Assignment #4: 位操作、栈、链表、堆和NN

Updated 1203 GMT+8 Mar 10, 2025

2025 spring, Complied by <mark>同学的姓名、院系</mark>

姓名：李彦臻
学号：2300010821
学院：数学科学学院

> **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/

<mark>请用位操作来实现，并且只使用常量额外空间。</mark>

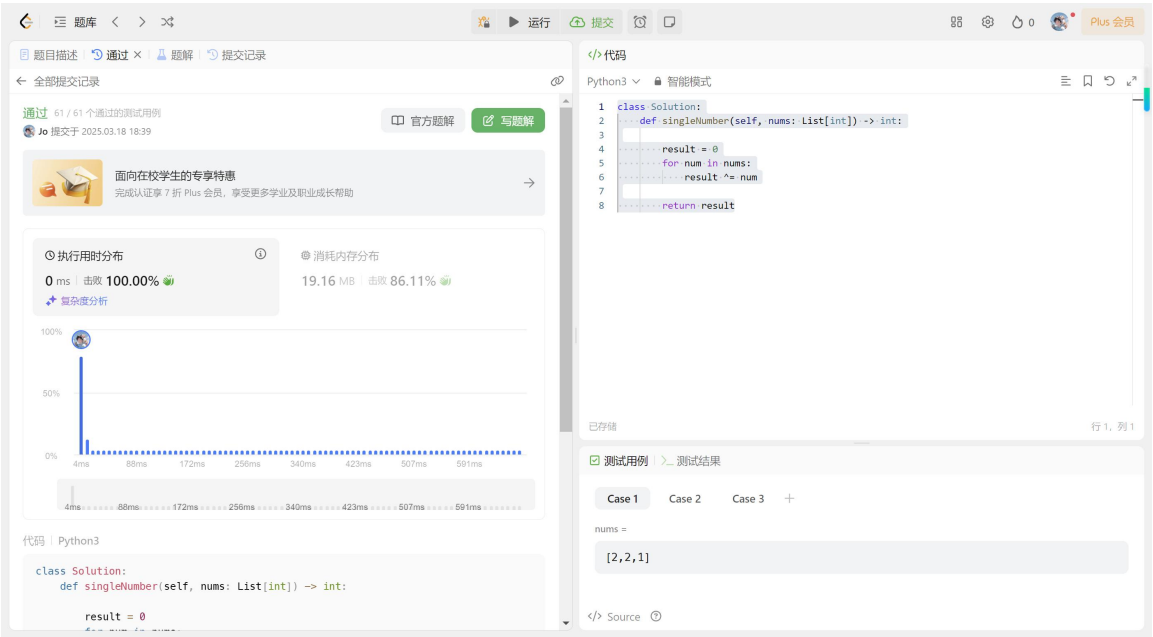
思路：利用异或运算的性质，即a^a=0的特性，把两个相同的数字抵消掉即可（并且因为a^0=a，因此两个数字相互抵消并不会影响后续的操作）
用时：15mins

