Practice

任务一:



9. 回文数



△ 相关企业

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。
- 分析题目: 判断将数字倒序读与顺序读是否相等
- •第一次选择方法: i. 将数字拆分成列表
 - ii. 进行顺序和倒序判断
- 检查逻辑漏洞: 若数字为负数则无法判断
- 第二次选择方法: i. 将数字转化成字符串
 - ii. 利用序列倒序切片的方法将字符串反转
 - iii.判断
- 参考答案分析: 将数字对半翻转,并判断是否与前半段数字相同



1773. 统计匹配检索规则的物品数量



△ 相关企业

给你一个数组 items , 其中 items[i] = [typei, colori, namei] , 描述第 i 件物品的类型、颜色以及名称。 另给你一条由两个字符串 ruleKey 和 ruleValue 表示的检索规则。

如果第 1 件物品能满足下述条件之一,则认为该物品与给定的检索规则 匹配:

- ruleKey == "type" 且 ruleValue == typei。
- ruleKey == "color" $oxed{\exists}$ ruleValue == color_i .
- ruleKey == "name" 且 ruleValue == name;

统计并返回 匹配检索规则的物品数量。

- 分析题目: 检查 items 列表中每个元素列表中的 ruleKey 是否等于 ruleValue,并返回元素列表的数量
- 选择方法: i. 创建一个字典来表示 ruleKey 在每个元素列表中的索引
 - ii. 设置一个 num 来记录符合条件的元素
 - iii.用 for 循环遍历 items
 - iv. 判断
- · 参考答案分析: 用哈希表将 ruleKey 的索引表达出来,用推导式来遍历 items, 最后 sum 求和



2114. 句子中的最多单词数





一个 句子 由一些 单词 以及它们之间的单个空格组成,句子的开头和结尾不会有多余空格。

给你一个字符串数组 sentences , 其中 sentences [i] 表示单个 句子。

请你返回单个句子里 单词的最多数目。

- 分析题目: 找出句子中单词数最多的句子, 返回其单词数
- · 选择方法: i. 设置一个 1st 统计每个句子的空格数
 - ii. for 循环遍历列表,并计算空格的数量
 - iii. 用 max 选出最大空格数,再加一得到单词数,并返回



1252. 奇数值单元格的数目

简单 ❸ 凸 149 ☆ ♂

△ 相关企业

给你一个 $m \times n$ 的矩阵,最开始的时候,每个单元格中的值都是 0。

另有一个二维索引数组 indices, indices[i] = [ri, ci] 指向矩阵中的某个位置,其中 ri 和 ci 分别表示指定的行和 列(从 0 开始编号)。

提示 💮

对 indices[i] 所指向的每个位置,应同时执行下述增量操作:

- 1. ri 行上的所有单元格,加1。
- 2. ci 列上的所有单元格,加1。

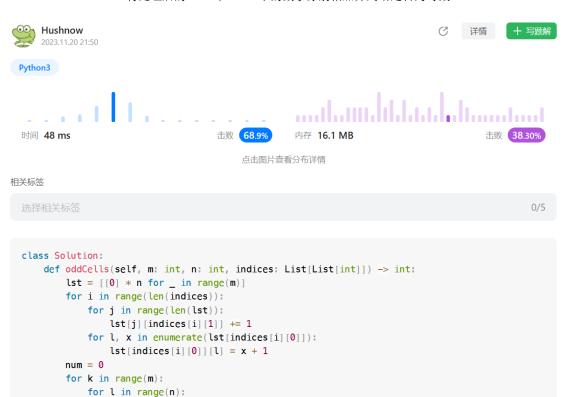
给你 m、n 和 indices 。请你在执行完所有 indices 指定的增量操作后,返回矩阵中 奇数值单元格 的数目。

