Assignment #4: 位操作、栈、链表、堆和NN

Updated 1203 GMT+8 Mar 10, 2025

2025 spring, Complied by 颜鼎堃 工学院

> **说明：**
>
> 1. **解题与记录：**
>
>   对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge，Codeforces，LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora https://typoraio.cn 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。
>
> 2. **提交安排：**提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。
>
> 3. **延迟提交：**如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。
>
> 请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

136.只出现一次的数字

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

请用位操作来实现，并且只使用常量额外空间。

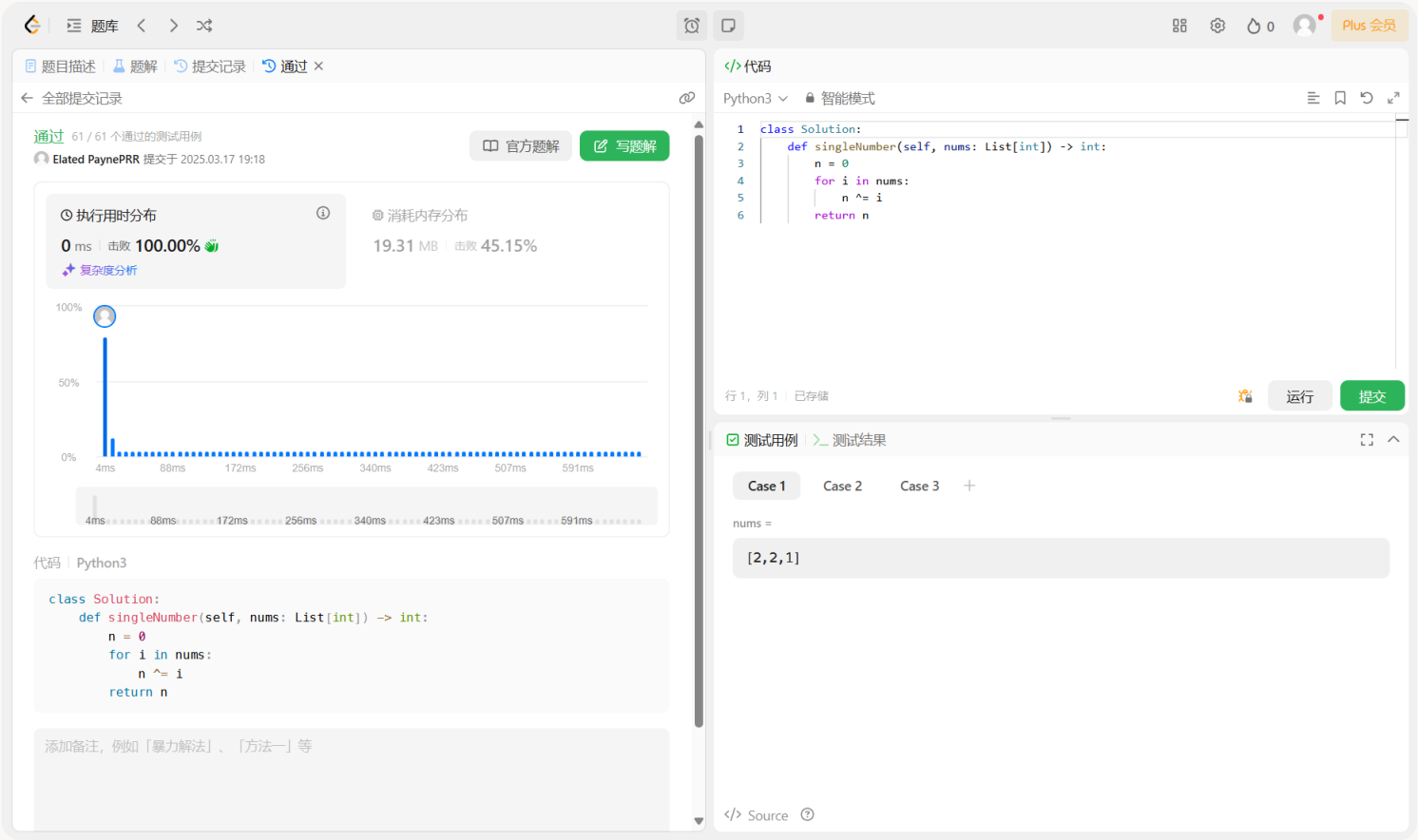
- 其实第四行换成 `n = -1`，第七行换成 `return ~n` 也可以，反正是一个道理

