

### 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。

代码：

```
class Solution(object):
```

```
    def singleNumber(self, nums):
```

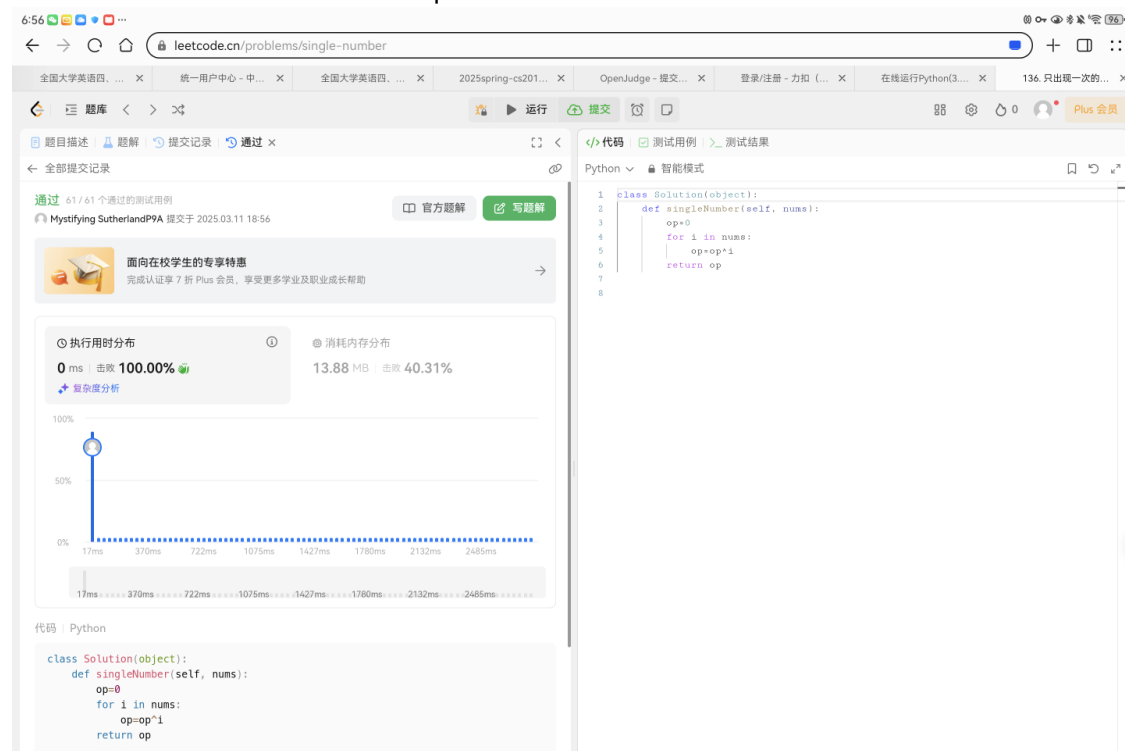
```
        op=0
```

```
        for i in nums:
```

```
            op=op^i
```

```
        return op
```

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：

代码：

```
s=input()
```

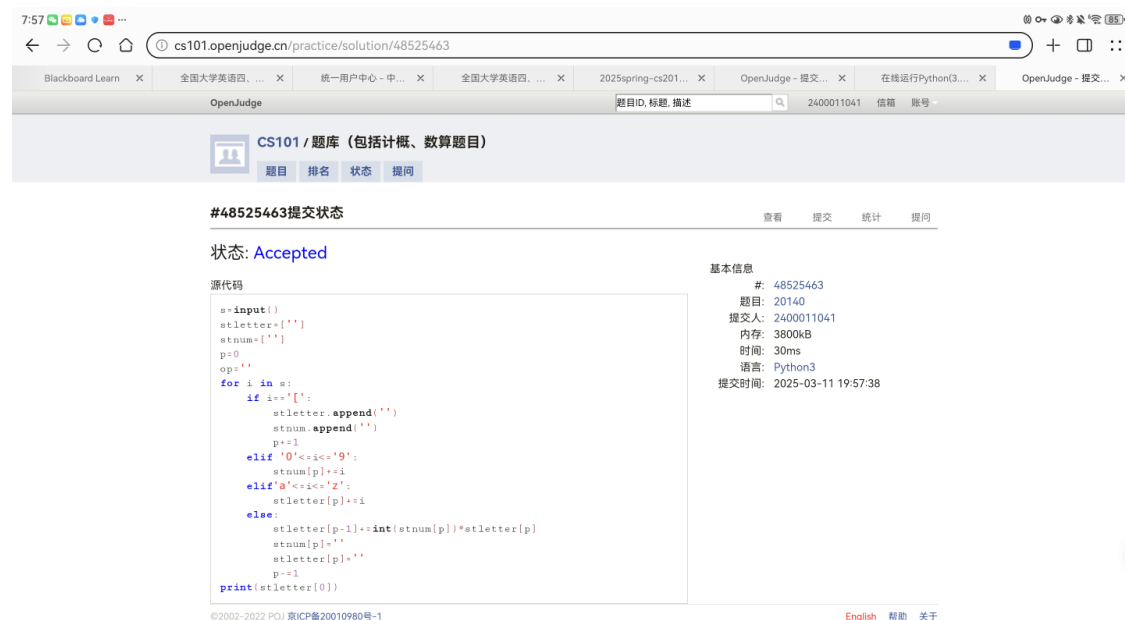
```
stletter=['']
```

```

stnum=[]
p=0
op=""
for i in s:
    if i!=' ':
        stletter.append(i)
        stnum.append(i)
        p+=1
    elif '0'<=i<='9':
        stnum[p]+=i
    elif 'a'<=i<='z':
        stletter[p]+=i
    else:
        stletter[p-1]+=int(stnum[p])*stletter[p]
        stnum[p]="
        stletter[p]="
        p-=1
print(stletter[0])