- •分析题目:将一个m*n 且初始参数为 0 的矩阵按照一定格式在行和列上分别加一,并返回矩阵中奇数的数量
- ·选择方法: i. 先建立一个初始矩阵
 - ii. 用 for 循环先遍历 indices 列表以获得需要处理的行和列
 - iii.用 for 循环遍历矩阵,并根据上步得到的数据进行加一处理
 - iv. 用 for 循环遍历得到的新矩阵,并记录奇数的数量
- 错误总结: i. 不能用[[0 * m] * n]来获得矩阵,否则会形成将 0 * m 的结果重复 3 遍的列表
 - ii. 不能用[[0] * m] * n 来获得列表,会形成浅拷贝,使形成的列表有关联
- 参考答案分析: i. 建立横行 rows 和竖行 cols 两个列表

if lst[k][l] % 2 == 1:
 num += 1

return num

- ii. 根据 indices 列表在将上一步的列表进行处理
- iii. 将处理后的 rows 和 cols 中的数字分别相加并判断是否为奇数



1652. 拆炸弹



△ 相关企业

你有一个炸弹需要拆除,时间紧迫! 你的情报员会给你一个长度为 n 的 循环 数组 code 以及一个密钥 k 。 为了获得正确的密码,你需要替换掉每一个数字。所有数字会 同时 被替换。

- 如果 k > 0 , 将第 i 个数字用 接下来 k 个数字之和替换。
- 如果 k < 0 , 将第 i 个数字用 **之前** k 个数字之和替换。
- 如果 k == 0 , 将第 i 个数字用 0 替换。

由于 code 是循环的, code[n-1] 下一个元素是 code[0] ,且 code[0] 前一个元素是 code[n-1] 。

给你循环数组 code 和整数密钥 k , 请你返回解密后的结果来拆除炸弹!

- 分析题目:根据 k 的值来将 code 中的数字依次进行处理,并得到密码
- · 选择方法: i. 建立一个列表记录密码
 - ii. 判断 k 的值
 - iii. 利用切片按题目规律来提取数据并求和
- 参考答案分析: i. 使用滑动窗口的方法(将 code 自身相加)
 - ii. 设置好遍历的范围,并根据 k 的值来处理数据



相关标签

```
选择相关标签
```

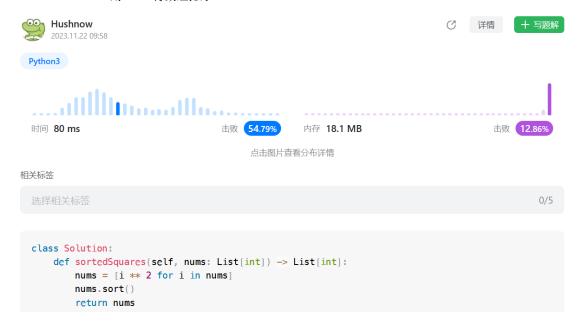
```
6
class Solution:
    def decrypt(self, code: List[int], k: int) -> List[int]:
       result = []
       if k > 0:
           for a in range(len(code)):
               if len(code) - k - a - 1 >= 0:
                   result.append(sum(code[a + 1:a + 1 + k]))
               else:
                   result.append(sum(code[a + 1:len(code)]) + sum(code[:k - len(code) + a + 1
       elif k == 0:
           result = [0 for _ in range(len(code))]
       else:
           k = -k
            for b in range(len(code)):
               if b < k:
                   result.append(sum(code[b - k:] + code[:b]))
               else:
                   result_append(sum(code[b - k:b]))
        return result
```

