1. 题目
LC108.将有序数组转换为二叉树
dfs, https://leetcode.cn/problems/convert-sorted-array-to-binary-search-tree/

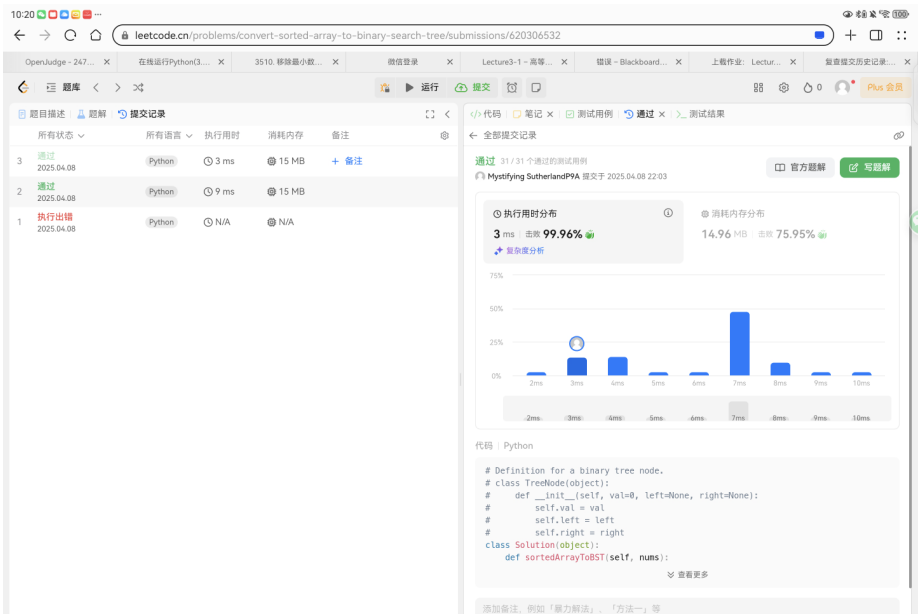
思路:

代码:

```
# Definition for a binary tree node.

# class TreeNode(object):

#         def __init__(self, val=0, left=None, right=None):

#                 self.val = val

#                 self.left = left

#                 self.right = right

class Solution(object):

        def sortedArrayToBST(self, nums):

                """

                :type nums: List[int]

                :rtype: Optional[TreeNode]

                """

                if len(nums)==1:

                        return TreeNode(nums[0])

                elif len(nums)==0:

                        return None

                mid=(len(nums)-1)//2

                root=TreeNode(nums[mid])

                left=nums[:mid]
```

```
            right=nums[mid+1:]

            root.left=self.sortedArrayToBST(left)

            root.right=self.sortedArrayToBST(right)

            return root
```