```

代码运行截图 （至少包含有"Accepted"）



The screenshot shows a web browser window with the URL [cs101.openjudge.cn/practice/solution/48525463](https://cs101.openjudge.cn/practice/solution/48525463). The page title is "CS101 / 题库 (包括计概、数算题目)". The submission status is "Accepted". The source code is as follows:

```

s=input()
stletter=[]
stnum=[]
p=0
op=""
for i in s:
    if i!=' ':
        stletter.append(i)
        stnum.append(i)
        p+=1
    elif '0'<=i<='9':
        stnum[p]+=i
    elif 'a'<=i<='z':
        stletter[p]+=i
    else:
        stletter[p-1]+=int(stnum[p])*stletter[p]
        stnum[p]=''
        stletter[p]=''
        p-=1
print(stletter[0])

```

The submission details on the right are:

- #: 48525463
- 题目: 20140
- 提交人: 2400011041
- 内存: 3800KB
- 时间: 30ms
- 语言: Python3
- 提交时间: 2025-03-11 19:57:38

## 160.相交链表

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

思路:

注意, `op` 是用来存储相交节点的变量。如果将 `op` 初始化为 `0`, 会导致类型不匹配的问题, 因为链表节点是一个对象, 而不是一个整数。具体来说, `op` 应该是一个

ListNode 类型的变量，而不是一个整数。

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):
```

```
        sta, stb = [], []
```

```
        a, b = headA, headB
```

```
        while a:
```

```
            sta.append(a)
```

```
            a = a.next
```

```
        while b:
```

```
            stb.append(b)
```

```
            b = b.next
```

```
        pa, pb = len(sta) - 1, len(stb) - 1
```

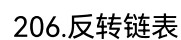
```
        op = None
```

```
        while pa >= 0 and pb >= 0 and sta[pa] is stb[pb]:
```

```
            op = sta[pa]
```

```
            pa, pb = pa - 1, pb - 1
```

代码运行截图（至少包含有"Accepted"）



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

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]
```

```
"""
```

```
pre=None
```

```
p=head
```

```
while p:
```

```
    r=p.next
```