977. 有序数组的平方



给你一个按**非递减顺序** 排序的整数数组 nums ,返回 **每个数字的平方** 组成的新数组,要求也按**非递减顺序** 排序。

- 分析题目: 将给定数组中的数字全部平方,并将其降序排列
- 选择方法: i. 用列表推导式将给定数组平方化
 - ii. 用 sort 将数组排序



1184. 公交站间的距离

简单 ❷ 凸 102 ☆ ②

△ 相关企业

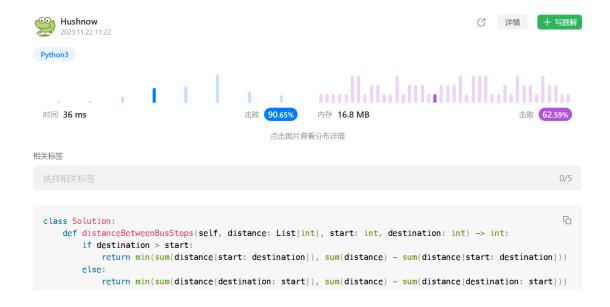
环形公交路线上有 n 个站,按次序从 0 到 n - 1 进行编号。我们已知每一对相邻公交站之间的距离,distance[i] 表示编号为 i 的车站和编号为 (i+1) % n 的车站之间的距离。

提示 ○

环线上的公交车都可以按顺时针和逆时针的方向行驶。

返回乘客从出发点 start 到目的地 destination 之间的最短距离。

- •题目分析: 求出给出两个公交站间的最小距离
- 思路转换:将题目转换成直接算出两站距离,再用总距离减去
- 选择方法: i. 判断 start 和 destination 的大小
 - ii. 按将 start 到 destination 切片 distance
 - iii. 将切片求和
 - iv. 用总距离减去切片和,并判断出最小量



1431. 拥有最多糖果的孩子

提示 💮

简单 ② 凸 177 ☆ ②

△ 相关企业

给你一个数组 candies 和一个整数 extraCandies , 其中 candies[i] 代表第 i 个孩子拥有的糖果数目。

对每一个孩子,检查是否存在一种方案,将额外的 extraCandies 个糖果分配给孩子们之后,此孩子有 **最多** 的糖果。注意,允许有多个孩子同时拥有 **最多** 的糖果数目。

示例 1:

输入: candies = [2,3,5,1,3], extraCandies = 3

输出: [true, true, true, false, true]

解释:

孩子 1 有 2 个糖果, 如果他得到所有额外的糖果 $(3 \land)$, 那么他总共有 5 个糖果, 他将成为拥有最多糖果的孩子。

孩子 2 有 3 个糖果,如果他得到至少 2 个额外糖果,那么他将成为拥有最多糖果的孩子。

孩子 3 有 5 个糖果, 他已经是拥有最多糖果的孩子。

孩子 4 有 1 个糖果,即使他得到所有额外的糖果,他也只有 4 个糖果,无法成为拥有糖果最多的孩子。

孩子 5 有 3 个糖果,如果他得到至少 2 个额外糖果,那么他将成为拥有最多糖果的孩子。

- 题目分析: 判断列表每个数字加上某个数字后是否为列表中最大
- · 选择方法: i. 建立一个表格用于记录判断结果
 - ii. 用 enumerate 函数获得各数字的索引
 - iii. 利用索引获得相应的值并加上所给值,并与列表中的最大值相比



605. 种花问题

简单 ❷ 凸 688 ☆ ②

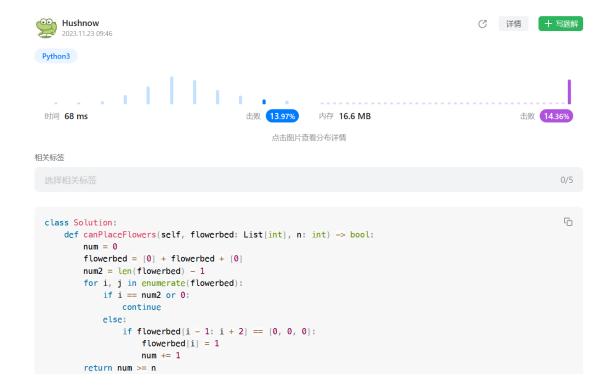
6 相关企业

假设有一个很长的花坛,一部分地块种植了花,另一部分却没有。可是,花不能种植在相邻的地块上,它们会争夺水源,两者都会死去。

0

给你一个整数数组 flowerbed 表示花坛,由若干 0 和 1 组成,其中 0 表示没种植花,1 表示种植了花。另有一个数 n ,能否在不打破种植规则的情况下种入 n 朵花? 能则返回 true ,不能则返回 false 。