代码：

```python
from typing import *
class Solution:
    def singleNumber(self, nums: List[int]) -> int:
        n = 0
        for i in nums:
            n ^= i
        return n

if __name__ == '__main__':
    sol = Solution()
    print(sol.singleNumber([2, 4, 2, 4, 1]))
```

代码运行截图 （至少包含有"Accepted"）



20140:今日化学论文

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

思路：

- 用栈来跟踪括号
- 中间的代码还是稍微丑陋了点

代码：

```python
stack = []
for c in input():
    if c != "]":
        stack.append(c)
    else:
        temp = []
        t = 0
        while((n := stack.pop()) != "["):
            temp.append(n)
        for i in range(len(temp)-1, len(temp)-5, -1):
            if not temp[i].isdigit():
                t = int("".join(reversed(temp[i+1:])))
                temp = list(reversed(temp[:i+1]))
                break
        for i in range(t):
            stack += temp
print("".join(stack))
```

代码运行截图 （至少包含有"Accepted"）

160.相交链表

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

思路：

- 确实没想出来空间复杂度$O(1)$的写法，我总想着是不是和异或有点关系，总也想不出来，最后看了题解发现是双指针
- 注释掉的是我提交的代码

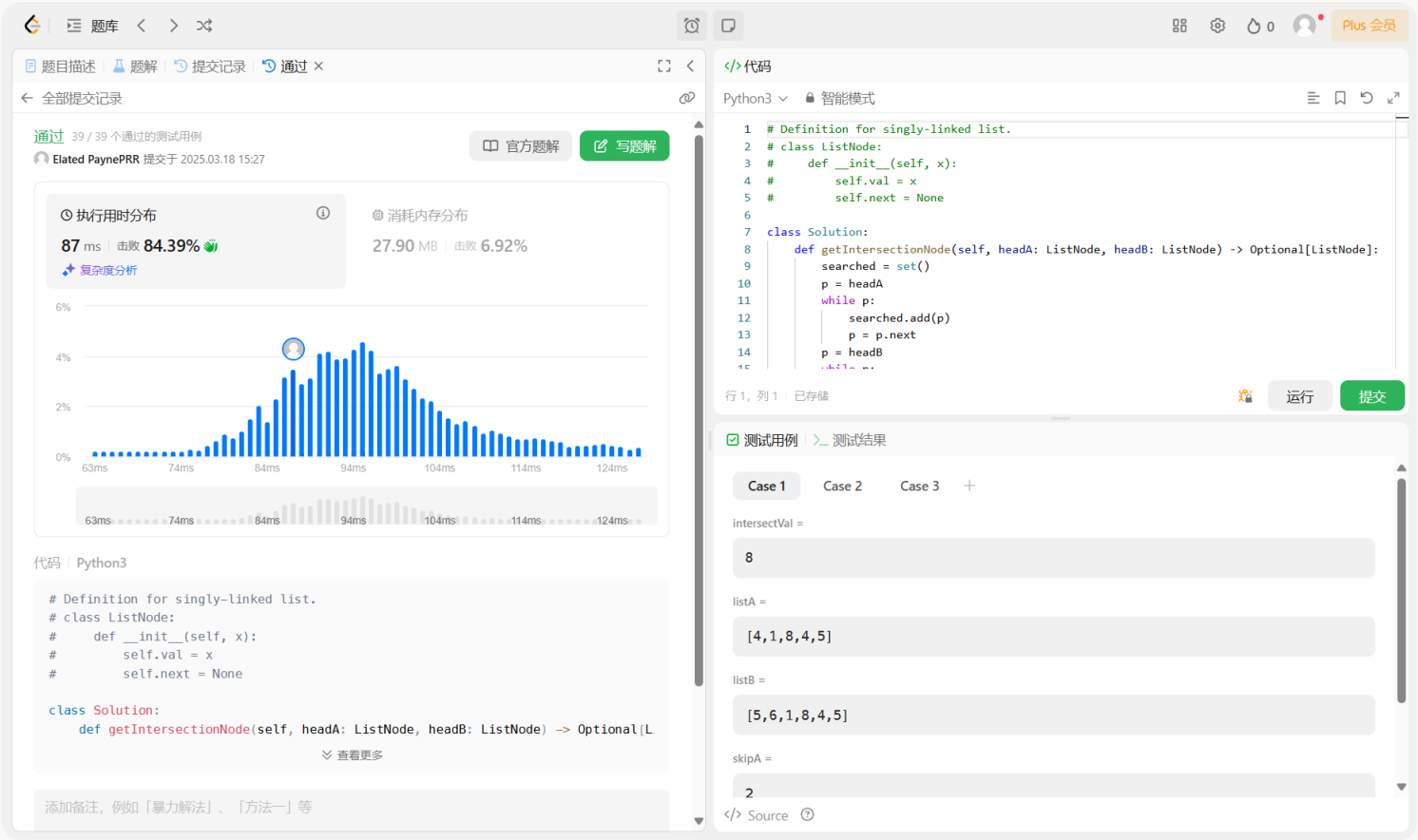
代码：

```python
# Definition for singly-linked list.
class ListNode:
    def __init__(self, x):
        self.val = x
        self.next = None

class Solution:
    def getIntersectionNode(self, headA: ListNode, headB: ListNode) -> Optional[ListNode]:
        # searched = set()
        # p = headA
        # while p:
        #     searched.add(p)
        #     p = p.next
        # p = headB
        # while p:
        #     if p in searched:
        #         return p
        #     p = p.next
        # return None

        A, B = headA, headB
        while A != B:
            A = A.next if A else headB
            B = B.next if B else headA
        return A
```

