**## 1. 题目**

**### 136.只出现一次的数字**

bit manipulation, https://leetcode.cn/problems/single-number/

<mark>请用位操作来实现，并且只使用常量额外空间。</mark>

思路：用了去年学过的异或计算，复习了一次二进制。

代码：

class Solution(object):

    def singleNumber(self, nums):

        # nums=list(map(int,input().split()))

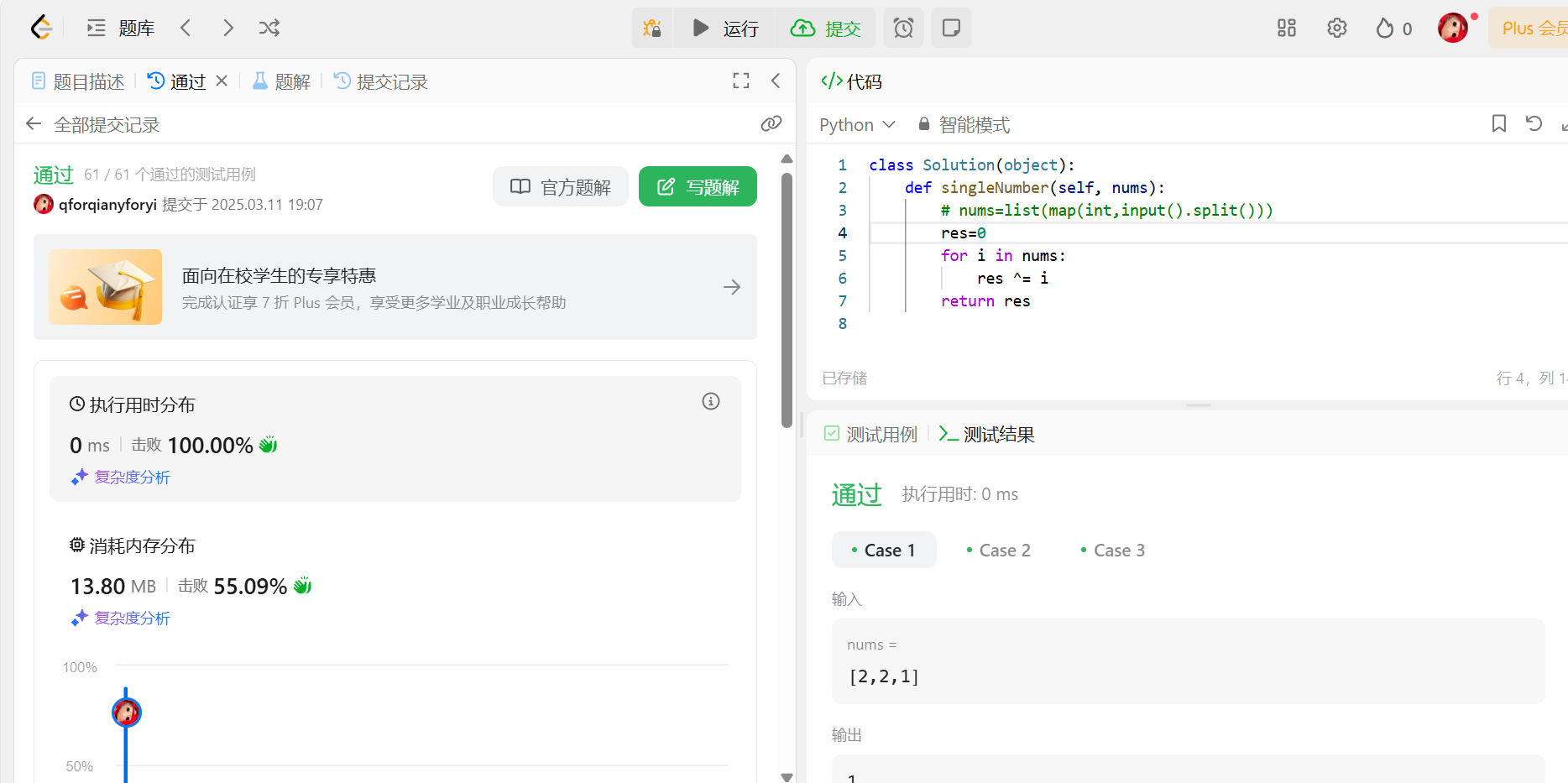
        res=0

        for i in nums:

            res ^= i

        return res

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

  
大约用时：15分钟

**### 20140:今日化学论文**

stack, http://cs101.openjudge.cn/practice/20140/

思路：遍历字符串，遇到 ] 时向前弹出直到遇到 [，将括号内的内容存入 words列表，再提取数字ans，通过words\*ans找到内容的重复次数，并将内容压回栈中，循环直到所有[ ]都被检查。