- 题目分析: 在非相邻的花盆中种花,并判断是否可以种入 n 盆花
- 选择方法: i. 在"花坛"两头加上 0 建立"新花坛"
 - ii. 从左往右遍历"花坛", 在符合条件的地方将"花"插下
 - iii. 记录下所插"花"的数量,并与n相比
- ·参考答案分析: i. 从贪心角度,尽可能种更多的花
 - ii. 利用两花间空余位置(p)的奇偶来计算可种植的花
 - iii.奇: (p 1) / 2 偶: (p 1) // 2
 - iv. 用循环遍历花坛,并记录种植花的数量



面试题 16.11. 种花问题

简单 112 ☆ ♂

△ 相关企业

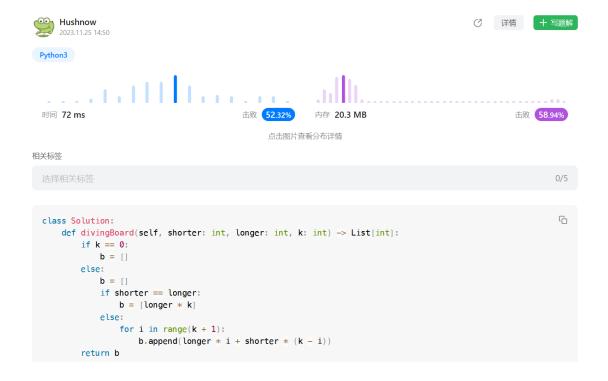
假设有一个很长的花坛,一部分地块种植了花,另一部分却没有。可是,花不能种植在相邻的地块上,它们会争夺水源,两者都会死去。

给你一个整数数组 flowerbed 表示花坛,由若干 0 和 1 组成,其中 0 表示没种植花,1 表示种植了花。另有一个数 n ,能否在不打破种植规则的情况下种入 n 朵花? 能则返回 true ,不能则返回 false 。

- 题目分析:将 shorter和 longer两种板取出共 k块,并输出所有可能出现的长度
- · 选择方法: i. 建立空列表用于记录排列结果
 - ii. 将 shorter 板从 k 块开始减少至 0,longer 板从 0 开始增加至 k 块,并将总长度添加至 列表

0

•错误分析: 未考虑 k=0 和 shorter=longer 两种情况



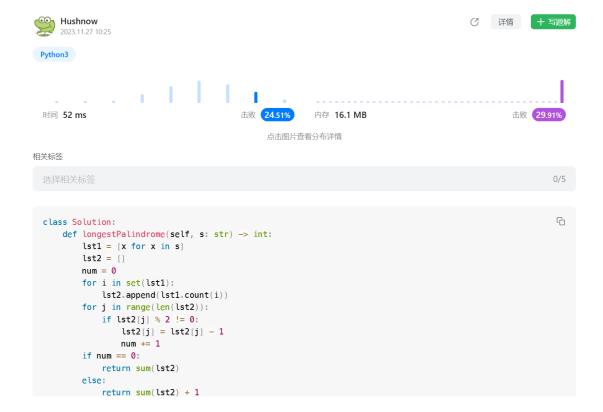
409. 最长回文串

简单 ❸ 凸 572 ☆ ♂

△ 相关企业

给定一个包含大写字母和小写字母的字符串 s , 返回 *通过这些字母构造成的 最长的回文串* 。 在构造过程中,请注意 **区分大小写** 。比如 "Aa" 不能当做一个回文字符串。

