

[题目列表](#)

[随机一题](#)

[上一题](#)

44/1158

[下一题](#) >

控制台 ▾

贡献 i

[▶ 执行代码](#)

[提交](#)



45. 跳跃游戏 II

难度 **困难** | 山 238 | ↗ | ❤ 收藏 | ⚡ 分享 | ⚙ 切换为英文

[题目描述](#)
[评论 \(189\)](#)
[题解\(32\) New](#)
[提交记录](#)
i Python3


给定一个非负整数数组，你最初位于数组的第一个位置。

数组中的每个元素代表你在该位置可以跳跃的最大长度。

你的目标是使用最少的跳跃次数到达数组的最后一个位置。

示例:

输入: [2,3,1,1,4]

输出: 2

解释: 跳到最后一个位置的最小跳跃数是 2。

从下标为 0 跳到下标为 1 的位置，跳 1 步，然后跳 3 步到达数组的最后一个位置。

说明:

假设你总是可以到达数组的最后一个位置。

在真实的面试中遇到过这道题?

贡献者

相关企业

相关标签

相似题目


```

1 class Solution:
2     def jump(self, nums: List[int]) -> int:
3

```

[题目列表](#)
[随机一题](#)
[上一题](#)

45/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



46. 全排列

难度 **中等** | 山 359 | 收藏 | 分享 |

通过次数 **40,584** | 提交次数 **56,863**

[题目描述](#) | [评论 \(355\)](#) | [题解\(61\) New](#) | [提交记录](#)

i Python3



```
1 class Solution:
2     def permute(self, nums: List[int]) -> List[List[int]]:
3         .
```

给定一个**没有重复数字**的序列，返回其所有可能的全排列。

示例：

输入： [1,2,3]

输出：

```
[  
    [1,2,3],  
    [1,3,2],  
    [2,1,3],  
    [2,3,1],  
    [3,1,2],  
    [3,2,1]  
]
```

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



相似题目



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

46/1158

[下一题](#) >

控制台 | 贡献 i

[▶ 执行代码](#)

[提交](#)



47. 全排列 II

难度 中等 150 收藏 分享 切换为英文

通过次数 提交次数

20,067 37,315

[题目描述](#) [评论 \(192\)](#) [题解\(35\) New](#) [提交记录](#)

给定一个可包含重复数字的序列，返回所有不重复的全排列。

示例：

输入： [1,1,2]

输出：

```
[  
    [1,1,2],  
    [1,2,1],  
    [2,1,1]  
]
```

在真实的面试中遇到过这道题？

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```
1 class Solution:  
2     def permuteUnique(self, nums: List[int]) -> List[List[int]]:  
3         ...
```



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



题目列表

随机一题

< 上一题

47/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交

48. 旋转图像

难度 中等 270 收藏 分享 切换为英文

[题目描述](#)
[评论 \(375\)](#)
[题解\(58\) New](#)
[提交记录](#)

给定一个 $n \times n$ 的二维矩阵表示一个图像。

将图像顺时针旋转 90 度。

说明:

你必须在原地旋转图像，这意味着你需要直接修改输入的二维矩阵。请不要使用另一个矩阵来旋转图像。

示例 1:

给定 `matrix =`

```
[  
    [1,2,3],  
    [4,5,6],  
    [7,8,9]  
,
```

原地旋转输入矩阵，使其变为：

```
[  
    [7,4,1],  
    [8,5,2],  
    [9,6,3]  
,
```

示例 2:

给定 `matrix =`

```
i Python3 ▾  
1 class Solution:  
2     def rotate(self, matrix: List[List[int]]) -> None:  
3         """  
4             Do not return anything, modify matrix in-place instead.  
5         """  
6
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)
[随机一题](#)
[上一题](#)
48/1158
[下一题](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



49. 字母异位词分组

难度 中等 177 收藏 分享 切换为英文

通过次数 提交次数

25,664 44,126

[题目描述](#)

[评论 \(181\)](#)

[题解\(34\) New](#)

[提交记录](#)

i Python3



```
1 class Solution:
2     def groupAnagrams(self, strs: List[str]) -> List[List[str]]:
3         .
```

给定一个字符串数组，将字母异位词组合在一起。字母异位词指字母相同，但排列不同的字符串。

示例:

输入： ["eat", "tea", "tan", "ate", "nat",
"bat"],

输出：

```
[  
    ["ate", "eat", "tea"],  
    ["nat", "tan"],  
    ["bat"]  
]
```

说明:

- 所有输入均为小写字母。
- 不考虑答案输出的顺序。

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

49/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)

50. Pow(x, n)

难度 中等 153 收藏 分享 切换为英文

[题目描述](#)

[评论 \(245\)](#)

[题解\(31\) New](#)

[提交记录](#)

实现 $\text{pow}(x, n)$ ，即计算 x 的 n 次幂函数。

示例 1:

输入: 2.00000, 10

输出: 1024.00000

示例 2:

输入: 2.10000, 3

输出: 9.26100

示例 3:

输入: 2.00000, -2

输出: 0.25000

解释: $2^{-2} = 1/2^2 = 1/4 = 0.25$

说明:

- $-100.0 < x < 100.0$
- n 是 32 位有符号整数，其数值范围是 $[-2^{31}, 2^{31} - 1]$ 。

在真实的面试中遇到过这道题?

[是](#)

[否](#)

贡献者



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

50/1158

[下一题](#)

控制台

贡献 *i*

[▶ 执行代码](#)

[提交](#)

51. N皇后

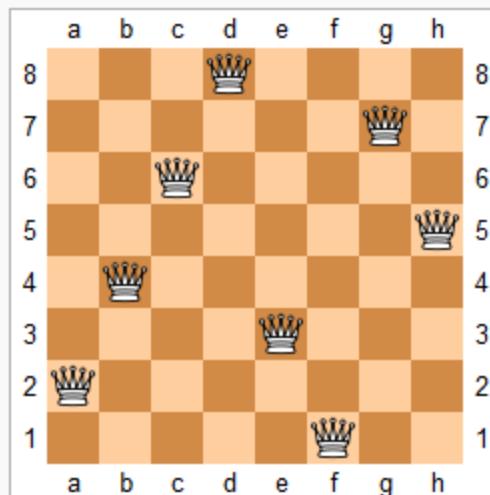
难度 困难 213 收藏 分享 切换为英文

通过次数 12,700 提交次数 19,595

[题目描述](#)[评论 \(170\)](#)[题解\(42\) ^{New}](#)[提交记录](#)

Python3

n 皇后问题研究的是如何将 *n* 个皇后放置在 $n \times n$ 的棋盘上，并且使皇后彼此之间不能相互攻击。



One solution to the eight queens puzzle

上图为 8 皇后问题的一种解法。

给定一个整数 *n*, 返回所有不同的 *n* 皇后问题的解决方案。

每一种解法包含一个明确的 *n* 皇后问题的棋子放置方案，该方案中 '*Q*' 和 '.' 分别代表了皇后和空位。

示例:

输入：4

输出：

[题目列表](#)[随机一题](#)[上一题](#)

51/1158

[下一题](#)

```

1 class Solution:
2     def solveNQueens(self, n: int) -> List[List[str]]:
3         :
        ...

```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

X

[执行代码](#)[提交](#)



53. 最大子序和

难度 简单 1150 收藏 分享 切换为英文

通过次数 86,168 提交次数 182,908

[题目描述](#)

[评论 \(554\)](#)

[题解\(93\) New](#)

[提交记录](#)

i Python3

```

1 class Solution:
2     def maxSubArray(self, nums: List[int]) -> int:
3

```

给定一个整数数组 `nums`，找到一个具有最大和的连续子数组（子数组最少包含一个元素），返回其最大和。

示例:

输入：[-2,1,-3,4,-1,2,1,-5,4],

输出：6

解释：连续子数组 [4,-1,2,1] 的和最大，为 6。

进阶:

如果你已经实现复杂度为 $O(n)$ 的解法，尝试使用更为精妙的分治法求解。

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



相似题目



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

53/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



54. 螺旋矩阵

难度 中等 197 收藏 分享 切换为英文

通过次数 提交次数

21,445 58,017

[题目描述](#)

[评论 \(292\)](#)

[题解\(63\) New](#)

[提交记录](#)

i Python3



```
1 class Solution:
2     def spiralOrder(self, matrix: List[List[int]]) -> List[int]:
```

给定一个包含 $m \times n$ 个元素的矩阵 (m 行, n 列)，请按照顺时针螺旋顺序，返回矩阵中的所有元素。

示例 1:

输入：

```
[  
 [ 1, 2, 3 ],  
 [ 4, 5, 6 ],  
 [ 7, 8, 9 ]  
]
```

输出：[1,2,3,6,9,8,7,4,5]

⋮
⋮
⋮

示例 2:

输入：

```
[  
 [1, 2, 3, 4],  
 [5, 6, 7, 8],  
 [9,10,11,12]  
]
```

输出：[1,2,3,4,8,12,11,10,9,5,6,7]

⋮

在真实的面试中遇到过这道题？

是

否

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



贡献者

[题目列表](#)

[随机一题](#)

[上一题](#)

54/1158

[下一题](#)

控制台 贡献 *i*

▶ 执行代码

提交



55. 跳跃游戏

难度 中等 278 收藏 分享 切换为英文

[题目描述](#) [评论 \(292\)](#) [题解\(41\) New](#) [提交记录](#)

给定一个非负整数数组，你最初位于数组的第一个位置。

数组中的每个元素代表你在该位置可以跳跃的最大长度。

判断你是否能够到达最后一个位置。

示例 1:

输入: [2,3,1,1,4]

输出: true

解释: 从位置 0 到 1 跳 1 步，然后跳 3 步到达最后一个位置。

示例 2:

输入: [3,2,1,0,4]

输出: false

解释: 无论怎样，你总会到达索引为 3 的位置。但该位置的最大跳跃长度是 0，所以你永远不可能到达最后一个位置。

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



[题目列表](#)

[随机一题](#)

[上一题](#)

55/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)

56. 合并区间

难度 **中等** | 山 171 | ↗ | ❤ 收藏 | ⚡ 分享 | 🇮切换为英文

通过次数 **24,786** | 提交次数 **65,099**

[题目描述](#) | [评论 \(268\)](#) | [题解\(41\) New](#) | [提交记录](#)

给出一个区间的集合，请合并所有重叠的区间。

示例 1:

输入: [[1,3],[2,6],[8,10],[15,18]]

输出: [[1,6],[8,10],[15,18]]

解释: 区间 [1,3] 和 [2,6] 重叠，将它们合并为
[1,6].