代码运行截图 （至少包含有"Accepted"）



M27928:遍历树

adjacency list, dfs, http://cs101.openjudge.cn/practice/27928/

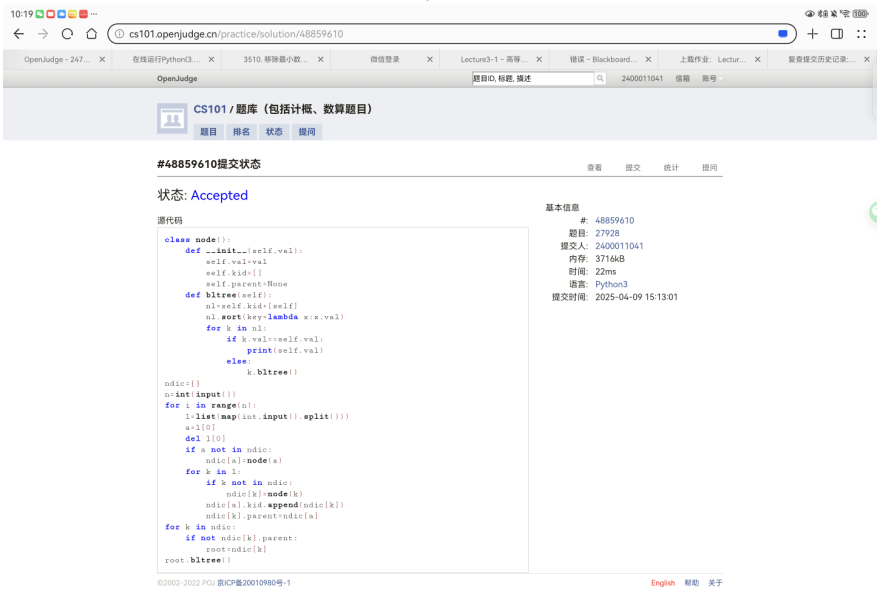
思路：

代码：

```
class node():
    def __init__(self,val):
        self.val=val
        self.kid=[]
        self.parent=None
    def bltree(self):
        nl=self.kid+[self]
        nl.sort(key=lambda x:x.val)
        for k in nl:
            if k.val==self.val:
                print(self.val)
            else:
```

```
                    k.bltree()
ndic={}
n=int(input())
for i in range(n):
    l=list(map(int,input().split()))
    a=l[0]
    del l[0]
    if a not in ndic:
        ndic[a]=node(a)
    for k in l:
        if k not in ndic:
            ndic[k]=node(k)
        ndic[a].kid.append(ndic[k])
        ndic[k].parent=ndic[a]
for k in ndic:
    if not ndic[k].parent:
        root=ndic[k]
root.bltree()
```

代码运行截图 （至少包含有"Accepted"）



LC129.求根节点到叶节点数字之和

dfs, https://leetcode.cn/problems/sum-root-to-leaf-numbers/

思路：

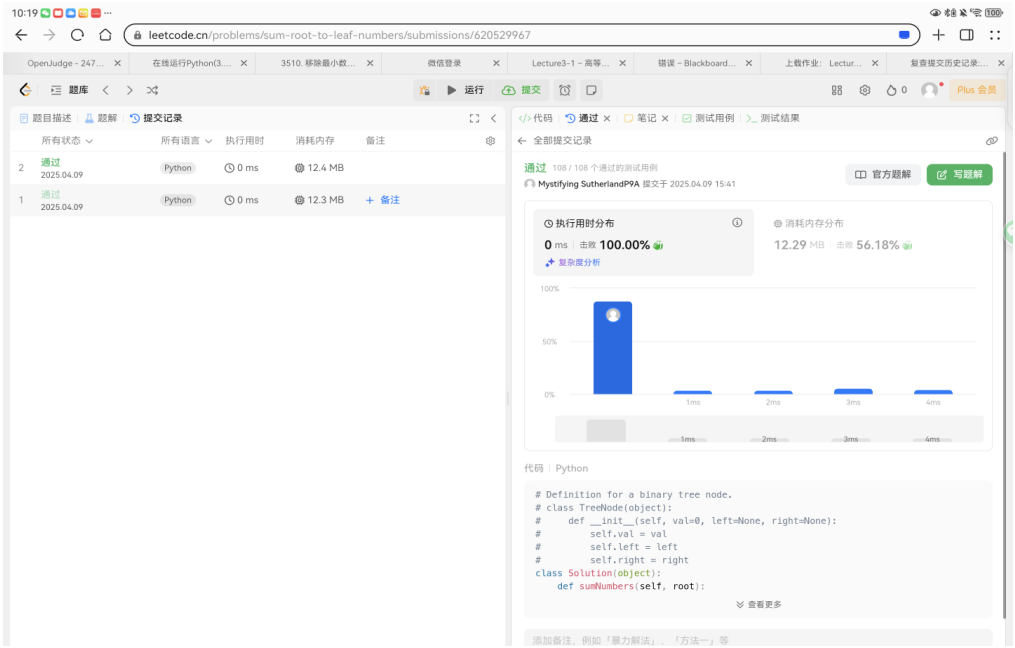
代码：

```
# Definition for a binary tree node.

# class TreeNode(object):

#     def __init__(self, val=0, left=None, right=None):
```

```
#                 self.val = val
#                 self.left = left
#                 self.right = right
class Solution(object):
        def sumNumbers(self, root):
                """
                :type root: Optional[TreeNode]
                :rtype: int
                """
                global rt
                rt=0
                def dfs(root,s):
                        global rt
                        s=s+str(root.val)
                        if not root.left and not root.right:
                                rt+=int(s)
                        else:
                                if root.left:
                                        dfs(root.left,s)
                                if root.right:
                                        dfs(root.right,s)
                dfs(root,'')
```

```
        return rt
```