代码：

word=input()  
stack=[]  
words=[]  
ans=''  
for i in range(len(word)):  
 stack.append(word[i])  
 if word[i] == ']':  
 stack.pop()  
 while stack[-1] != '[':  
 words.append(stack.pop())  
 stack.pop()  
 ans = ''  
 while words[-1].isdigit():  
 ans+=str(words.pop())  
 words\*=int(ans)  
 while words:  
 stack.append(words.pop())  
print(''.join(stack))

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

  
大约用时：1小时

**### 160.相交链表**

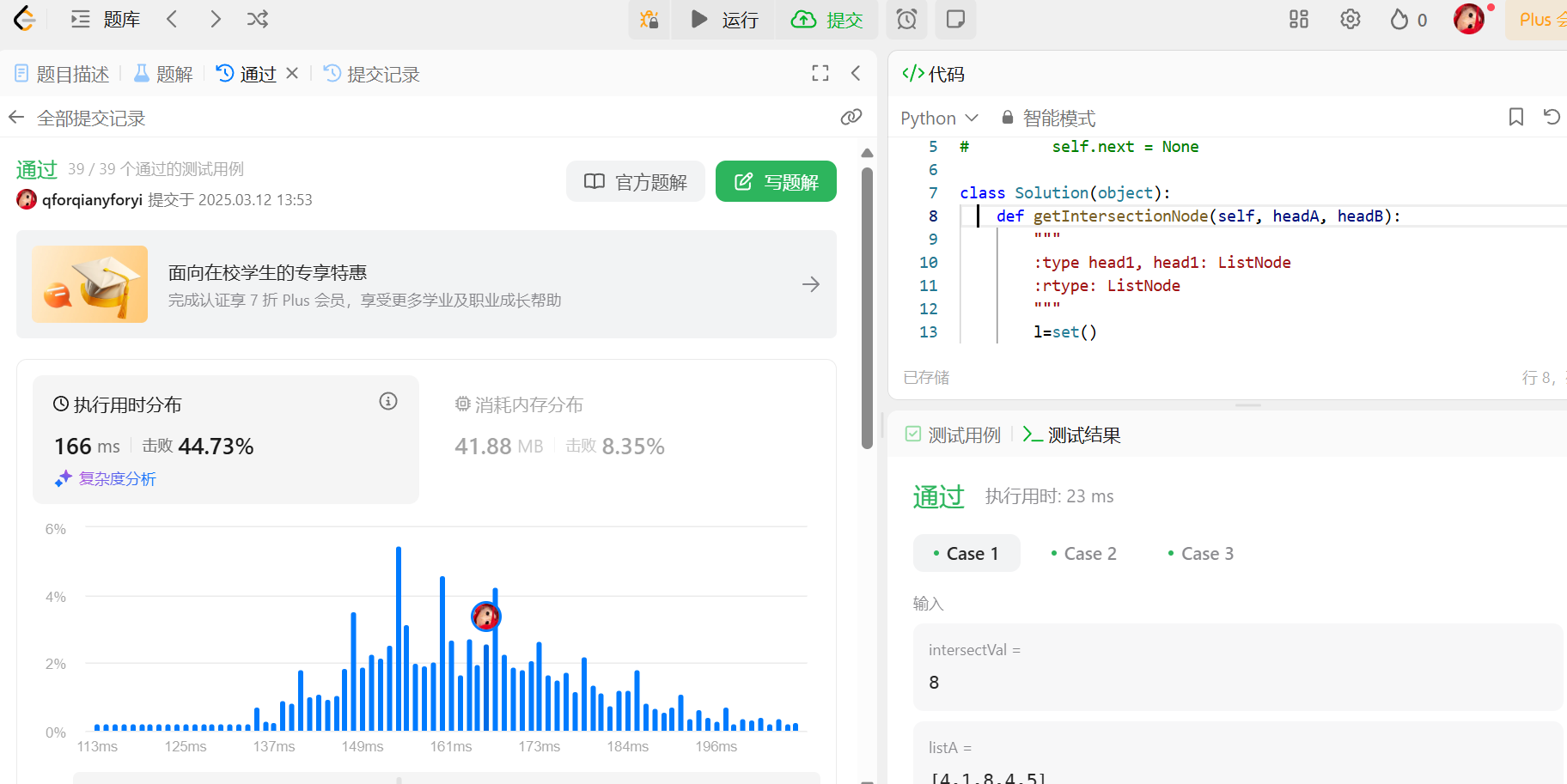
linked list, https://leetcode.cn/problems/intersection-of-two-linked-lists/