示例 2:

输入: [[1,4],[4,5]]

输出: [[1,5]]

解释: 区间 [1,4] 和 [4,5] 可被视为重叠区间。

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业 

相关标签

相似题目

题目列表

随机一题

上一题

56/1158

下一题

i Python3

```
1 class Solution:
2     def merge(self, intervals: List[List[int]]) ->
3         List[List[int]]:
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

▶ 执行代码

提交



57. 插入区间

难度 困难 57 收藏 分享 切换为英文

[题目描述](#)
[评论 \(96\)](#)
[题解\(28\) ^{New}](#)
[提交记录](#)
i Python3


给出一个无重叠的，按照区间起始端点排序的区间列表。

在列表中插入一个新的区间，你需要确保列表中的区间仍然有序且不重叠（如果有必要的话，可以合并区间）。

示例 1:

输入: intervals = [[1,3],[6,9]], newInterval = [2,5]
输出: [[1,5],[6,9]]

示例 2:

输入: intervals = [[1,2],[3,5],[6,7],[8,10],[12,16]], newInterval = [4,8]
输出: [[1,2],[3,10],[12,16]]
解释: 这是因为新的区间 [4,8] 与 [3,5],[6,7],[8,10] 重叠。

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业



相关标签


[题目列表](#)
[随机一题](#)
[上一题](#)

57/1158

[下一题 >](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)

```

1 class Solution:
2     def insert(self, intervals: List[List[int]], newInterval: List[int]) -> List[List[int]]:
3

```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



60. 第k个排列

难度 中等 110 收藏 分享 切换为英文

通过次数 12,354 提交次数 26,402

[题目描述](#)

[评论 \(188\)](#)

[题解\(30\) New](#)

[提交记录](#)

i Python3

i { } < > ⌂ ⌃ ⌄

给出集合 $[1, 2, 3, \dots, n]$ ，其所有元素共有 $n!$ 种排列。

按大小顺序列出所有排列情况，并一一标记，当 $n = 3$ 时，所有排列如下：

1. "123"
2. "132"
3. "213"
4. "231"
5. "312"
6. "321"

给定 n 和 k ，返回第 k 个排列。

说明:

- 给定 n 的范围是 $[1, 9]$ 。
- 给定 k 的范围是 $[1, n!]$ 。

示例 1:

输入: $n = 3$, $k = 3$

输出: "213"

```
1 class Solution:
2     def getPermutation(self, n: int, k: int) -> str:
3
```

示例 2:

输入: $n = 4$, $k = 9$

输出: "2314"

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

60/1158

[下一题](#)

控制台

贡献 *i*

▶ 执行代码

提交



61. 旋转链表

难度 中等 132 收藏 分享 切换为英文

通过次数 23,219 提交次数 59,545

[题目描述](#)

[评论 \(308\)](#)

[题解 \(53\) New](#)

[提交记录](#)

给定一个链表，旋转链表，将链表每个节点向右移动 k 个位置，其中 k 是非负数。

示例 1:

输入: 1->2->3->4->5->NULL, $k = 2$

输出: 4->5->1->2->3->NULL

解释:

向右旋转 1 步: 5->1->2->3->4->NULL

向右旋转 2 步: 4->5->1->2->3->NULL

示例 2:

输入: 0->1->2->NULL, $k = 4$

输出: 2->0->1->NULL

解释:

向右旋转 1 步: 2->0->1->NULL

向右旋转 2 步: 1->2->0->NULL

向右旋转 3 步: 0->1->2->NULL

向右旋转 4 步: 2->0->1->NULL

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业

[题目列表](#)

[随机一题](#)

[上一题](#)

61/1158

[下一题](#)

控制台 贡献 *i*

[执行代码](#)

[提交](#)

Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def rotateRight(self, head: ListNode, k: int) -> ListNode:
9

```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

62. 不同路径

难度 **中等** | 展示 299 | 收藏 | 分享 | 切换为英文

通过次数 34,427

提交次数 61,792

[题目描述](#)

[评论 \(351\)](#)

[题解 \(54\) New](#)

[提交记录](#)

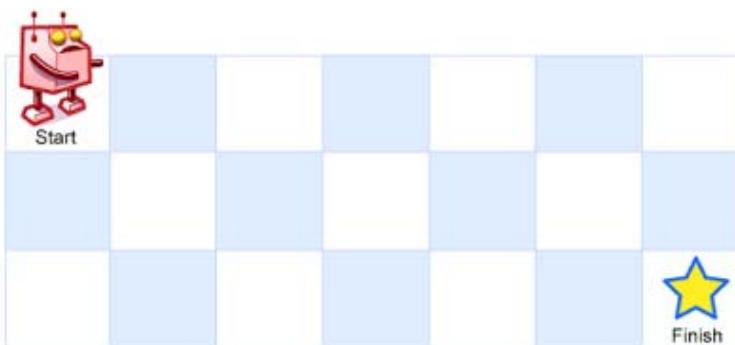
i Python3

```
1 class Solution:
2     def uniquePaths(self, m: int, n: int) -> int:
3         .
```

一个机器人位于一个 $m \times n$ 网格的左上角 (起始点在下图中标记为 "Start")。

机器人每次只能向下或者向右移动一步。机器人试图达到网格的右下角 (在下图中标记为 "Finish")。

问总共有多少条不同的路径?



例如，上图是一个 7×3 的网格。有多少可能的路径？

说明: m 和 n 的值均不超过 100。

示例 1:

输入: $m = 3$, $n = 2$

输出: 3

解释:

从左上角开始，总共有 3 条路径可以到达右下角。

1. 向右 -> 向右 -> 向下

控制台 | 贡献 i

[题目列表](#)

[随机一题](#)

[上一题](#)

62/1158

[下一题](#)

[▶ 执行代码](#)

[提交](#)

63. 不同路径 II

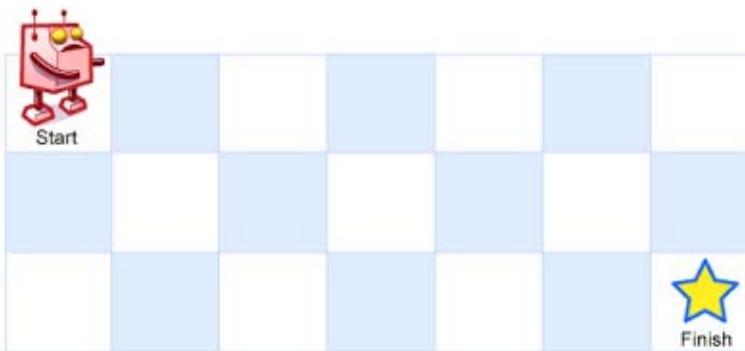
难度 中等 152 收藏 分享 切换为英文

[题目描述](#)
[评论 \(275\)](#)
[题解\(38\) ^{New}](#)
[提交记录](#)

一个机器人位于一个 $m \times n$ 网格的左上角 (起始点在下图中标记为 "Start")。

机器人每次只能向下或者向右移动一步。机器人试图达到网格的右下角 (在下图中标记为 "Finish")。

现在考虑网格中有障碍物。那么从左上角到右下角将会有多少条不同的路径?



网格中的障碍物和空位置分别用 1 和 0 来表示。

说明: m 和 n 的值均不超过 100。

示例 1:

输入:

```
[  
 [0,0,0],  
 [0,1,0],  
 [1,0,0]
```

i Python3

```
1 class Solution:  
2     def uniquePathsWithObstacles(self, obstacleGrid:  
3         List[List[int]]) -> int:  
4             ...
```

i { } < > ⌂ ⌃ ⌄

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#) X

[题目列表](#)
[随机一题](#)
[上一题](#)

63/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



64. 最小路径和

难度 中等 260 收藏 分享 切换为英文

通过次数 提交次数

26,718 42,886

[题目描述](#) [评论 \(272\)](#) [题解\(47\) New](#) [提交记录](#)

给定一个包含非负整数的 $m \times n$ 网格，请找出一条从左上角到右下角的路径，使得路径上的数字总和为最小。

说明：每次只能向下或者向右移动一步。

示例：

输入：

```
[  
    [1,3,1],  
    [1,5,1],  
    [4,2,1]  
]
```

输出：7

解释：因为路径 1→3→1→1→1 的总和最小。

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签

相似题目

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

64/1158

[下一题](#)

控制台 贡献 *i*

[执行代码](#)

[提交](#)



65. 有效数字

难度 困难 47 收藏 分享 切换为英文

通过次数 4,898 提交次数 30,817

[题目描述](#) [评论 \(142\)](#) [题解\(21\) ^{New}](#) [提交记录](#)

i Python3 ▾



验证给定的字符串是否可以解释为十进制数字。

例如:

```
"0" => true
" 0.1 " => true
"abc" => false
"1 a" => false
"2e10" => true
"-90e3" => true
" 1e" => false
"e3" => false
" 6e-1" => true
" 99e2.5 " => false
"53.5e93" => true
" --6 " => false
" -+3 " => false
"95a54e53" => false
```

```
1 class Solution:
2     def isNumber(self, s: str) -> bool:
3
```