代码：

```python

```

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

        result = 0
        for num in nums:
            result ^= num

        return result
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



20140:今日化学论文

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

思路：使用栈，对字符挨个儿处理；遇到“[”，就将当前字符串和数字压入栈；遇到“]”，就弹出栈顶的字符串和数字，并进行解压缩即可

用时：40mins

代码：

```python

```
```
def decompress_string(s):
    stack = []
    current_string = ""
    i = 0
    while i < len(s):
        if s[i] == '[':
            # 遇到 '[', 将当前字符串和数字压入栈
            stack.append((current_string, 0))
            current_string = ""
            i += 1
        elif s[i] == ']':
            # 遇到 ']', 弹出栈顶的字符串和数字, 进行解压缩
            prev_string, num = stack.pop()
            current_string = prev_string + current_string * num
            i += 1
        elif s[i].isdigit():
            # 解析数字
            num = 0
            while i < len(s) and s[i].isdigit():
                num = num * 10 + int(s[i])
                i += 1
            # 将数字压入栈
            stack[-1] = (stack[-1][0], num)
        else:
            # 普通字符, 直接加入当前字符串
            current_string += s[i]
            i += 1

    print(current_string)

s = input().strip()
decompress_string(s)
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

160.相交链表

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

思路：用双指针方法，使用两个指针 pA 和 pB，分别从链表 headA 和 headB 的头节点开始遍历；当 pA 到达链表 headA 的末尾时，将其重定位到链表 headB 的头节点；同理，当 pB 到达链表 headB 的末尾时，将其重定位到链表 headA 的头节点。如果两个链表相交，pA 和 pB 会在相交节点相遇；如果不相交，pA 和 pB 会同时到达链表的末尾~
用时：10mins

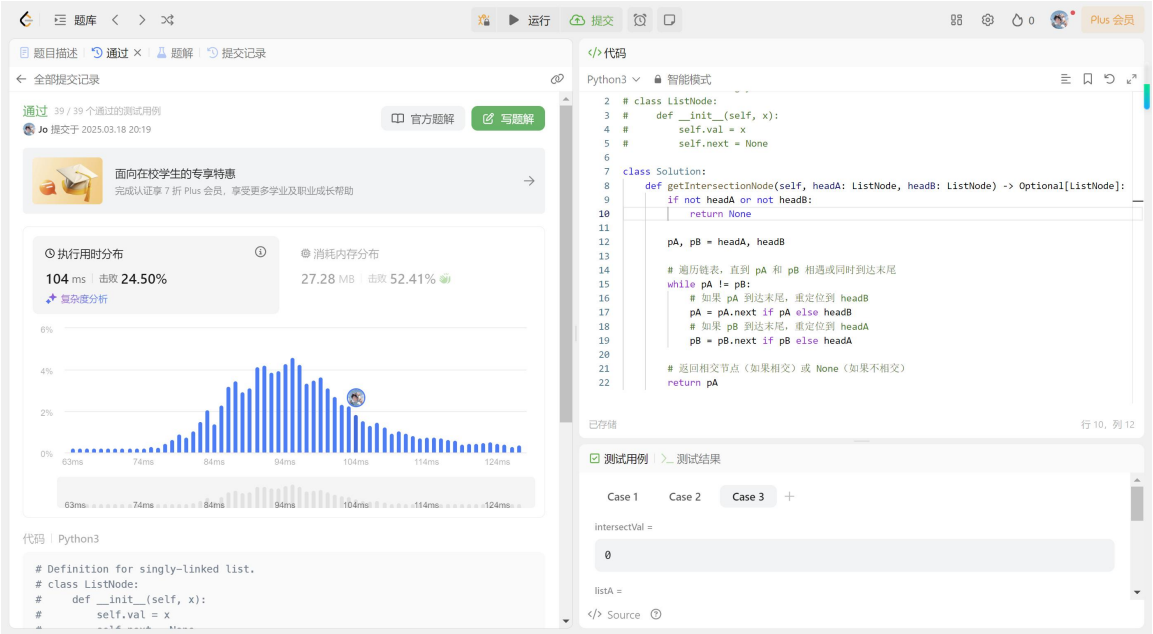
代码：

```python

```

```
class Solution:
    def getIntersectionNode(self, headA: ListNode, headB: ListNode) -> Optional[ListNode]:
        if not headA or not headB:
            return None

        pA, pB = headA, headB
```

```
        # 遍历链表，直到 pA 和 pB 相遇或同时到达末尾
        while pA != pB:
            # 如果 pA 到达末尾，重定位到 headB
            pA = pA.next if pA else headB
            # 如果 pB 到达末尾，重定位到 headA
            pB = pB.next if pB else headA