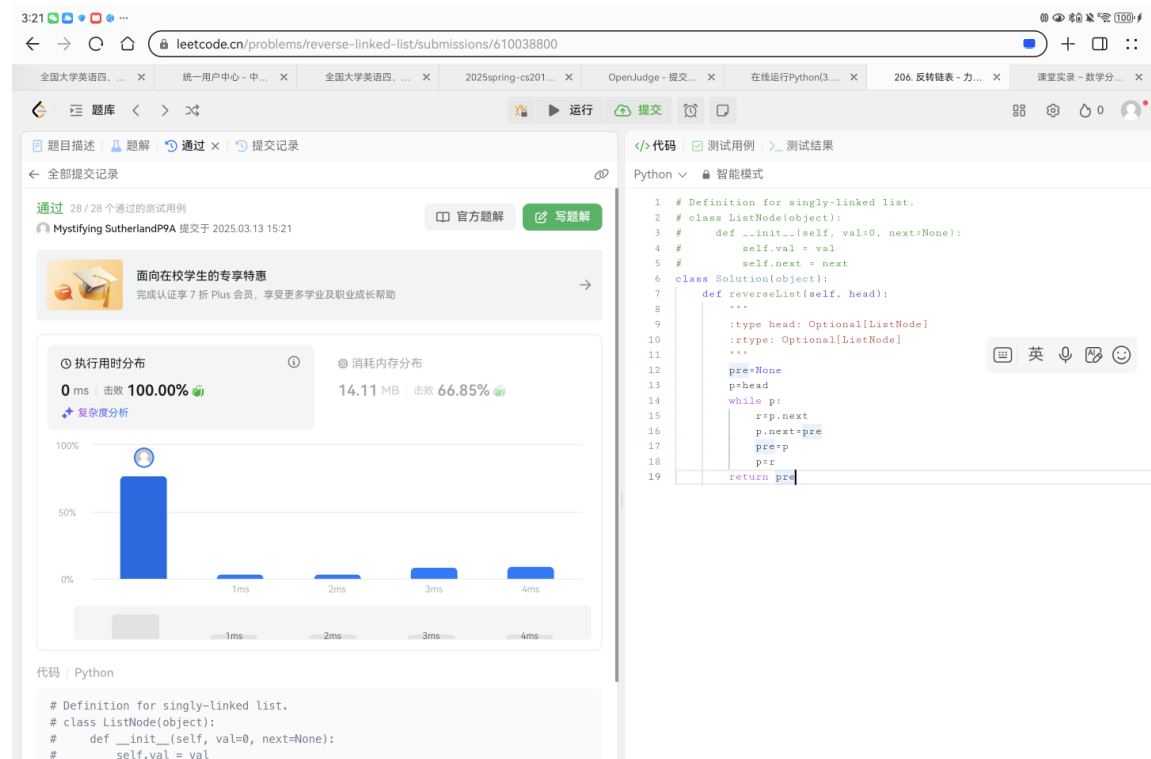
```
    p.next=pre
```

```
    pre=p
```

```
    p=r
```

```
return pre
```

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:

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
```

```
        l=[]
```

```
        n=len(nums1)
```

```
        for i in range(n):
```

```
            l.append((nums1[i],nums2[i],i))
```

```
        op=[0]*n
```

```
        heapq.heapify(l)
```

```
        maxl=[]
```

```
        s=0
```

```
        prenum1,prei=-99999999,-1
```

```
        for t in range(n):
```

```
            num1,num2,i=heapq.heappop(l)
```

```
            op[i]=s
```

```
            if len(maxl)<k:
```

```
heapq.heappush(maxl,num2)
```

```
s+=num2
```

```
else:
```

```
m=heapq.heappop(maxl)
```

```
if m<num2:
```

```
heapq.heappush(maxl,num2)
```

```
s=s-m+num2
```

```
else:
```

```
heapq.heappush(maxl,m)
```

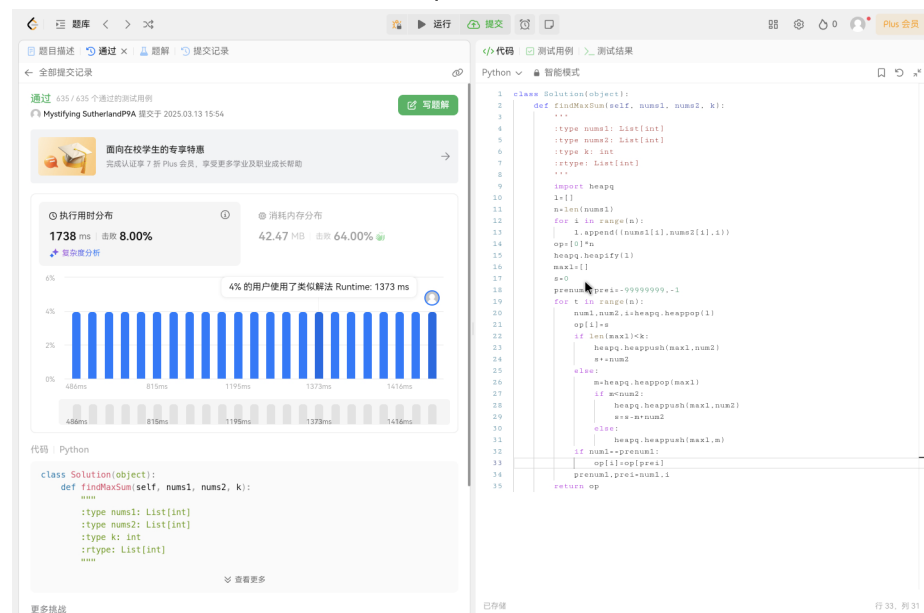
```
if num1==prenum1:
```

```
op[i]=op[prei]
```

```
prenum1,prei=num1,i
```

```
return op
```

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/inter>

## active-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:

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.

Click the Play button above the diagram to train the neural network model using the specified parameters.

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.

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。

## 2. 学习总结和收获

本次作业大部分是 leetcode 上的题目，借此熟悉了 ooc 的语法

题目难度适中

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。

日常跟进 OJ 每日选做