代码运行截图（至少包含有"Accepted"）



206.反转链表

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

思路：

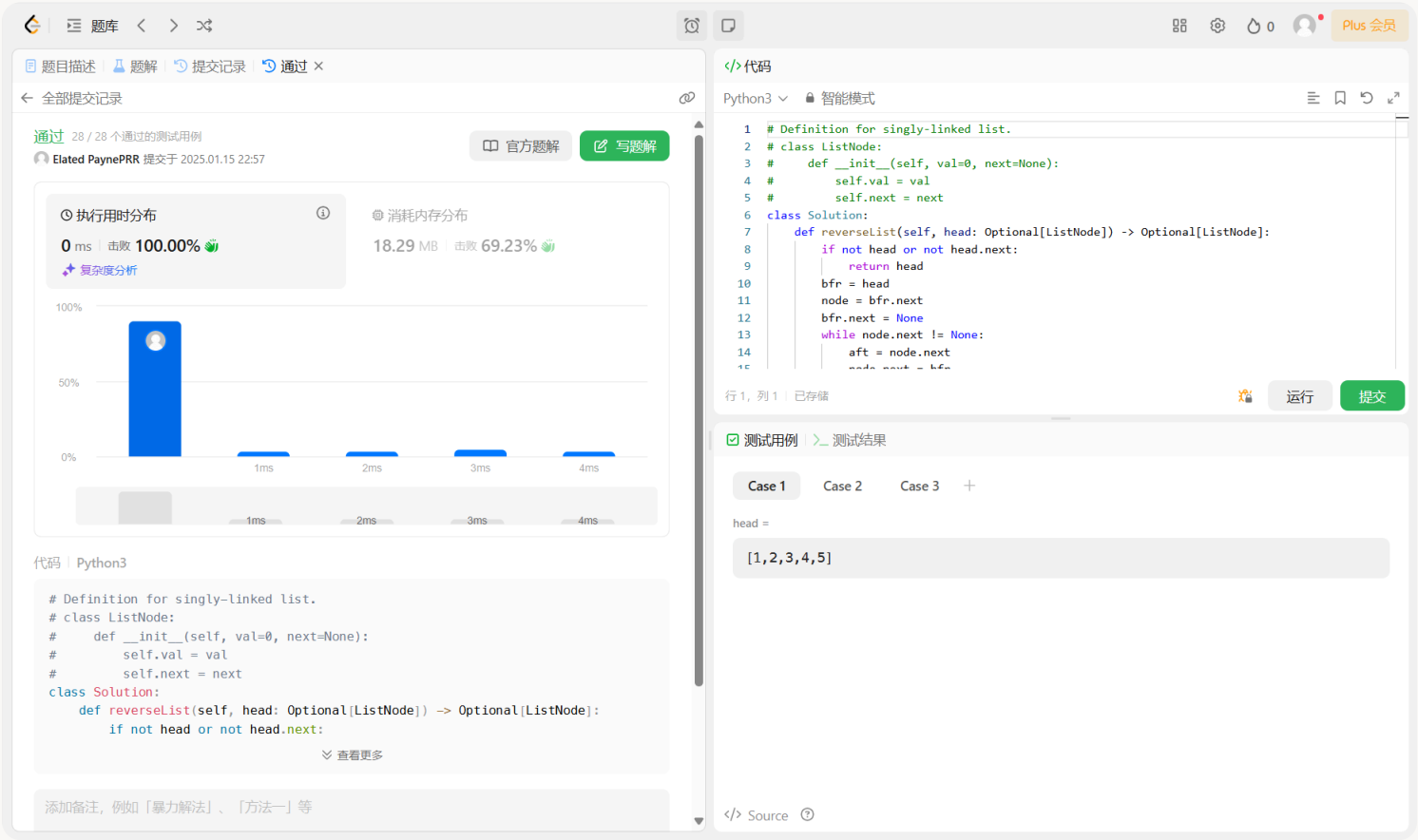
- 寒假写的，记录前后节点就能实现$O(1)$空间复杂度

代码：

Python

```python
# Definition for singly-linked list.
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next
class Solution:
    def reverseList(self, head: Optional[ListNode]) -> Optional[ListNode]:
        if not head or not head.next:
            return head
        bfr = head
        node = bfr.next
        bfr.next = None
        while node.next != None:
            aft = node.next
            node.next = bfr
            bfr = node
            node = aft
        node.next = bfr
        return node
```