- •题目分析:将字符串 s 可以构成的最长回文串(即左右对称的字符串)的长度输出
- 方法选择: i. 建立一个列表 lst1,将字符串 s 中的字母逐个提出
 - ii. 用 for 循环和 lstl 集合化去重,将 lstl 中的字母数量全部数出并记录在 lst2 中 iii. 将 lst2 中的奇数全部减一后,将 lst2 求和并返回结果
- 错误分析: 回文串中间可以有单个字母



任务二:

 \odot

中等 **⊘ △** 1.2K **☆ ⊘**

△ 相关企业

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
Χ	10
L	50
С	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。

给你一个整数,将其转为罗马数字。

- 题目分析:输入一个整数,将其转化为罗马数字,其中一般罗马小的数字在大的数字右边,除了 4 和 9 等特殊数字,因此可将整数分为(0,3],(3,5],(5,8],9 四个区间进行判断
- 方法选择: i. 建立列表,分别记录罗马数字中的个位,十位…
 - ii. 将整数的每个数字取出,并倒序(显然从个位开始更加方便),得到1st列表
 - iii. 建立空字符串,用于记录所得到的罗马数字
 - iv. 遍历 lst, 判断其所在区间, 并做出相应处理
 - v. 将得到的罗马数字倒序并输出其长度
- •参考答案分析: i. 建立列表,记录每个特殊的值和对应罗马符号(1,4,5,9)
 - ii. 将 num 从大到小开始减去特殊值,并添加相应的罗马符号
 - iii. 直到 num 减为 0 时,输出得到的罗马数字



31. 下一个排列 ⊙

中等 ⊘ 凸 2.4K ☆ ♂

△ 相关企业

整数数组的一个排列 就是将其所有成员以序列或线性顺序排列。

• 例如, arr = [1,2,3] ,以下这些都可以视作 arr 的排列: [1,2,3] 、[1,3,2] 、[2,3,1] 。

整数数组的 下一个排列 是指其整数的下一个字典序更大的排列。更正式地,如果数组的所有排列根据其字典顺序从小到大排列在一个容器中,那么数组的 下一个排列 就是在这个有序容器中排在它后面的那个排列。如果不存在下一个更大的排列,那么这个数组必须重排为字典序最小的排列(即,其元素按升序排列)。

- 例如, arr = [1,2,3] 的下一个排列是 [1,3,2] 。
- 类似地, arr = [2,3,1] 的下一个排列是 [3,1,2] 。
- 而 arr = [3,2,1] 的下一个排列是 [1,2,3] , 因为 [3,2,1] 不存在一个字典序更大的排列。

给你一个整数数组 nums , 找出 nums 的下一个排列。

必须 原地 修改, 只允许使用额外常数空间。

- •题目分析:给出一个列表 nums,给出比 nums 字典序大的下一个列表,若已经是最大字典序,则将其改为最小字典序状态,且 nums 必须原地修改,只能使用额外的常数空间
- 第一次方法选择: 从后往前遍历 nums, 如果后一位数比前一位数大就交换位置

- •错误分析:数字改变过多,不为'下一排列'
- 第二次方法选择: i. 第一步先判断是否为最大字典序,是则将 nums 倒序并输出 None, 否则进入第二 步
 - ii. 第二步从后往前(除了第一位数)遍历 nums,若后数比前数大,则直接输出 None, 否则进入第三步
 - iii. 第三步取出第一个比第一位数大的数并交换位置,在将剩下的数字顺序排列后, 将取出的数字添加到第一位
- 错误分析: 只考虑了第一位的改变, 若 nums 超过 3 位则不符合条件
- 第三次方法选择: i. 第一步仍为判断是否为最大字典序与第二种方法相同
 - ii. 用两个循环来将 nums 从后往前切片,并判断顺序排列后的切片中是否有比前一位数大的数,若有则将第一个比前数大的数与前数交换位置