代码运行截图 （至少包含有"Accepted"）



M22158:根据二叉树前中序序列建树

tree, http://cs101.openjudge.cn/practice/22158/

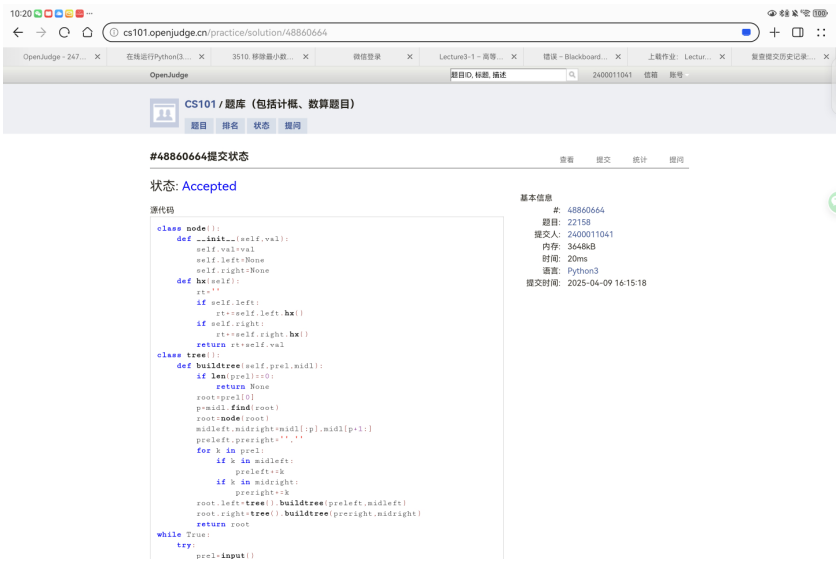
思路：

代码：

```
class node():
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
    def hx(self):
        rt=''
        if self.left:
            rt+=self.left.hx()
        if self.right:
            rt+=self.right.hx()
        return rt+self.val
class tree():
    def buildtree(self,prel,midl):
        if len(prel)==0:
            return None
        root=prel[0]
        p=midl.find(root)
        root=node(root)
```

```
            midleft,midright=midl[:p],midl[p+1:]
            preleft,preright='',''
            for k in prel:
                if k in midleft:
                    preleft+=k
                if k in midright:
                    preright+=k
            root.left=tree().buildtree(preleft,midleft)
            root.right=tree().buildtree(preright,midright)
            return root
while True:
    try:
        prel=input()
        midl=input()
        root=tree().buildtree(prel,midl)
        print(root.hx())
    except EOFError:
        break
```

代码运行截图 （至少包含有"Accepted"）



M24729:括号嵌套树

dfs, stack, http://cs101.openjudge.cn/practice/24729/

思路：

代码：

```
class node():
    def __init__(self):
        self.val=None
        self.kid=[]
        self.parent=None
```

```
def qx(root):
    rt=root.val
    for k in root.kid:
        rt+=qx(k)
    return rt
def hx(root):
    rt=''
    for k in root.kid:
        rt+=hx(k)
    rt+=root.val
    return rt
l=input()
root=node()
root.val=l[0]
cur=root
for k in l[1:]:
    if k=='(':
        new=node()
        new.parent=cur
        cur.kid.append(new)
        cur=new
    elif k==',':
        new=node()
        new.parent=cur.parent
        cur.parent.kid.append(new)
        cur=new
    elif k==')':
        cur=cur.parent
    else:
        cur.val=k
print(qx(root))
print(hx(root))
```

代码运行截图 （至少包含有"Accepted"）

LC3510.移除最小数对使数组有序 II

doubly-linked list + heap,

https://leetcode.cn/problems/minimum-pair-removal-to-sort-array-ii/

思路:

代码:

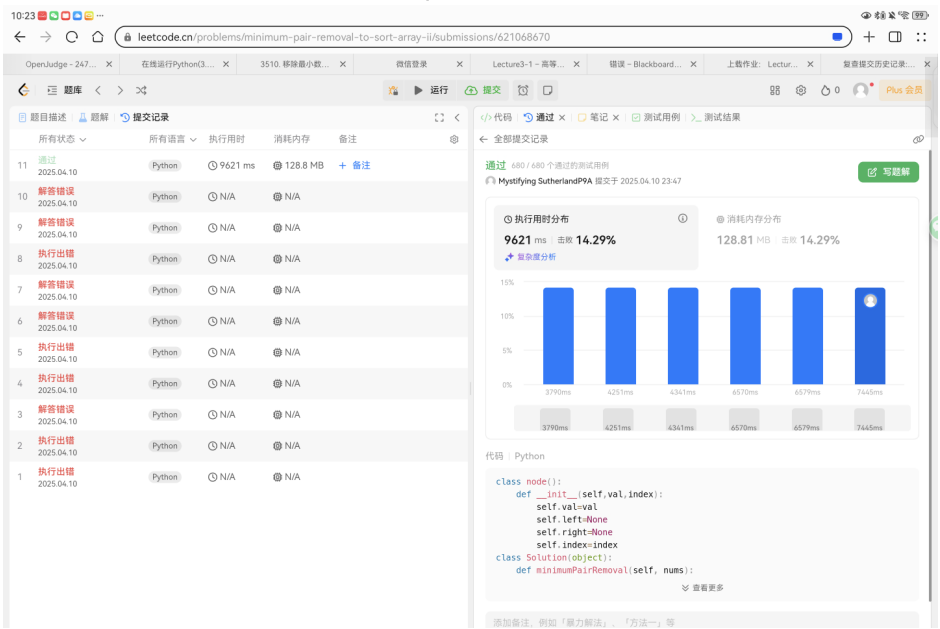
```
class node():

    def __init__(self,val,index):

        self.val=val

        self.left=None

        self.right=None

        self.index=index

class Solution(object):

    def minimumPairRemoval(self, nums):

        """

        :type nums: List[int]

        :rtype: int

        """
```

```
import heapq
nodedic={}
heap=[]
nodel=[]
ct=0
op=0
for i in range(len(nums)-1):
    if nums[i]>nums[i+1]:
        ct+=1
    heapq.heappush(heap,(nums[i]+nums[i+1],i))
    nodedic[(nums[i]+nums[i+1],i)]=1
    nodel.append(node(nums[i],i))
nodel.append(node(nums[len(nums)-1],len(nums)-1))
for k in range(len(nums)-1):
    nodel[k].right=nodel[k+1]
    nodel[k+1].left=nodel[k]
while ct!=0:
    s,p=heapq.heappop(heap)
    if nodedic[(s,p)]==0:
        continue
    new=nodel[p]
    pre=new.left
```

```
nxt=new.right

nn=nxt.right

op+=1



if nxt and new.val>nxt.val:

    ct-=1

if nn and nn.val<nxt.val:

    ct-=1

if nn and nn.val<s:

    ct+=1

if pre and pre.val>new.val:

    ct-=1

if pre and pre.val>s:

    ct+=1



nodedic[(s,p)]=0

if nn:

    nn.left=new

    nodedic[(nn.val+nxt.val,nxt.index)]=0

    heapq.heappush(heap,(s+nn.val,new.index))

    nodedic[(s+nn.val,new.index)]=1

if pre:
```

```
                nodedic[(pre.val+new.val,pre.index)]=0

                heapq.heappush(heap,(pre.val+s,pre.index))

                nodedic[(s+pre.val,pre.index)]=1



            new.val=s

            new.right=nn



        return op
```

代码运行截图 （至少包含有"Accepted"）



2. 学习总结和收获

本次作业让我更加熟悉了类的使用和树的操作。最后一题确实难，想到思路后仍然有细节问题容易 WA（如添加与删除的先后）。

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。

仍然期中周，暂停选做