说明: 我们有意将问题陈述地比较模糊。在实现代码之前，你应当事先思考所有可能的情况。这里给出一份可能存在于有效十进制数中的字符列表：

- 数字 0-9
- 指数 - "e"
- 正/负号 - "+" / "-"
- 小数点 - "

当然，在输入中，这些字符的上下文也很重要。

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#) X

[题目列表](#)

[随机一题](#)

[上一题](#)

65/1158

[下一题](#)

控制台

贡献 i

[执行代码](#)

[提交](#)



66. 加一

难度 简单 | 330 | 收藏 | 分享 | 切换为英文

通过次数

71,958

提交次数

177,365

[题目描述](#)
[评论 \(854\)](#)
[题解\(126\) New](#)
[提交记录](#)
i Python3


```

1 class Solution:
2     def plusOne(self, digits: List[int]) -> List[int]:
3

```

给定一个由整数组成的非空数组所表示的非负整数，在该数的基础上加一。

最高位数字存放在数组的首位， 数组中每个元素只存储单个数字。

你可以假设除了整数 0 之外，这个整数不会以零开头。

示例 1:

输入: [1,2,3]

输出: [1,2,4]

解释: 输入数组表示数字 123。

示例 2:

输入: [4,3,2,1]

输出: [4,3,2,2]

解释: 输入数组表示数字 4321。

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业



相关标签



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)


[题目列表](#)
[随机一题](#)
[上一题](#)

66/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



67. 二进制求和

难度 简单 237 收藏 分享 切换为英文

通过次数

36,352

提交次数

72,403

[题目描述](#)[评论 \(514\)](#)[题解\(73\) ^{New}](#)[提交记录](#)

Python3

给定两个二进制字符串，返回他们的和（用二进制表示）。

输入为**非空**字符串且只包含数字 1 和 0。

示例 1:

输入: a = "11", b = "1"

输出: "100"

示例 2:

输入: a = "1010", b = "1011"

输出: "10101"

在真实的面试中遇到过这道题?

[是](#)[否](#)

```
1 class Solution:
2     def addBinary(self, a: str, b: str) -> str:
3         ...
```

贡献者

相关企业

相关标签

相似题目

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)[随机一题](#)[上一题](#)

67/1158

[下一题](#)

控制台

贡献 i

[执行代码](#)[提交](#)



69. x 的平方根

难度 简单 212 收藏 分享 切换为英文

通过次数 53,827 提交次数 146,360

[题目描述](#)
[评论 \(430\)](#)
[题解\(63\) ^{New}](#)
[提交记录](#)
i Python3

```
1 class Solution:
2     def mySqrt(self, x: int) -> int:
3         pass
```

实现 `int sqrt(int x)` 函数。

计算并返回 x 的平方根，其中 x 是非负整数。

由于返回类型是整数，结果只保留整数的部分，小数部分将被舍去。

示例 1:

输入: 4

输出: 2

示例 2:

输入: 8

输出: 2

说明: 8 的平方根是 2.82842...,

由于返回类型是整数，小数部分将被舍去。

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业



相关标签

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)


[题目列表](#)
[随机一题](#)
[上一题](#)

69/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



70. 爬楼梯

难度 简单 601 收藏 分享 切换为英文

通过次数 72,854 提交次数 156,567

[题目描述](#)

[评论 \(792\)](#)

[题解\(79\) New](#)

[提交记录](#)

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？

注意：给定 n 是一个正整数。

示例 1：

输入： 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： 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

在真实的面试中遇到过这道题？

是

否

i Python3 ▾

```
1 class Solution:
2     def climbStairs(self, n: int) -> int:
3
```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

上一题

[题目列表](#)

[随机一题](#)

上一题

70/1158

下一题

控制台 贡献 i

▶ 执行代码

提交



71. 简化路径

难度 中等 68 收藏 分享 切换为英文

[题目描述](#) | [评论 \(170\)](#) | [题解\(26\) New](#)

[提交记录](#)

以 Unix 风格给出一个文件的**绝对路径**，你需要简化它。或者换句话说，将其转换为规范路径。

在 Unix 风格的文件系统中，一个点（`.`）表示当前目录本身；此外，两个点（`..`）表示将目录切换到上一级（指向父目录）；两者都可以是复杂相对路径的组成部分。更多信息请参阅：Linux / Unix 中的**绝对路径 vs 相对路径**

请注意，返回的规范路径必须始终以斜杠 / 开头，并且两个目录名之间必须只有一个斜杠 /。最后一个目录名（如果存在）**不能**以 / 结尾。此外，规范路径必须是表示绝对路径的**最短字符串**。

示例 1：

输入： `"/home/"`

输出： `"/home"`

解释： 注意，最后一个目录名后面没有斜杠。

示例 2：

输入： `"/../"`

输出： `"/"`

解释： 从根目录向上一级是不可行的，因为根是你可以到达的最高级。

示例 3：

i Python3 ▾

```
1 class Solution:
2     def simplifyPath(self, path: str) -> str:
3
```



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

71/1158

[下一题](#) >

控制台 ▾

贡献 i

[▶ 执行代码](#)

[提交](#)



72. 编辑距离

难度 **困难** | 山 296 | 手收藏 | 分享 | 切换为英文

通过次数 12,021 | 提交次数 22,339

[题目描述](#)
[评论 \(102\)](#)
[题解\(22\) New](#)
[提交记录](#)
i Python3


```
1 class Solution:
2     def minDistance(self, word1: str, word2: str) -> int:
3
```

给定两个单词 $word1$ 和 $word2$ ，计算出将 $word1$ 转换成 $word2$ 所使用的最少操作数。

你可以对一个单词进行如下三种操作：

1. 插入一个字符
2. 删除一个字符
3. 替换一个字符

示例 1:

输入: $word1 = "horse"$, $word2 = "ros"$

输出: 3

解释:

$horse \rightarrow rorse$ (将 'h' 替换为 'r')

$rorse \rightarrow rose$ (删除 'r')

$rose \rightarrow ros$ (删除 'e')

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

示例 2:

输入: $word1 = "intention"$, $word2 = "execution"$

输出: 5

解释:

$intention \rightarrow inention$ (删除 't')

$inention \rightarrow enention$ (将 'i' 替换为 'e')

$enention \rightarrow exention$ (将 'n' 替换为 'x')

$exention \rightarrow exection$ (将 'n' 替换为 'c')

$exection \rightarrow execution$ (插入 'u')

[题目列表](#)
[随机一题](#)
[上一题](#)

72/1158

[下一题 >](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)

75. 颜色分类

难度 **中等** | 山 222 | ⚡ | ❤ 收藏 | ⚡ 分享 | ⚡ 切换为英文

[题目描述](#) | [评论 \(274\)](#) | [题解\(34\) New](#) | [提交记录](#)

给定一个包含红色、白色和蓝色，一共 n 个元素的数组，**原地**对它们进行排序，使得相同颜色的元素相邻，并按照红色、白色、蓝色顺序排列。

此题中，我们使用整数 0、1 和 2 分别表示红色、白色和蓝色。

注意:

不能使用代码库中的排序函数来解决这道题。

示例:

输入： [2,0,2,1,1,0]

输出： [0,0,1,1,2,2]

进阶:

- 一个直观的解决方案是使用计数排序的两趟扫描算法。
首先，迭代计算出0、1 和 2 元素的个数，然后按照0、1、2 的排序，重写当前数组。
- 你能想出一个仅使用常数空间的一趟扫描算法吗？

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



[题目列表](#)

[随机一题](#)

[上一题](#)

75/1158

[下一题](#)

i Python3

```

1 class Solution:
2     def sortColors(self, nums: List[int]) -> None:
3         """
4             Do not return anything, modify nums in-place instead.
5         """
6

```

您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[▶ 执行代码](#)

[提交](#)



76. 最小覆盖子串

难度 困难 204 收藏 分享 切换为英文

通过次数 提交次数

10,151 28,157

[题目描述](#)

[评论 \(73\)](#)

[题解\(17\) New](#)

[提交记录](#)

i Python3

i { } < > ⌂ ⌃ ⌄

```
1 class Solution:
2     def minWindow(self, s: str, t: str) -> str:
3         :
```

给你一个字符串 S、一个字符串 T，请在字符串 S 里面找出：包含 T 所有字母的最小子串。

示例：

输入：S = "ADOBECODEBANC", T = "ABC"

输出："BANC"

说明：

- 如果 S 中不存在这样的子串，则返回空字符串 ""。
- 如果 S 中存在这样的子串，我们保证它是唯一的答案。

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



相似题目



显示提示1



显示提示2



您上次编辑到这里，代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)



[题目列表](#)

[随机一题](#)

[上一题](#)

76/1158

[下一题](#)

控制台 贡献 i

▶ 执行代码

提交



78. 子集

难度 中等 315 收藏 分享 切换为英文

[题目描述](#)
[评论 \(339\)](#)
[题解\(43\) New](#)
[提交记录](#)
i Python3

```

1 class Solution:
2     def subsets(self, nums: List[int]) -> List[List[int]]:
3

```

给定一组不含重复元素的整数数组 $nums$, 返回该数组所有可能的子集 (幂集)。

说明: 解集不能包含重复的子集。

示例:

输入: $nums = [1,2,3]$

输出:

```
[
    [3],
    [1],
    [2],
    [1,2,3],
    [1,3],
    [2,3],
    [1,2],
    []
]
```

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业

您上次编辑到这里, 代码已从您浏览器本地的临时存储中恢复了 [还原默认代码模版](#)

[题目列表](#)
[随机一题](#)
[上一题](#)