540. 有序数组中的单一元素





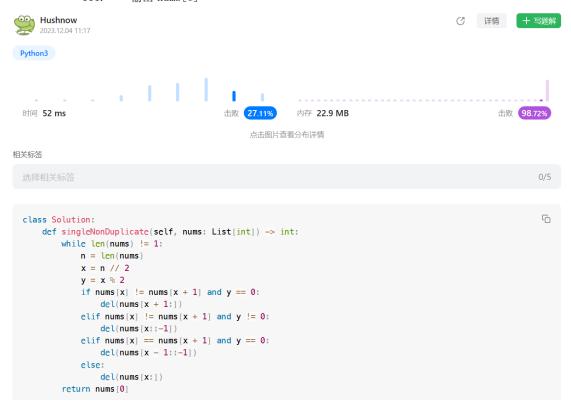
给你一个仅由整数组成的有序数组,其中每个元素都会出现两次,唯有一个数只会出现一次。 请你找出并返回只出现一次的那个数。

你设计的解决方案必须满足 O(log n) 时间复杂度和 O(1) 空间复杂度。

- 题目分析: i. 在数组 nums 中每个元素出现两次,只有一个单一元素,要求找出这个元素,且时间复 杂度满足 0(log n), 空间复杂度满足 0(1)。
 - ii. 由时间复杂度可知题目要求用二分查找
- 寻找规律: 发现单一元素所在位置与最中间的数字 (nums[x]) 和数字数量的奇偶性有一定关系, 可分为

四种情况:

- i. 与右数不等,右边数字数量为偶数
- ii. 不等,奇数
- iii. 相等,偶数
- iv. 相等,奇数
- 方法选择: 因此可根据情况的不同来删除相应的数组切片, 但保持留下的数组始终由出现两次的元素和 单一元素组成
 - i. 用 while 循环使数组长度等于 1 时停止
 - ii. 对 nums 进行删除处理
 - iii. 输出 nums[0]



任务三:

37. 解数独

困难 ② △ 1.8K ☆ ♂

△ 相关企业

编写一个程序,通过填充空格来解决数独问题。

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。(请参考示例图)

数独部分空格内已填入了数字,空白格用 1.1 表示。

示例 1:

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

- 题目分析: 在输入的数组棋盘中填入数字
 - 数独规则: i. 同一行和同一列上不能出现重复数字
 - ii. 九宫格内不能有重复数字
 - iii. 数字只能是 1-9
- 选择方法: 显然需要枚举出多种可能的答案,并输出唯一答案,此时使用回溯法
 - i. 定义一个函数用于检查三种情况是否有重复数字,并返回布尔值
 - ii. 定义一个主函数用于遍历数字(回溯)
- 函数构建: i. 检查函数: a. 行和列直接通过所在坐标来进行遍历检查
 - b. 九宫格用 (x // 3) * 3 和 (y // 3) * 3 来得到所在九宫格坐标后遍历 检查
 - ii. 主函数: a. 首先对 x, y 进行判断, 看是否需要换行, 并进行处理

- b. 判断在(x, y)位置是否已有数字,若有则跳过该位置
- c. 在上述处理后遍历 1-9 数字并检查是否符合填写要求
- d. 再次调用主函数, 并使 x 加一, 进行第二步判断
- e. 若在之后有不符合条件的数字则返回该步进行初始化处理



点击图片查看分布详情

相关标签

```
class Solution:
   def solveSudoku(self, board: List[List[str]]) -> None:
       def check_board(num, x, y):
           for i in range(9):
               if board[y][i] == str(num):
                   return False
               if board[i][x] == str(num):
                   return False
               if board[(y//3)*3 + i // 3][(x//3)*3 + i % 3] == str(num):
                   return False
           return True
       def Sudoku(board, x, y):
           if x == 9:
               return Sudoku(board, 0, y + 1)
           if y == 9:
               return True
           if board[y][x] != ".":
               return Sudoku(board, x + 1, y)
           for num in range(1, 10):
               if not check_board(num, x, y):
                   continue
               board[y][x] = str(num)
               if Sudoku(board, x + 1, y):
                  return True
               board[y][x] = "."
       Sudoku(board, 0, 0)
```