        # 返回相交节点（如果相交）或 None（如果不相交）
        return pA
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



206.反转链表

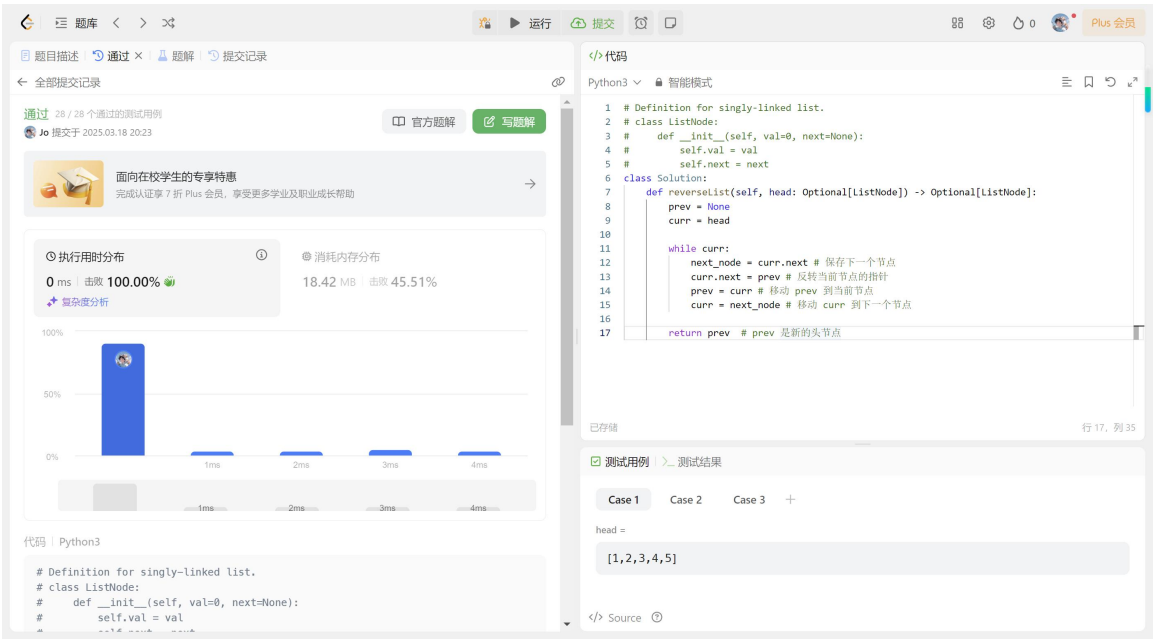
linked list, https://leetcode.cn/problems/reverse-linked-list/

思路：使用迭代法，通过遍历链表，逐个反转节点的指针方向即可
用时：10mins

代码：

```python

```
```
class Solution:
    def reverseList(self, head: Optional[ListNode]) -> Optional[ListNode]:
        prev = None
        curr = head

        while curr:
            next_node = curr.next # 保存下一个节点
            curr.next = prev # 反转当前节点的指针
            prev = curr # 移动 prev 到当前节点
            curr = next_node # 移动 curr 到下一个节点

        return prev  # prev 是新的头节点
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



3478. 选出和最大的 K 个元素

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

思路：

代码：

```python

```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

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<mark>截图</mark>，并说明学习到的概念和原理。

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>

这次作业前四题总体来讲并不算难，但我也从中收获了不少。第一题我一开始做的时候并没有想到题解的思路，后来经过一个同学的提示之后就很快做出来了；通过做这道题我也复习了不少和位操作有关的知识点，蛮有收获。第二题的话因为我对 stack 的性质和使用方法并不是非常熟练，所以做了很久才做出来；但通过仔细琢磨这道题的的嵌套过程我也对 stack 的认知提升了不少；第三题和第四题我个人则认为难度不是很大，只要熟悉了单链表的性质，应该就能比较快做出来。第五题则非常的有难度，我至今还没 ac，仍然在琢磨。。