78/1158

[下一题 >](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



79. 单词搜索

难度 中等 193 收藏 分享 切换为英文

通过次数 提交次数

17,526 45,173

[题目描述](#) [评论 \(160\)](#) [题解\(31\) New](#) [提交记录](#)

i Python3



```
1 class Solution:
2     def exist(self, board: List[List[str]], word: str) -> bool:
3         pass
```

给定一个二维网格和一个单词，找出该单词是否存在与网格中。

单词必须按照字母顺序，通过相邻的单元格内的字母构成，其中“相邻”单元格是那些水平相邻或垂直相邻的单元格。同一个单元格内的字母不允许被重复使用。

示例：

```
board =
[
    ['A', 'B', 'C', 'E'],
    ['S', 'F', 'C', 'S'],
    ['A', 'D', 'E', 'E']
]
```

给定 word = "ABCED"，返回 true。
 给定 word = "SEE"，返回 true。
 给定 word = "ABCB"，返回 false。

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



[题目列表](#)

[随机一题](#)

[上一题](#)

79/1158

[下一题](#)

控制台

贡献 i

[执行代码](#)

[提交](#)



81. 搜索旋转排序数组 II

难度 中等 58 收藏 分享 切换为英文

通过次数

9,327

提交次数

27,473

[题目描述](#)[评论 \(99\)](#)[题解\(16\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def search(self, nums: List[int], target: int) -> bool:
3

```

假设按照升序排序的数组在预先未知的某个点上进行了旋转。

(例如，数组 `[0, 0, 1, 2, 2, 5, 6]` 可能变为 `[2, 5, 6, 0, 0, 1, 2]`)。

编写一个函数来判断给定的目标值是否存在于数组中。若存在返回 `true`，否则返回 `false`。

示例 1:

输入: `nums = [2,5,6,0,0,1,2]`, `target = 0`

输出: `true`

示例 2:

输入: `nums = [2,5,6,0,0,1,2]`, `target = 3`

输出: `false`

进阶:

- 这是 搜索旋转排序数组 的延伸题目，本题中的 `nums` 可能包含重复元素。
- 这会影响到程序的时间复杂度吗？会有怎样的影响，为什么？

在真实的面试中遇到过这道题？

[是](#)

[否](#)

贡献者

[题目列表](#)

[随机一题](#)

[上一题](#)

81/1158

[下一题](#)

控制台

贡献 *i*

[执行代码](#)

[提交](#)



82. 删除排序链表中的重复元素 II

难度 中等 132 收藏 分享 切换为英文

通过次数 提交次数

16,378 37,803

[题目描述](#)

[评论 \(197\)](#)

[题解\(41\) New](#)

[提交记录](#)

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def deleteDuplicates(self, head: ListNode) -> ListNode:
9

```

给定一个排序链表，删除所有含有重复数字的节点，只保留原始链表中 没有重复出现的数字。

示例 1:

输入：1->2->3->3->4->4->5

输出：1->2->5

示例 2:

输入：1->1->1->2->3

输出：2->3

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



相关标签



相似题目



[题目列表](#)

[随机一题](#)

< 上一题

82/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



83. 删除排序链表中的重复元素

难度 简单 189 收藏 分享 切换为英文

通过次数 40,889 提交次数 86,170

[题目描述](#) [评论 \(340\)](#) [题解\(43\) New](#) [提交记录](#)

给定一个排序链表，删除所有重复的元素，使得每个元素只出现一次。

示例 1:

输入: 1->1->2

输出: 1->2

示例 2:

输入: 1->1->2->3->3

输出: 1->2->3

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def deleteDuplicates(self, head: ListNode) -> ListNode:
9

```

[题目列表](#)

[随机一题](#)

[上一题](#)

83/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



85. 最大矩形

难度 困难 181 收藏 分享 切换为英文

通过次数

7,398

提交次数

17,108

[题目描述](#)[评论 \(41\)](#)[题解\(13\) ^{New}](#)[提交记录](#)

Python3

给定一个仅包含 0 和 1 的二维二进制矩阵，找出只包含 1 的最大矩形，并返回其面积。

示例:

输入：

```
[["1","0","1","0","0"],  
 ["1","0","1","1","1"],  
 ["1","1","1","1","1"],  
 ["1","0","0","1","0"]]
```

]

输出：6

在真实的面试中遇到过这道题？

[是](#)[否](#)

贡献者

相关企业

相关标签

相似题目

```
1 class Solution:  
2     def maximalRectangle(self, matrix: List[List[str]]) -> int:  
3         pass
```

[题目列表](#)[随机一题](#)[上一题](#)

85/1158

[下一题](#)

控制台

贡献 i

[执行代码](#)[提交](#)



88. 合并两个有序数组

难度 简单 300 收藏 分享 切换为英文

通过次数 64,194 提交次数 142,309

[题目描述](#)
[评论 \(699\)](#)
[题解 \(81\) New](#)
[提交记录](#)

给定两个有序整数数组 $nums1$ 和 $nums2$ ，将 $nums2$ 合并到 $nums1$ 中，使得 $nums1$ 成为一个有序数组。

说明:

- 初始化 $nums1$ 和 $nums2$ 的元素数量分别为 m 和 n 。
- 你可以假设 $nums1$ 有足够的空间（空间大小大于或等于 $m + n$ ）来保存 $nums2$ 中的元素。

示例:

输入：

```
nums1 = [1,2,3,0,0,0], m = 3
nums2 = [2,5,6], n = 3
```

输出：[1,2,2,3,5,6]

在真实的面试中遇到过这道题？

贡献者



相关企业



相关标签



相似题目


i Python3

```

1 class Solution:
2     def merge(self, nums1: List[int], m: int, nums2: List[int],
3             n: int) -> None:
4         """
5             Do not return anything, modify nums1 in-place instead.
6         """

```

[题目列表](#)
[随机一题](#)
[上一题](#)

88/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



91. 解码方法

难度 中等 179 收藏 分享 切换为英文

通过次数 提交次数

13,729 63,376

题目描述

评论 (185)

题解(35) New

提交记录

i Python3 ▾

i { } < > ⌂ ⌃ ⌁

一条包含字母 A-Z 的消息通过以下方式进行了编码：

```
'A' -> 1  
'B' -> 2  
...  
'Z' -> 26
```

给定一个只包含数字的非空字符串，请计算解码方法的总数。

示例 1:

输入： "12"

输出： 2

解释： 它可以解码为 "AB" (1 2) 或者 "L" (12)。

示例 2:

输入： "226"

输出： 3

解释： 它可以解码为 "BZ" (2 26), "VF" (22 6), 或者 "BBF" (2 2 6)。

在真实的面试中遇到过这道题？

是

否

贡献者



控制台 ▾ 贡献 i

三 题目列表

随机一题

< 上一题

91/1158

下一题 >

▶ 执行代码

提交



92. 反转链表 II

难度 中等 188 收藏 分享 切换为英文

通过次数 提交次数

17,738 38,276

[题目描述](#) [评论 \(206\)](#) [题解\(52\) New](#) [提交记录](#)

反转从位置 m 到 n 的链表。请使用一趟扫描完成反转。

说明:

$1 \leq m \leq n \leq$ 链表长度。

示例:

输入: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow \text{NULL}$, $m = 2$, $n = 4$

输出: $1 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 5 \rightarrow \text{NULL}$

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def reverseBetween(self, head: ListNode, m: int, n: int) ->
9     ListNode:

```

题目列表

随机一题

< 上一题

92/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



93. 复原IP地址

难度 中等 123 收藏 分享 切换为英文

通过次数

12,734

提交次数

28,236

[题目描述](#)[评论 \(161\)](#)[题解\(23\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def restoreIpAddresses(self, s: str) -> List[str]:
3

```

给定一个只包含数字的字符串，复原它并返回所有可能的 IP 地址格式。

示例:

输入： "25525511135"

输出： ["255.255.11.135", "255.255.111.35"]

在真实的面试中遇到过这道题？

[是](#)[否](#)

贡献者



相关企业



相关标签



相似题目

[题目列表](#)[随机一题](#)[上一题](#)

93/1158

[下一题](#)

控制台

贡献

[▶ 执行代码](#)[提交](#)



97. 交错字符串

难度 困难 85 收藏 分享 切换为英文

[题目描述](#) [评论 \(63\)](#) [题解\(11\) New](#) [提交记录](#)

给定三个字符串 s_1, s_2, s_3 , 验证 s_3 是否是由 s_1 和 s_2 交错组成的。

示例 1:

输入: $s_1 = "aabcc"$, $s_2 = "dbbca"$, $s_3 = "adbcbcac"$

输出: true

示例 2:

输入: $s_1 = "aabcc"$, $s_2 = "dbbca"$, $s_3 = "aadbbbaccc"$

输出: false

在真实的面试中遇到过这道题?

是

否

贡献者

相关企业

相关标签

i Python3 ▾

```
1 class Solution:
2     def isInterleave(self, s1: str, s2: str, s3: str) -> bool:
3         :
```



[题目列表](#)

[随机一题](#)

[上一题](#)

97/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



98. 验证二叉搜索树

难度 中等 264 收藏 分享 切换为英文

通过次数 35,768 提交次数 130,480

[题目描述](#) [评论 \(316\)](#) [题解\(44\) New](#) [提交记录](#)

i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def isValidBST(self, root: TreeNode) -> bool:
10

```

给定一个二叉树，判断其是否是一个有效的二叉搜索树。

假设一个二叉搜索树具有如下特征：

- 节点的左子树只包含**小于**当前节点的数。
- 节点的右子树只包含**大于**当前节点的数。
- 所有左子树和右子树自身必须也是二叉搜索树。

示例 1:

输入：

```

2
/
1   3

```

输出：true

示例 2:

输入：

```

5
/
1   4
  /
 3   6

```

输出：false

解释：输入为：[5,1,4,null,null,3,6]。

根节点的值为 5，但是其右子节点值为 4。

[题目列表](#)

[随机一题](#)