思路：用哈希表，将一个链表先储存到哈希表中，再遍历另一个链表找到交点。

代码：

# Definition for singly-linked list.  
# class ListNode(object):  
# def \_\_init\_\_(self, x):  
# self.val = x  
# self.next = None  
class Solution(object):  
 def getIntersectionNode(self, headA, headB):  
 *"""  
 :type head1, head1: ListNode  
 :rtype: ListNode  
 """* l=set()  
 a,b=headA,headB  
 while a:  
 l.add(a)  
 a=a.next  
 while b:  
 if b in l:  
 return b  
 b=b.next  
 return None

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

  
大约用时：30分钟

**### 206.反转链表**

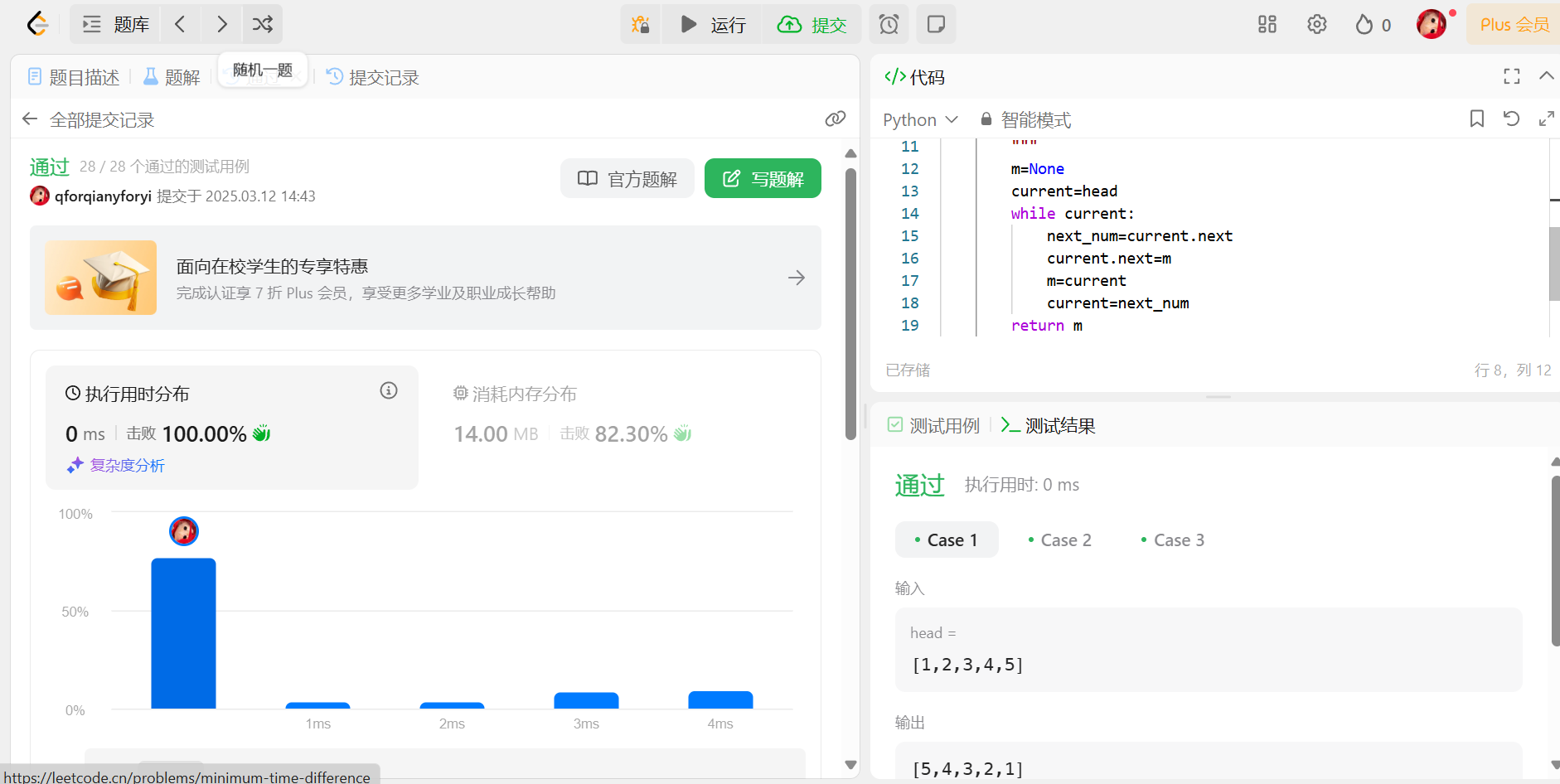
linked list, https://leetcode.cn/problems/reverse-linked-list/