代码运行截图（至少包含有"Accepted"）



3478.选出和最大的K个元素

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

思路：

- 最开始还没想到这个题目为什么需要heap
- 处理元素相等的情况时有点棘手

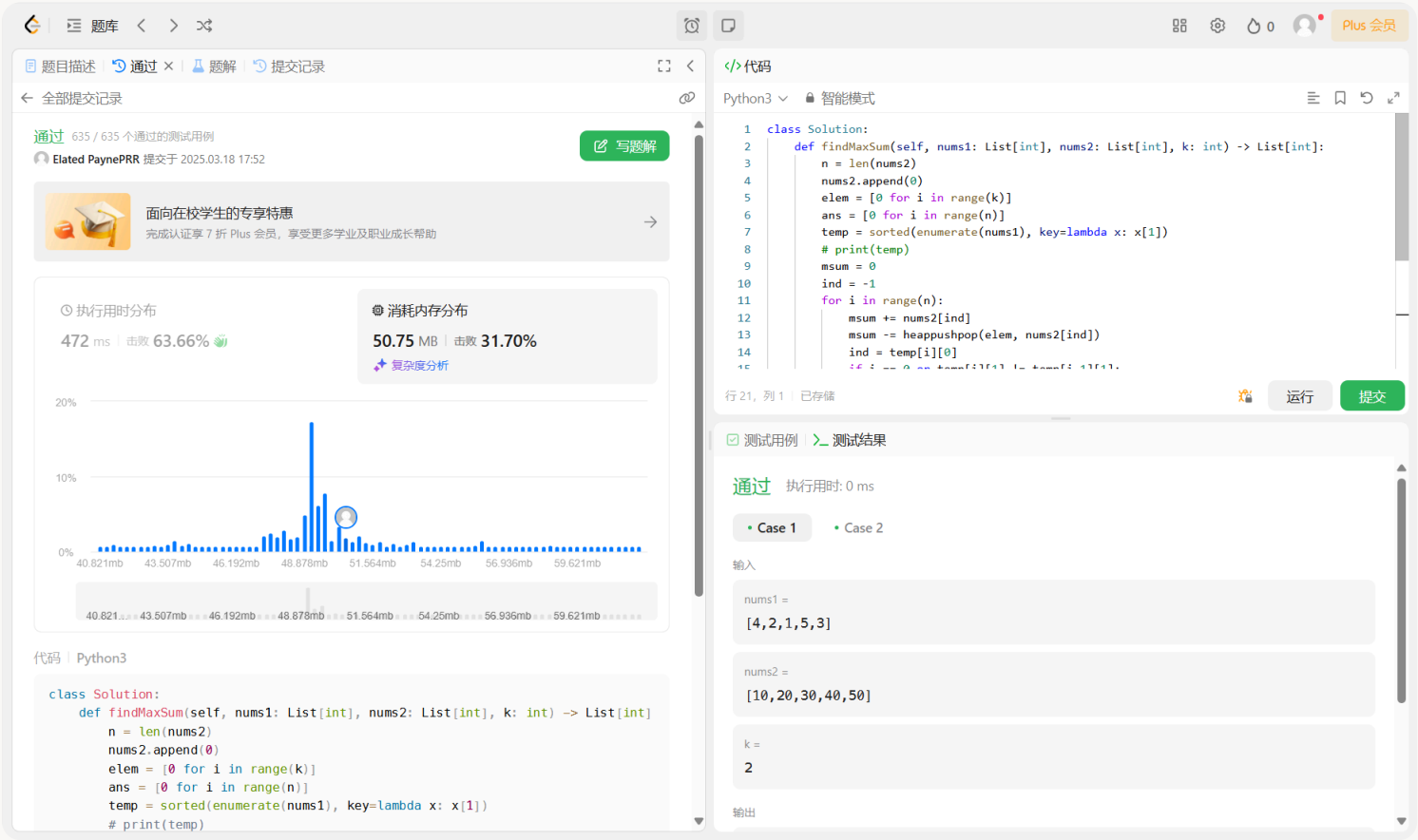
代码：

Python

```python
from heapq import heappushpop
from typing import *
class Solution:
    def findMaxSum(self, nums1: List[int], nums2: List[int], k: int) -> List[int]:
        n = len(nums2)
        nums2.append(0)
        elem = [0 for i in range(k)]
        ans = [0 for i in range(n)]
        temp = sorted(enumerate(nums1), key=lambda x: x[1])
        msum = 0
        ind = -1
        for i in range(n):
            msum += nums2[ind]
            msum -= heappushpop(elem, nums2[ind])
            ind = temp[i][0]
            if i == 0 or temp[i][1] != temp[i-1][1]:
                ans[ind] = msum
            else:
                ans[ind] = ans[temp[i-1][0]]
        return ans

```

代码运行截图（至少包含有"Accepted"）



Q6.交互可视化neural network

https://developers.google.com/machine-learning/crash-course/neural-networks/interactive-exercises

Your task: configure a neural network that can separate the orange dots from the blue dots in the diagram, achieving a loss of less than 0.2 on both the training and test data.

Instructions:

In the interactive widget:

1. Modify the neural network hyperparameters by experimenting with some of the following config settings:
   - Add or remove hidden layers by clicking the **+** and **-** buttons to the left of the **HIDDEN LAYERS** heading in the network diagram.
   - Add or remove neurons from a hidden layer by clicking the **+** and **-** buttons above a hidden-layer column.
   - Change the learning rate by choosing a new value from the **Learning rate** drop-down above the diagram.
   - Change the activation function by choosing a new value from the **Activation** drop-down above the diagram.
2. Click the Play button above the diagram to train the neural network model using the specified parameters.
3. Observe the visualization of the model fitting the data as training progresses, as well as the **Test loss** and **Training loss** values in the **Output** section.
4. If the model does not achieve loss below 0.2 on the test and training data, click reset, and repeat steps 1–3 with a different set of configuration settings. Repeat this process until you achieve the preferred results.

给出满足约束条件的截图，并说明学习到的概念和原理。

- 哎还没写，今天先把作业交了，明天写

2. 学习总结和收获

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。

这几次作业都赶着最后一天写完交，这很不健康，还是多泡泡图书馆提前一点写吧