[上一题](#)

98/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)

101. 对称二叉树

难度 简单  422   收藏  分享  切换为英文

通过次数 提交次数
50,536 104,887

题目描述 评论 (319) 题解(46) New 提交记录

给定一个二叉树，检查它是否是镜像对称的。

例如，二叉树 [1, 2, 2, 3, 4, 4, 3] 是对称的。

```

graph TD
    1 --- 2
    1 --- 3
    2 --- 3L[3]
    2 --- 4L[4]
    3 --- 4M[4]
    3 --- 3R[3]
  
```

但是下面这个 [1, 2, 2, null, 3, null, 3] 则不是镜像对称的。

$$\begin{array}{ccccc} & & 1 & & \\ & / & \backslash & & \\ 2 & & 2 & & \\ & \backslash & & & \backslash \\ & 3 & & & 3 \end{array}$$

说明:

如果你可以运用递归和迭代两种方法解决这个问题，会很加分。

在真实的面试中遇到过这道题？

是 否

贡献者

6

三 题目列表

随机一题

< 上一題

101/1158

下一题 >

控制台

贡献

▶ 执行代码

提交



102. 二叉树的层次遍历

难度 中等 256 收藏 分享 切换为英文

通过次数 40,692 提交次数 69,739

[题目描述](#)
[评论 \(270\)](#)
[题解\(55\) ^{New}](#)
[提交记录](#)
i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def levelOrder(self, root: TreeNode) -> List[List[int]]:
10

```

给定一个二叉树，返回其按层次遍历的节点值。 (即逐层地，从左到右访问所有节点)。

例如：

给定二叉树: [3, 9, 20, null, null, 15, 7]，

```

3
/
9  20
/
15  7

```

返回其层次遍历结果：

```
[
    [3],
    [9,20],
    [15,7]
]
```

在真实的面试中遇到过这道题？

贡献者



相关企业



102/1158

控制台

贡献 i



106. 从中序与后序遍历序列构造二叉树

难度 中等 106 收藏 分享 切换为英文

通过次数 12,500 提交次数 19,566

[题目描述](#)
[评论 \(87\)](#)
[题解\(25\) ^{New}](#)
[提交记录](#)
i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def buildTree(self, inorder: List[int], postorder: List[int]) -> TreeNode:
10

```

根据一棵树的中序遍历与后序遍历构造二叉树。

注意:

你可以假设树中没有重复的元素。

例如，给出

中序遍历 `inorder = [9,3,15,20,7]`

后序遍历 `postorder = [9,15,7,20,3]`

返回如下的二叉树：

```

3
/ \
9  20
 / \
15   7

```

在真实的面试中遇到过这道题？

[是](#)
[否](#)

贡献者



相关企业



相关标签


[题目列表](#)
[随机一题](#)
[上一题](#)

106/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



107. 二叉树的层次遍历 II

难度 简单 129 收藏 分享 切换为英文

通过次数 23,147 提交次数 37,204

题目描述

评论 (256)

题解(39) ^{New}

提交记录

Python3

给定一个二叉树，返回其节点值自底向上的层次遍历。 (即按从叶子节点所在层到根节点所在的层，逐层从左向右遍历)

例如：

给定二叉树 [3, 9, 20, null, null, 15, 7]，



返回其自底向上的层次遍历为：

```
[
  [15,7],
  [9,20],
  [3]
]
```

在真实的面试中遇到过这道题？

是

否

贡献者



相关企业



```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def levelOrderBottom(self, root: TreeNode) ->
List[List[int]]:
10

```

题目列表

随机一题

< 上一题

107/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



110. 平衡二叉树

难度 简单 148 收藏 分享 切换为英文

通过次数 26,929 提交次数 54,866

[题目描述](#)

[评论 \(303\)](#)

[题解 \(42\) New](#)

[提交记录](#)

i Python3 ▾

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def isBalanced(self, root: TreeNode) -> bool:
10

```

给定一个二叉树，判断它是否是高度平衡的二叉树。

本题中，一棵高度平衡二叉树定义为：

一个二叉树每个节点的左右两个子树的高度差的绝对值不超过 1。

示例 1:

给定二叉树 [3, 9, 20, null, null, 15, 7]

```

3
/
9  20
/ \
15   7

```

返回 true。

示例 2:

给定二叉树 [1, 2, 2, 3, 3, null, null, 4, 4]

```

1
/
2  2
/ \
3   3

```

[题目列表](#)

[随机一题](#)

[上一题](#)

110/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



111. 二叉树的最小深度

难度 简单 151 收藏 分享 切换为英文

通过次数 27,584 提交次数 69,169

[题目描述](#)

[评论 \(307\)](#)

[题解\(34\) New](#)

[提交记录](#)

i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def minDepth(self, root: TreeNode) -> int:
10

```

给定一个二叉树，找出其最小深度。

最小深度是从根节点到最近叶子节点的最短路径上的节点数量。

说明: 叶子节点是指没有子节点的节点。

示例:

给定二叉树 [3, 9, 20, null, null, 15, 7]，

```

3
/
9  20
/
15  7

```

返回它的最小深度 2。

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



相似题目



[题目列表](#)

[随机一题](#)

< 上一题

111/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



113. 路径总和 II

难度 中等 117 收藏 分享 切换为英文

通过次数 14,583 提交次数 25,584

[题目描述](#)

[评论 \(177\)](#)

[题解\(25\) New](#)

[提交记录](#)

i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def pathSum(self, root: TreeNode, sum: int) ->
10    List[List[int]]:

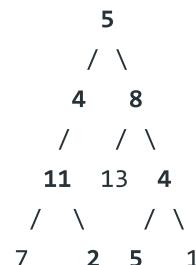
```

给定一个二叉树和一个目标和，找到所有从根节点到叶子节点路径总和等于给定目标和的路径。

说明: 叶子节点是指没有子节点的节点。

示例:

给定如下二叉树，以及目标和 $sum = 22$ ，



返回:

```
[
  [5,4,11,2],
  [5,8,4,5]
]
```

在真实的面试中遇到过这道题?

是

否

贡献者



[题目列表](#)

[随机一题](#)

[上一题](#)

113/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交



114. 二叉树展开为链表

难度 中等 165 收藏 分享 切换为英文

通过次数 12,325 提交次数 19,123

[题目描述](#)

[评论 \(163\)](#)

[题解\(27\) New](#)

[提交记录](#)

i Python3

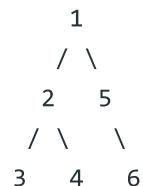
```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def flatten(self, root: TreeNode) -> None:
10         """
11             Do not return anything, modify root in-place instead.
12         """
13

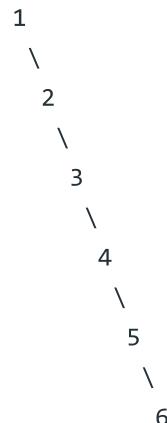
```

给定一个二叉树，原地将它展开为链表。

例如，给定二叉树



将其展开为：



在真实的面试中遇到过这道题？

是

否

[题目列表](#)

[随机一题](#)

[上一题](#)

114/1158

[下一题](#)

控制台

贡献 i

▶ 执行代码

提交

118. 杨辉三角

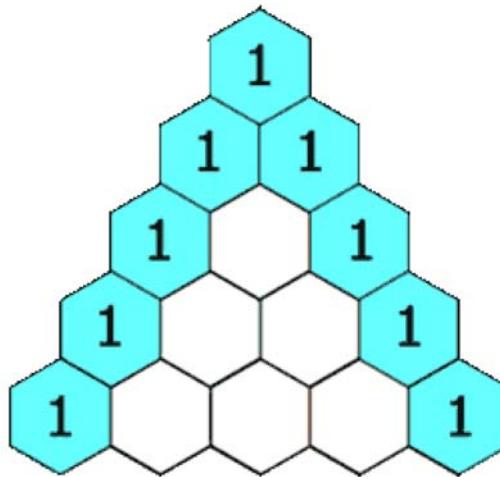
难度 简单 182 收藏 分享 切换为英文

通过次数 34,314 提交次数 53,866

[题目描述](#)
[评论 \(377\)](#)
[题解\(49\) ^{New}](#)
[提交记录](#)
i Python3

```
1 class Solution:
2     def generate(self, numRows: int) -> List[List[int]]:
3         ...
```

给定一个非负整数 $\textit{numRows}$, 生成杨辉三角的前 $\textit{numRows}$ 行。



在杨辉三角中，每个数是它左上方和右上方的数的和。

示例:

输入： 5

输出：

```
[  
    [1],  
    [1,1],  
    [1,2,1],  
    [1,3,3,1],  
    [1,4,6,4,1]
```

[题目列表](#)
[随机一题](#)
[上一题](#)

118/1158

[下一题 >](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



120. 三角形最小路径和

难度 中等 201 收藏 分享 切换为英文

通过次数 19,587 提交次数 31,920

题目描述

评论 (232)

题解(39) New

提交记录

i Python3 ▾

i { } < > ⌂ ⌃ ⌄

给定一个三角形，找出自顶向下的最小路径和。每一步只能移动到下一行中相邻的结点上。

例如，给定三角形：

```
[  
    [2],  
    [3,4],  
    [6,5,7],  
    [4,1,8,3]  
]
```

自顶向下的最小路径和为 11 (即, $2 + 3 + 5 + 1 = 11$)。

说明:

如果你可以只使用 $O(n)$ 的额外空间 (n 为三角形的总行数) 来解决这个问题，那么你的算法会很加分。

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



```
1 class Solution:  
2     def minimumTotal(self, triangle: List[List[int]]) -> int:  
3         pass
```

题目列表

随机一题

< 上一题

120/1158

下一题 >

控制台 ▾

贡献 i

▶ 执行代码

提交



121. 买卖股票的最佳时机

难度 简单 515 收藏 分享 切换为英文

通过次数

69,604

提交次数

137,708

[题目描述](#)[评论 \(367\)](#)[题解\(51\) New](#)[提交记录](#)

Python3