思路：用迭代方法，遍历链表的过程中，每个节点通过.next指向前一个节点，使得节点顺序被反转，最后一个元素成为头节点（最后一个变成null？）。

代码：

# Definition for singly-linked list.  
# class ListNode(object):  
# def \_\_init\_\_(self, val=0, next=None):  
# self.val = val  
# self.next = next  
class Solution(object):  
 def reverseList(self, head):  
 *"""  
 :type head: Optional[ListNode]  
 :rtype: Optional[ListNode]  
 """* m=None  
 current=head  
 while current:  
 next\_num=current.next  
 current.next=m  
 m=current  
 current=next\_num  
 return m

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

  
大约用时：30分钟

**### 3478.选出和最大的K个元素**

heap, https://leetcode.cn/problems/choose-k-elements-with-maximum-sum/

思路：先按nums1升序排列其索引成index列表，遍历这个列表找到原始索引下对应的nums1&2的值，并将最大的k个nums2列表中的值存入最小堆中，同时计算总和；如果当前nums1[j]和前一个数相同则直接继承ans[ind]，避免重复计算，最后每次添加nums2[j]到堆中，超过k个时移除最小值。

代码：

class Solution(object):

    def findMaxSum(self, nums1, nums2, k):

        """

        :type nums1: List[int]

        :type nums2: List[int]

        :type k: int

        :rtype: List[int]

        """

# import heapq

# nums1=list(map(int,input().split()))

# nums2=list(map(int,input().split()))

# k=int(input())

        ans=[0]\*len(nums1)

        index=sorted(range(len(nums1)),key=lambda x:nums1[x])

        min\_heap=[]

        total=0

        for i,j in enumerate(index):

            n1=nums1[j]

            n2=nums2[j]

            ind=index[i-1]

            if i and n1 == nums1[ind]:

                ans[j] = ans[ind]

            else:

                ans[j]=total

            total+=n2

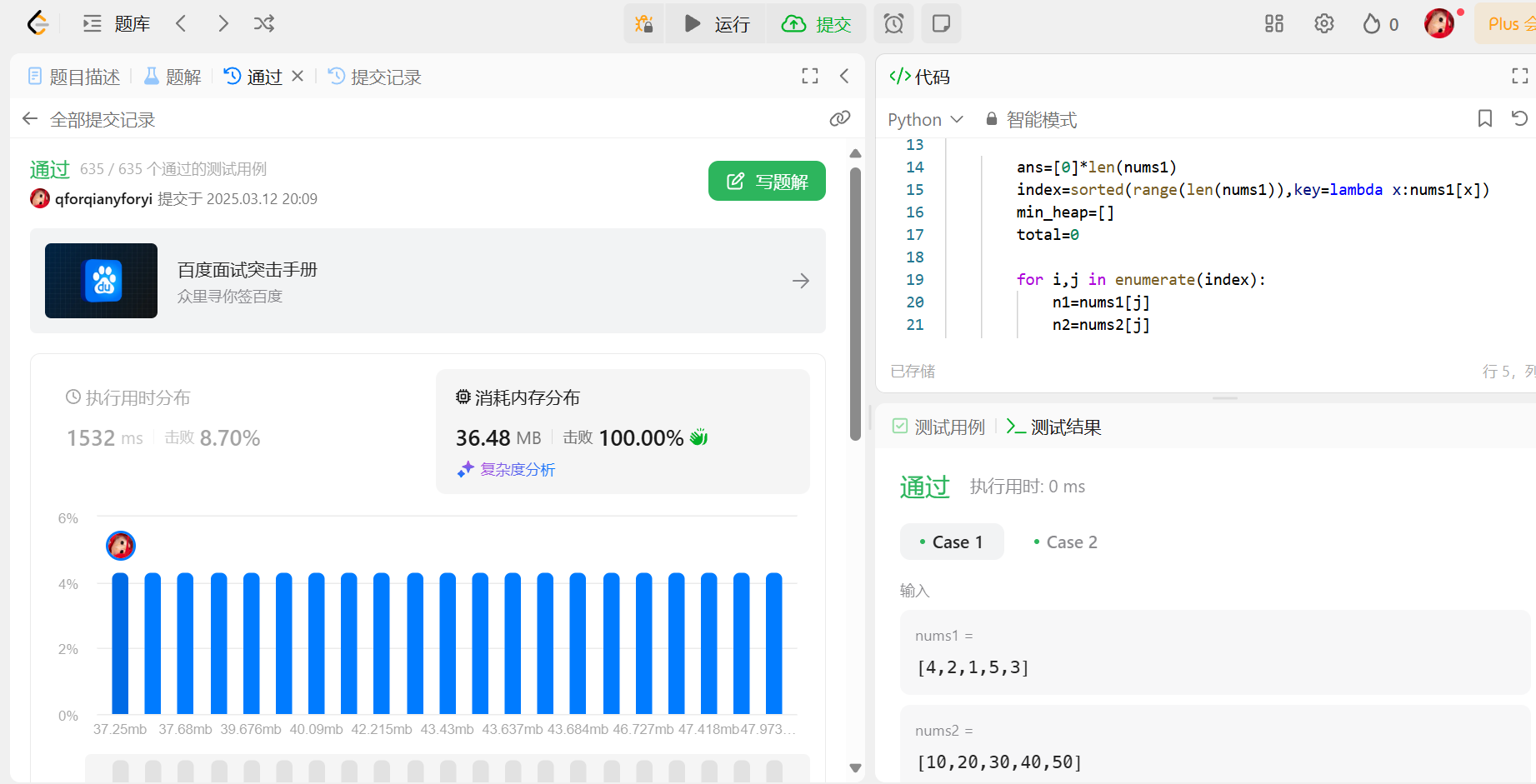
            heapq.heappush(min\_heap,n2)

            if len(min\_heap)>k:

                total-=heapq.heappop(min\_heap)

        return ans

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

  
大约用时：1小时

**## 2. 学习总结和收获**

<mark>如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算2025spring每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。</mark>

第二题有点难受，没想到可以用向前弹出的方法，就死磕在找到]后向前找[并将整段[]内的字符串替换成重复字符串的方法，自己找了样例运行都是对的，但不知道为什么提交一直是WA，之后问了朋友才知道这个方法，就用这个方法来做了。第三四题一开始没看好题目直接进pycharm很轻松就做出来了，结果发现把代码复制进leetcode后不是那么回事hhh，也是第一次接触.next和.val，知道了链表的生成原理，虽然不是很会但感觉会是蛮简单的题目，但还是让第一次接触链表的我头脑过载了。第五题还是heap，尝试自己做但是还是错了www，所以最后还是参考了题解，但比起上一次已经能更容易明白代码在说什么了。