```

1 class Solution:
2     def maxProfit(self, prices: List[int]) -> int:
3

```

给定一个数组，它的第 i 个元素是一支给定股票第 i 天的价格。

如果你最多只允许完成一笔交易（即买入和卖出一支股票），设计一个算法来计算你所能获取的最大利润。

注意你不能在买入股票前卖出股票。

示例 1:

输入: [7,1,5,3,6,4]

输出: 5

解释: 在第 2 天 (股票价格 = 1) 的时候买入，在第 5 天 (股票价格 = 6) 的时候卖出，最大利润 = $6-1 = 5$

。

注意利润不能是 $7-1 = 6$ ，因为卖出价格需要大于买入价格。

示例 2:

输入: [7,6,4,3,1]

输出: 0

解释: 在这种情况下，没有交易完成，所以最大利润为 0。

在真实的面试中遇到过这道题?

[是](#)

[否](#)

二三五

[题目列表](#)

[随机一题](#)

[上一题](#)

121/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



122. 买卖股票的最佳时机 II

难度 简单 | 453 | [收藏](#) | [分享](#) | [切换为英文](#)

通过次数

72,865

提交次数

132,178

[题目描述](#)

[评论 \(425\)](#)

[题解\(48\) New](#)

[提交记录](#)

i Python3



```
1 class Solution:
2     def maxProfit(self, prices: List[int]) -> int:
3
```

给定一个数组，它的第 i 个元素是一支给定股票第 i 天的价格。

设计一个算法来计算你所能获取的最大利润。你可以尽可能地完成更多的交易（多次买卖一支股票）。

注意：你不能同时参与多笔交易（你必须在再次购买前出售掉之前的股票）。

示例 1:

输入: [7,1,5,3,6,4]

输出: 7

解释: 在第 2 天 (股票价格 = 1) 的时候买入，在第 3 天 (股票价格 = 5) 的时候卖出，这笔交易所能获得利润 = $5-1 = 4$ 。

随后，在第 4 天 (股票价格 = 3) 的时候买入，在第 5 天 (股票价格 = 6) 的时候卖出，这笔交易所能获得利润 = $6-3 = 3$ 。

示例 2:

输入: [1,2,3,4,5]

输出: 4

解释: 在第 1 天 (股票价格 = 1) 的时候买入，在第 5 天 (股票价格 = 5) 的时候卖出，这笔交易所能获得利润 = $5-1 = 4$ 。

注意你不能在第 1 天和第 2 天接连购买股票，之后再将它们卖出

[题目列表](#)

[随机一题](#)

[上一题](#)

122/1158

[下一题](#)

控制台

贡献 i

[▶ 执行代码](#)

[提交](#)



123. 买卖股票的最佳时机 III

难度 困难 178 收藏 分享 切换为英文

通过次数 9,978 提交次数 25,494

[题目描述](#)

[评论 \(86\)](#)

[题解\(20\) New](#)

[提交记录](#)

i Python3 ▾

i { } < > ⌂ ⌃ ⌁

```
1 class Solution:
2     def maxProfit(self, prices: List[int]) -> int:
3
```

给定一个数组，它的第 i 个元素是一支给定的股票在第 i 天的价格。

设计一个算法来计算你所能获取的最大利润。你最多可以完成 **两笔交易**。

注意: 你不能同时参与多笔交易（你必须在再次购买前出售掉之前的股票）。

示例 1:

输入: [3,3,5,0,0,3,1,4]

输出: 6

解释: 在第 4 天 (股票价格 = 0) 的时候买入，在第 6 天 (股票价格 = 3) 的时候卖出，这笔交易所能获得利润 = $3-0 = 3$ 。

随后，在第 7 天 (股票价格 = 1) 的时候买入，在第 8 天 (股票价格 = 4) 的时候卖出，这笔交易所能获得利润 = $4-1 = 3$ 。

示例 2:

输入: [1,2,3,4,5]

输出: 4

解释: 在第 1 天 (股票价格 = 1) 的时候买入，在第 5 天 (股票价格 = 5) 的时候卖出，这笔交易所能获得利润 = $5-1 = 4$ 。

注意你不能在第 1 天和第 2 天接连购买股票。→

[题目列表](#)

[随机一题](#)

< 上一题

123/1158

下一题 >

控制台 ▾

贡献 i

▶ 执行代码

提交



124. 二叉树中的最大路径和

难度 困难 179 收藏 分享 切换为英文

通过次数 12,375 提交次数 33,104

[题目描述](#)
[评论 \(120\)](#)
[题解\(25\) ^{New}](#)
[提交记录](#)
i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def maxPathSum(self, root: TreeNode) -> int:
10

```

给定一个**非空**二叉树，返回其最大路径和。

本题中，路径被定义为一条从树中任意节点出发，达到任意节点的序列。该路径**至少包含一个节点**，且不一定经过根节点。

示例 1:

输入: [1,2,3]

```

1
/
2  3

```

输出: 6

示例 2:

输入: [-10,9,20,null,null,15,7]

```

-10
/
9  20
  / \
 15   7

```

输出: 42

[题目列表](#)
[随机一题](#)
[上一题](#)

124/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



125. 验证回文串

难度 简单 | 山 104 | ⚡ | ❤ 收藏 | 📁 分享 | ⚙ 切换为英文

通过次数 49,236 | 提交次数 120,650

题目描述

评论 (370)

题解(54) New

提交记录

i Python3 ▾

i { } ⌂ > ⌂ ⌂

给定一个字符串，验证它是否是回文串，只考虑字母和数字字符，可以忽略字母的大小写。

说明：本题中，我们将空字符串定义为有效的回文串。

示例 1:

输入: "A man, a plan, a canal: Panama"

输出: true

```
1 class Solution:
2     def isPalindrome(self, s: str) -> bool:
3         .
```

示例 2:

输入: "race a car"

输出: false

⋮

在真实的面试中遇到过这道题?

是

否

⋮

贡献者



⋮

相关企业



⋮

相关标签



⋮

相似题目



⋮

三 题目列表

随机一题

上一题

125/1158

下一题 >

控制台 ▾

贡献 i

▶ 执行代码

提交



126. 单词接龙 II

难度 困难 | 66 | 收藏 | 分享 | 切换为英文

通过次数 2,586 提交次数 9,322

[题目描述](#)
[评论 \(48\)](#)
[题解\(12\) New](#)
[提交记录](#)
i Python3 ▾


给定两个单词 (`beginWord` 和 `endWord`) 和一个字典 `wordList`, 找出所有从 `beginWord` 到 `endWord` 的最短转换序列。转换需遵循如下规则:

1. 每次转换只能改变一个字母。
2. 转换过程中的中间单词必须是字典中的单词。

说明:

- 如果不存在这样的转换序列, 返回一个空列表。
- 所有单词具有相同的长度。
- 所有单词只由小写字母组成。
- 字典中不存在重复的单词。
- 你可以假设 `beginWord` 和 `endWord` 是非空的, 且二者不相同。

示例 1:

输入:

```
beginWord = "hit",
endWord = "cog",
wordList = ["hot", "dot", "dog", "lot", "log", "cog"]
```

输出:

```
[["hit", "hot", "dot", "dog", "cog"],
 ["hit", "hot", "lot", "log", "cog"]]
```

```
1 class Solution:
2     def findLadders(self, beginWord: str, endWord: str,
3 wordList: List[str]) -> List[List[str]]:
```

[题目列表](#)
[随机一题](#)
[上一题](#)

126/1158

[下一题](#)
[控制台](#) ▾

[贡献](#) i

[▶ 执行代码](#)
[提交](#)



128. 最长连续序列

难度 困难 154 收藏 分享 切换为英文

通过次数 13,330 提交次数 28,630

[题目描述](#)
[评论 \(121\)](#)
[题解\(21\) New](#)
[提交记录](#)
i Python3 ▾


给定一个未排序的整数数组，找出最长连续序列的长度。

要求算法的时间复杂度为 $O(n)$ 。

示例:

输入: [100, 4, 200, 1, 3, 2]

输出: 4

解释: 最长连续序列是 [1, 2, 3, 4]。它的长度为 4。

在真实的面试中遇到过这道题?

[是](#)
[否](#)

贡献者



相关企业



相关标签



相似题目



```

1 class Solution:
2     def longestConsecutive(self, nums: List[int]) -> int:
3

```

[题目列表](#)
[随机一题](#)
[上一题](#)

128/1158

[下一题](#)
[控制台](#) ▾

[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



130. 被围绕的区域

难度 中等 92 收藏 分享 切换为英文

通过次数 提交次数

8,907 23,614

[题目描述](#)

[评论 \(85\)](#)

[题解\(21\) New](#)

[提交记录](#)

i Python3



```

1 class Solution:
2     def solve(self, board: List[List[str]]) -> None:
3         """
4             Do not return anything, modify board in-place instead.
5         """
6

```

给定一个二维的矩阵，包含 'X' 和 'O' (字母 O)。

找到所有被 'X' 围绕的区域，并将这些区域里所有的 'O' 用 'X' 填充。

示例:

```

X X X X
X O O X
X X O X
X O X X

```

运行你的函数后，矩阵变为：

```

X X X X
X X X X
X X X X
X O X X

```

解释:

被围绕的区间不会存在于边界上，换句话说，任何边界上的 'O' 都不会被填充为 'X'。任何不在边界上，或不与边界上的 'O' 相连的 'O' 最终都会被填充为 'X'。如果两个元素在水平或垂直方向相邻，则称它们是“相连”的。

在真实的面试中遇到过这道题？

是

否

[题目列表](#)

[随机一题](#)

< 上一题

130/1158

下一题 >

控制台 ▾

贡献 i

▶ 执行代码

提交



131. 分割回文串

难度 中等 150 收藏 分享 切换为英文

通过次数 提交次数

11,958 18,839

[题目描述](#)

[评论 \(122\)](#)

[题解\(18\) New](#)

[提交记录](#)

i Python3



给定一个字符串 s , 将 s 分割成一些子串, 使每个子串都是回文串。

返回 s 所有可能的分割方案。

示例:

输入: "aab"

输出:

```
[["aa", "b"], ["a", "a", "b"]]
```

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签



相似题目



```
1 class Solution:
2     def partition(self, s: str) -> List[List[str]]:
```

[题目列表](#)

[随机一题](#)

< 上一题

131/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



132. 分割回文串 II

难度 困难 69 收藏 分享 切换为英文

通过次数

3,780

提交次数

9,417

题目描述

评论 (44)

题解(14) ^{New}

提交记录

给定一个字符串 s , 将 s 分割成一些子串, 使每个子串都是回文串。

返回符合要求的最少分割次数。

示例:

输入: "aab"

输出: 1

解释: 进行一次分割就可将 s 分割成 ["aa", "b"] 这样两个回文子串。

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签

相似题目

Python3

```
1 class Solution:
2     def minCut(self, s: str) -> int:
3
```



题目列表

随机一题

< 上一题

132/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



135. 分发糖果

难度 困难 93 收藏 分享 切换为英文

[题目描述](#)

[评论 \(42\)](#)

[题解\(15\) New](#)

[提交记录](#)

i Python3 ▾

i { } < > ⌂ ⌃ ⌁

老师想给孩子们分发糖果，有 N 个孩子站成了一条直线，老师会根据每个孩子的表现，预先给他们评分。

你需要按照以下要求，帮助老师给这些孩子分发糖果：

- 每个孩子至少分配到 1 个糖果。
- 相邻的孩子中，评分高的孩子必须获得更多的糖果。

那么这样下来，老师至少需要准备多少颗糖果呢？

示例 1:

输入： [1,0,2]

输出： 5

解释：你可以分别给这三个孩子分发 2、1、2 颗糖果。

```
1 class Solution:
2     def candy(self, ratings: List[int]) -> int:
3
```

示例 2:

输入： [1,2,2]

输出： 4

解释：你可以分别给这三个孩子分发 1、2、1 颗糖果。

第三个孩子只得到 1 颗糖果，这已满足上述两个条件。

在真实的面试中遇到过这道题？

是

否

贡献者



[题目列表](#)

[随机一题](#)

[上一题](#)

135/1158

[下一题](#) >

控制台 ▾

贡献 i

▶ 执行代码

提交



136. 只出现一次的数字

难度 简单 811 收藏 分享 切换为英文

通过次数 94,232 提交次数 149,385

[题目描述](#) [评论 \(402\)](#) [题解\(82\) New](#) [提交记录](#)

给定一个非空整数数组，除了某个元素只出现一次以外，其余每个元素均出现两次。找出那个只出现了一次的元素。

说明：

你的算法应该具有线性时间复杂度。你可以不使用额外空间来实现吗？

示例 1：

输入： [2,2,1]

输出： 1

示例 2：

输入： [4,1,2,1,2]

输出： 4

在真实的面试中遇到过这道题？

是

否

贡献者

相关企业

相关标签

热门标签

题目列表

随机一题

< 上一题

136/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交

i Python3

```
1 class Solution:
2     def singleNumber(self, nums: List[int]) -> int:
3         .
```





137. 只出现一次的数字 II

难度 中等 175 收藏 分享 切换为英文

通过次数 11,790 提交次数 18,334

[题目描述](#)
[评论 \(132\)](#)
[题解\(17\) New](#)
[提交记录](#)
i Python3


给定一个非空整数数组，除了某个元素只出现一次以外，其余每个元素均出现了三次。找出那个只出现了一次的元素。

说明：

你的算法应该具有线性时间复杂度。你可以不使用额外空间来实现吗？

示例 1：

输入： [2,2,3,2]

输出： 3

示例 2：

输入： [0,1,0,1,0,1,99]

输出： 99

在真实的面试中遇到过这道题？

[是](#)
[否](#)
贡献者

相关企业

相关标签


```
1 class Solution:
2     def singleNumber(self, nums: List[int]) -> int:
3
```

[举报](#)
[题目列表](#)
[随机一题](#)
[上一题](#)

137/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



138. 复制带随机指针的链表

难度 中等 126 收藏 分享 切换为英文

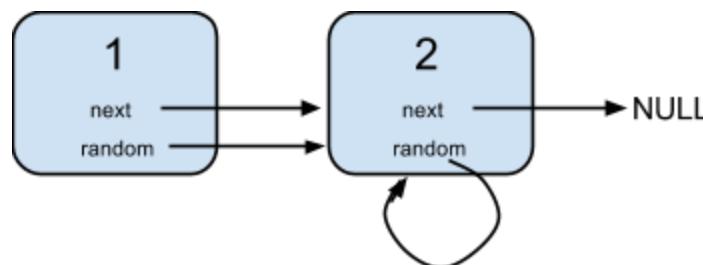
通过次数 10,498 提交次数 29,253

[题目描述](#)
[评论 \(106\)](#)
[题解\(18\) New](#)
[提交记录](#)
i Python3


给定一个链表，每个节点包含一个额外增加的随机指针，该指针可以指向链表中的任何节点或空节点。

要求返回这个链表的深拷贝。

示例：



输入：

```
{
  "$id": "1",
  "next": {
    "$id": "2",
    "next": null,
    "random": {
      "$ref": "2",
      "val": 2
    }
  },
  "random": {
    "$ref": "2",
    "val": 1
  }
}
```

解释：

节点 1 的值是 1，它的下一个指针和随机指针都指向节点 2。

节点 2 的值是 2，它的下一个指针指向 null，随机指针指向它自己。

```

1 """
2 # Definition for a Node.
3 class Node:
4     def __init__(self, val, next, random):
5         self.val = val
6         self.next = next
7         self.random = random
8 """
9 class Solution:
10     def copyRandomList(self, head: 'Node') -> 'Node':
11
  
```

[题目列表](#)
[随机一题](#)
[上一题](#)

138/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



139. 单词拆分

难度 中等 185 收藏 分享 切换为英文

通过次数 提交次数

16,994 39,841

[题目描述](#)
[评论 \(95\)](#)
[题解\(23\) New](#)
[提交记录](#)
i Python3

i { } < > ⌂ ⌃ ⌄

给定一个非空字符串 s 和一个包含非空单词列表的字典 wordDict，判定 s 是否可以被空格拆分为一个或多个在字典中出现的单词。

说明:

- 拆分时可以重复使用字典中的单词。
- 你可以假设字典中没有重复的单词。

示例 1:

输入: s = "leetcode", wordDict = ["leet", "code"]

输出: true

解释: 返回 true 因为 "leetcode" 可以被拆分成
"leet code"。

```
1 class Solution:
2     def wordBreak(self, s: str, wordDict: List[str]) -> bool:
3         pass
```

示例 2:

输入: s = "applepenapple", wordDict = ["apple",
"pen"]

输出: true

解释: 返回 true 因为 "applepenapple" 可以被拆分成
"apple pen apple"。

注意你可以重复使用字典中的单词。

示例 3:

输入: s = "catsandog", wordDict = ["cats", "dog",

[题目列表](#)
[随机一题](#)
[上一题](#)

139/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



140. 单词拆分 II

难度 困难 67 收藏 分享 切换为英文

通过次数 5,624 提交次数 15,025

[题目描述](#)
[评论 \(46\)](#)
[题解\(15\) New](#)
[提交记录](#)
i Python3 ▾

i { } < > ⌂ ⌃ ⌄

给定一个非空字符串 s 和一个包含非空单词列表的字典 wordDict，在字符串中增加空格来构建一个句子，使得句子中所有的单词都在词典中。返回所有这些可能的句子。

说明:

- 分隔时可以重复使用字典中的单词。
- 你可以假设字典中没有重复的单词。

示例 1:

输入:

```
s = "catsanddog"
wordDict = ["cat", "cats", "and", "sand", "dog"]
```

输出:

```
[ "cats and dog",
  "cat sand dog" ]
```

```
1 class Solution:
2     def wordBreak(self, s: str, wordDict: List[str]) ->
3         List[str]:
```

示例 2:

输入:

```
s = "pineapplepenapple"
wordDict = ["apple", "pen", "applepen", "pine",
"pineapple"]
```

输出:

```
[
```

[题目列表](#)
[随机一题](#)
[上一题](#)

140/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)

141. 环形链表

难度 简单 361 收藏 分享 切换为英文

通过次数 60,598 提交次数 142,536

[题目描述](#)
[评论 \(409\)](#)
[题解\(60\) New](#)
[提交记录](#)

给定一个链表，判断链表中是否有环。

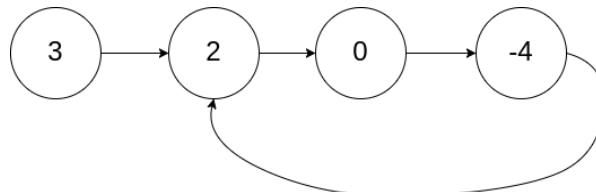
为了表示给定链表中的环，我们使用整数 `pos` 来表示链表尾连接到链表中的位置（索引从 0 开始）。如果 `pos` 是 `-1`，则在该链表中没有环。

示例 1:

输入: head = [3,2,0,-4], pos = 1

输出: true

解释: 链表中有一个环，其尾部连接到第二个节点。

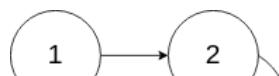


示例 2:

输入: head = [1,2], pos = 0

输出: true

解释: 链表中有一个环，其尾部连接到第一个节点。


i C++

```

1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode(int x) : val(x), next(NULL) {}
7  * };
8 */
9 class Solution {
10 public:
11     bool hasCycle(ListNode *head) {
12
13     }
14 };
  
```

[题目列表](#)
[随机一题](#)
[上一题](#)

141/1158

[下一题](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)

142. 环形链表 II

难度 中等 223 收藏 分享 切换为英文

[题目描述](#)
[评论 \(228\)](#)
[题解\(31\) New](#)
[提交记录](#)

给定一个链表，返回链表开始入环的第一个节点。如果链表无环，则返回 `null`。

为了表示给定链表中的环，我们使用整数 `pos` 来表示链表尾连接到链表中的位置（索引从 0 开始）。如果 `pos` 是 `-1`，则在该链表中没有环。

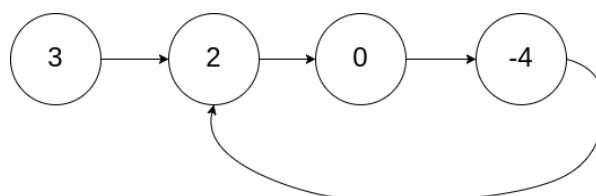
说明：不允许修改给定的链表。

示例 1:

输入: `head = [3,2,0,-4]`, `pos = 1`

输出: `tail connects to node index 1`

解释: 链表中有一个环，其尾部连接到第二个节点。



示例 2:

输入: `head = [1,2]`, `pos = 0`

输出: `tail connects to node index 0`

解释: 链表中有一个环，其尾部连接到第一个节点。

i C++ ▾

```

1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode(int x) : val(x), next(NULL) {}
7  * };
8 */
9 class Solution {
10 public:
11     ListNode *detectCycle(ListNode *head) {
12
13
14 }
  
```

[题目列表](#)
[随机一题](#)
[上一题](#)

142/1158

[下一题](#)
[控制台](#) ▾

[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



143. 重排链表

难度 中等 103 收藏 分享 切换为英文

通过次数 8,156 提交次数 15,940

[题目描述](#) [评论 \(108\)](#) [题解\(19\) New](#) [提交记录](#)

给定一个单链表 $L: L_0 \rightarrow L_1 \rightarrow \dots \rightarrow L_{n-1} \rightarrow L_n$ ，
将其重新排列后变为： $L_0 \rightarrow L_n \rightarrow L_1 \rightarrow L_{n-1} \rightarrow L_2 \rightarrow L_{n-2} \rightarrow \dots$

你不能只是单纯的改变节点内部的值，而是需要实际的进行节点交换。

示例 1:

给定链表 $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ ，重新排列为 $1 \rightarrow 4 \rightarrow 2 \rightarrow 3$.

示例 2:

给定链表 $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ ，重新排列为 $1 \rightarrow 5 \rightarrow 2 \rightarrow 4 \rightarrow 3$.

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业



相关标签

i Python3

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def reorderList(self, head: ListNode) -> None:
9         """
10             Do not return anything, modify head in-place instead.
11         """
12

```

题目列表

随机一题

< 上一题

143/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



145. 二叉树的后序遍历

难度 困难 151 收藏 分享 切换为英文

通过次数 26,883 提交次数 40,001

[题目描述](#) [评论 \(175\)](#) [题解\(34\) New](#) [提交记录](#)

给定一个二叉树，返回它的 *后序遍历*。

示例：

输入： [1,null,2,3]

```

1
 \
2
 /
3

```

输出： [3,2,1]

进阶：递归算法很简单，你可以通过迭代算法完成吗？

在真实的面试中遇到过这道题？

是

否

贡献者

相关企业

相关标签

相似题目

i Python3

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def postorderTraversal(self, root: TreeNode) -> List[int]:
10

```

[题目列表](#)

[随机一题](#)

< 上一题

145/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



146. LRU缓存机制

难度 中等 220 收藏 分享 切换为英文

[题目描述](#) [评论 \(140\)](#) [题解\(39\) New](#) [提交记录](#)

运用你所掌握的数据结构，设计和实现一个 LRU (最近最少使用) 缓存机制。它应该支持以下操作：获取数据 `get` 和写入数据 `put`。

获取数据 `get(key)` - 如果密钥 (`key`) 存在于缓存中，则获取密钥的值（总是正数），否则返回 -1。

写入数据 `put(key, value)` - 如果密钥不存在，则写入其数据值。当缓存容量达到上限时，它应该在写入新数据之前删除最近最少使用的数据值，从而为新的数据值留出空间。

进阶：

你是否可以在 $O(1)$ 时间复杂度内完成这两种操作？

示例：

```
LRUCache cache = new LRUCache( 2 /* 缓存容量 */ );

```

```
cache.put(1, 1);
cache.put(2, 2);
cache.get(1);      // 返回 1
cache.put(3, 3);  // 该操作会使得密钥 2 作废
cache.get(2);      // 返回 -1 (未找到)
cache.put(4, 4);  // 该操作会使得密钥 1 作废
cache.get(1);      // 返回 -1 (未找到)
cache.get(3);      // 返回 3
cache.get(4);      // 返回 4
```

i Python3

```

1 class LRUCache:
2
3     def __init__(self, capacity: int):
4
5
6         def get(self, key: int) -> int:
7
8
9             def put(self, key: int, value: int) -> None:
10
11
12
13     # Your LRUCache object will be instantiated and called as such:
14     # obj = LRUCache(capacity)
15     # param_1 = obj.get(key)
16     # obj.put(key,value)

```



147. 对链表进行插入排序

难度 中等 79 收藏 分享 切换为英文

通过次数 10,244 提交次数 17,142

[题目描述](#)
[评论 \(123\)](#)
[题解\(21\) ^{New}](#)
[提交记录](#)
i Python3


对链表进行插入排序。

6 5 3 1 8 7 2 4

插入排序的动画演示如上。从第一个元素开始，该链表可以被认为已经部分排序（用黑色表示）。

每次迭代时，从输入数据中移除一个元素（用红色表示），并原地将其插入到已排好序的链表中。

插入排序算法：

1. 插入排序是迭代的，每次只移动一个元素，直到所有元素可以形成一个有序的输出列表。
2. 每次迭代中，插入排序只从输入数据中移除一个待排序的元素，找到它在序列中适当的位置，并将其插入。
3. 重复直到所有输入数据插入完为止。

```

1 # Definition for singly-linked list.
2 # class ListNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.next = None
6
7 class Solution:
8     def insertionSortList(self, head: ListNode) -> ListNode:
9

```

示例 1：

[题目列表](#)
[随机一题](#)
[上一题](#)

147/1158

[下一题 >](#)
[控制台](#)
[贡献 i](#)
[▶ 执行代码](#)
[提交](#)



153. 寻找旋转排序数组中的最小值

难度 中等 84 收藏 分享 切换为英文

通过次数 15,785 提交次数 31,949

[题目描述](#)
[评论 \(132\)](#)
[题解\(32\) ^{New}](#)
[提交记录](#)
i Python3


```

1 class Solution:
2     def findMin(self, nums: List[int]) -> int:
3

```

假设按照升序排序的数组在预先未知的某个点上进行了旋转。

(例如，数组 `[0, 1, 2, 4, 5, 6, 7]` 可能变为 `[4, 5, 6, 7, 0, 1, 2]`)。

请找出其中最小的元素。

你可以假设数组中不存在重复元素。

示例 1:

输入: `[3,4,5,1,2]`

输出: 1

示例 2:

输入: `[4,5,6,7,0,1,2]`

输出: 0

在真实的面试中遇到过这道题?

贡献者



相关企业



相关标签


[题目列表](#)
[随机一题](#)
[上一题](#)

153/1158

[下一题](#)
[控制台](#)
[贡献](#)
[▶ 执行代码](#)
[提交](#)



155. 最小栈

难度 简单 268 收藏 分享 切换为英文

通过次数 40,372 提交次数 81,259

[题目描述](#)

[评论 \(262\)](#)

[题解 \(54\) ^{New}](#)

[提交记录](#)

i Python3

设计一个支持 push, pop, top 操作，并能在常数时间内检索到最小元素的栈。

- push(x) -- 将元素 x 推入栈中。
- pop() -- 删除栈顶的元素。
- top() -- 获取栈顶元素。
- getMin() -- 检索栈中的最小元素。

示例:

```
MinStack minStack = new MinStack();
minStack.push(-2);
minStack.push(0);
minStack.push(-3);
minStack.getMin();   --> 返回 -3.
minStack.pop();
minStack.top();      --> 返回 0.
minStack.getMin();   --> 返回 -2.
```

在真实的面试中遇到过这道题?

是

否

贡献者



相关企业

相关标签

```

1 class MinStack:
2
3     def __init__(self):
4         """
5             initialize your data structure here.
6         """
7
8
9     def push(self, x: int) -> None:
10
11
12     def pop(self) -> None:
13
14
15     def top(self) -> int:
16
17
18     def getMin(self) -> int:
19
20
21
22 # Your MinStack object will be instantiated and called as such:
23 # obj = MinStack()
24 # obj.push(x)
25 # obj.pop()
26 # param_3 = obj.top()
27 # param_4 = obj.getMin()

```

[题目列表](#)

[随机一题](#)

< 上一题

155/1158

下一题 >

控制台

贡献 i

▶ 执行代码

提交



666. 路径和 IV

通过次数

提交次数

难度 中等

4



收藏

分享

切换为英文

102

153

[题目描述](#)[评论 \(5\)](#)[题解 New](#)[提交记录](#)[i](#)[i](#) [{}
o](#) [o
o](#) [o
o](#)

联系



升级账号解锁题目

[前往会员中心](#)

1

[题目列表](#)[随机一题](#)[上一题](#)

666/1158

[下一题 >](#)[执行代码](#)[提交](#)