Assignment #7: April 月考

Updated 1557 GMT+8 Apr 3, 2024

2024 spring, Complied by 同学的姓名、院系

说明：

1）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

2）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。

3）如果不能在截止前提交作业，请写明原因。

编程环境

（请改为同学的操作系统、编程环境等）

操作系统：macOS Ventura 13.4.1 (c)

Python编程环境：Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境：Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

1. 题目

27706: 逐词倒放

http://cs101.openjudge.cn/practice/27706/

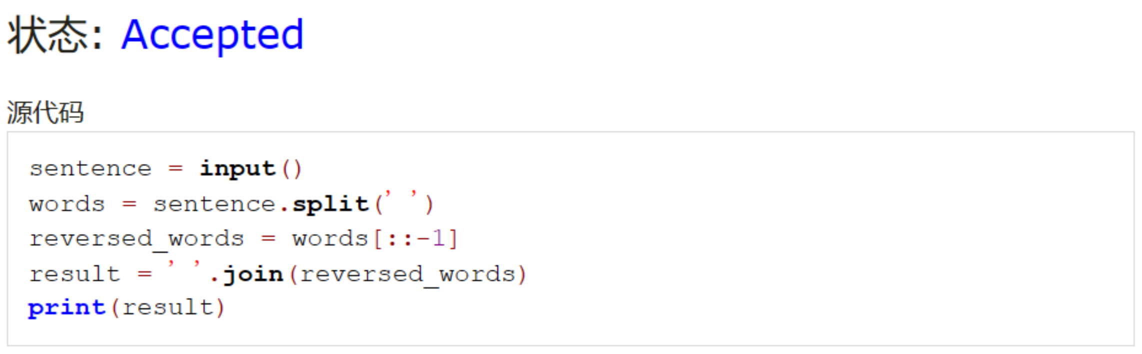
思路：

字符串简单题

代码

```python
# 
sentence = input()
words = sentence.split(' ')
reversed_words = words[::-1]
result = ' '.join(reversed_words)
print(result)
```

代码运行截图 （至少包含有"Accepted"）


状态: Accepted

源代码

```python
sentence = input()
words = sentence.split(' ')
reversed_words = words[::-1]
result = ' '.join(reversed_words)
print(result)
```

27951: 机器翻译

http://cs101.openjudge.cn/practice/27951/

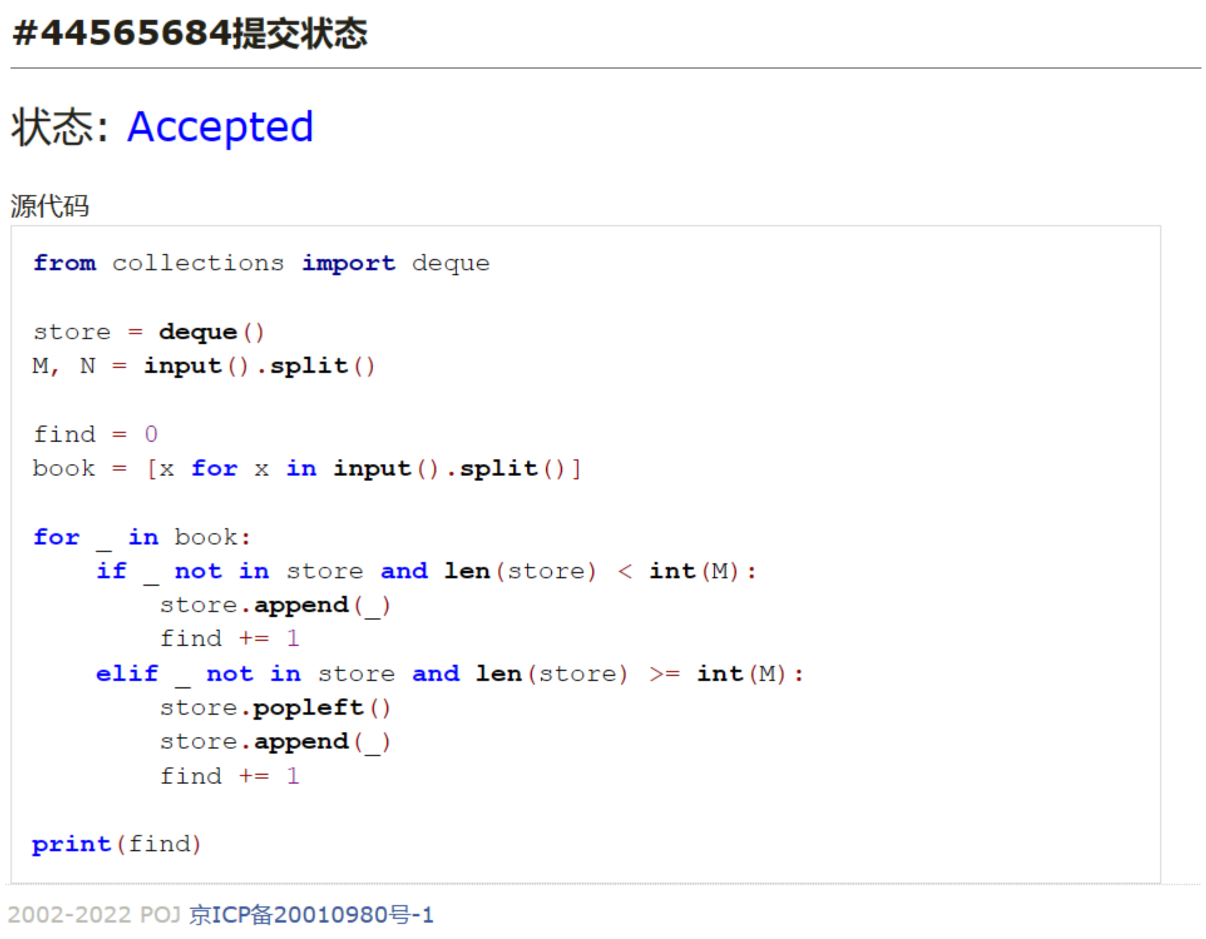
思路:

很简单的双端队列

代码

```python
# from collections import deque

store = deque()
M, N = input().split()

find = 0
book = [x for x in input().split()]

for _ in book:
    if _ not in store and len(store) < int(M):
        store.append(_)
        find += 1
    elif _ not in store and len(store) >= int(M):
        store.popleft()
        store.append(_)
        find += 1

print(find)
```

代码运行截图 (至少包含有"Accepted")



#44565684提交状态

状态: Accepted

源代码

```
from collections import deque

store = deque()
M, N = input().split()

find = 0
book = [x for x in input().split()]

for _ in book:
    if _ not in store and len(store) < int(M):
        store.append(_)
        find += 1
    elif _ not in store and len(store) >= int(M):
        store.popleft()
        store.append(_)
        find += 1

print(find)
```

2002-2022 POJ 京ICP备20010980号-1

27932: Less or Equal

http://cs101.openjudge.cn/practice/27932/

思路:

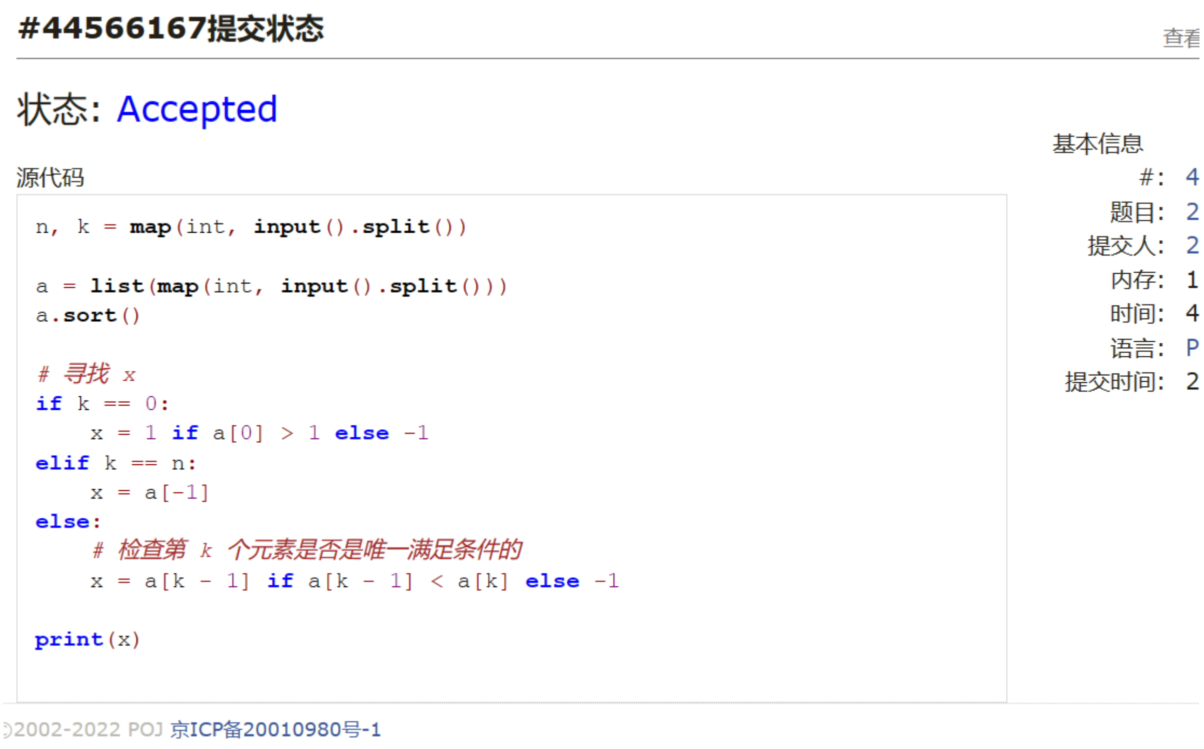
数学题。用自带的sort()的时间复杂度为nlogn

代码

```
#
n, k = map(int, input().split())

a = list(map(int, input().split()))
a.sort()

# 寻找 x
if k == 0:
    x = 1 if a[0] > 1 else -1
elif k == n:
    x = a[-1]
else:
    # 检查第 k 个元素是否是唯一满足条件的
    x = a[k - 1] if a[k - 1] < a[k] else -1
```

```
print(x)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



#44566167提交状态 查看

状态: Accepted

源代码

```
n, k = map(int, input().split())

a = list(map(int, input().split()))
a.sort()

# 寻找 x
if k == 0:
    x = 1 if a[0] > 1 else -1
elif k == n:
    x = a[-1]
else:
    # 检查第 k 个元素是否是唯一满足条件的
    x = a[k - 1] if a[k - 1] < a[k] else -1

print(x)
```

基本信息

#:	4
题目:	2
提交人:	2
内存:	1
时间:	4
语言:	P
提交时间:	2

©2002-2022 POJ 京ICP备20010980号-1

27948: FBI树

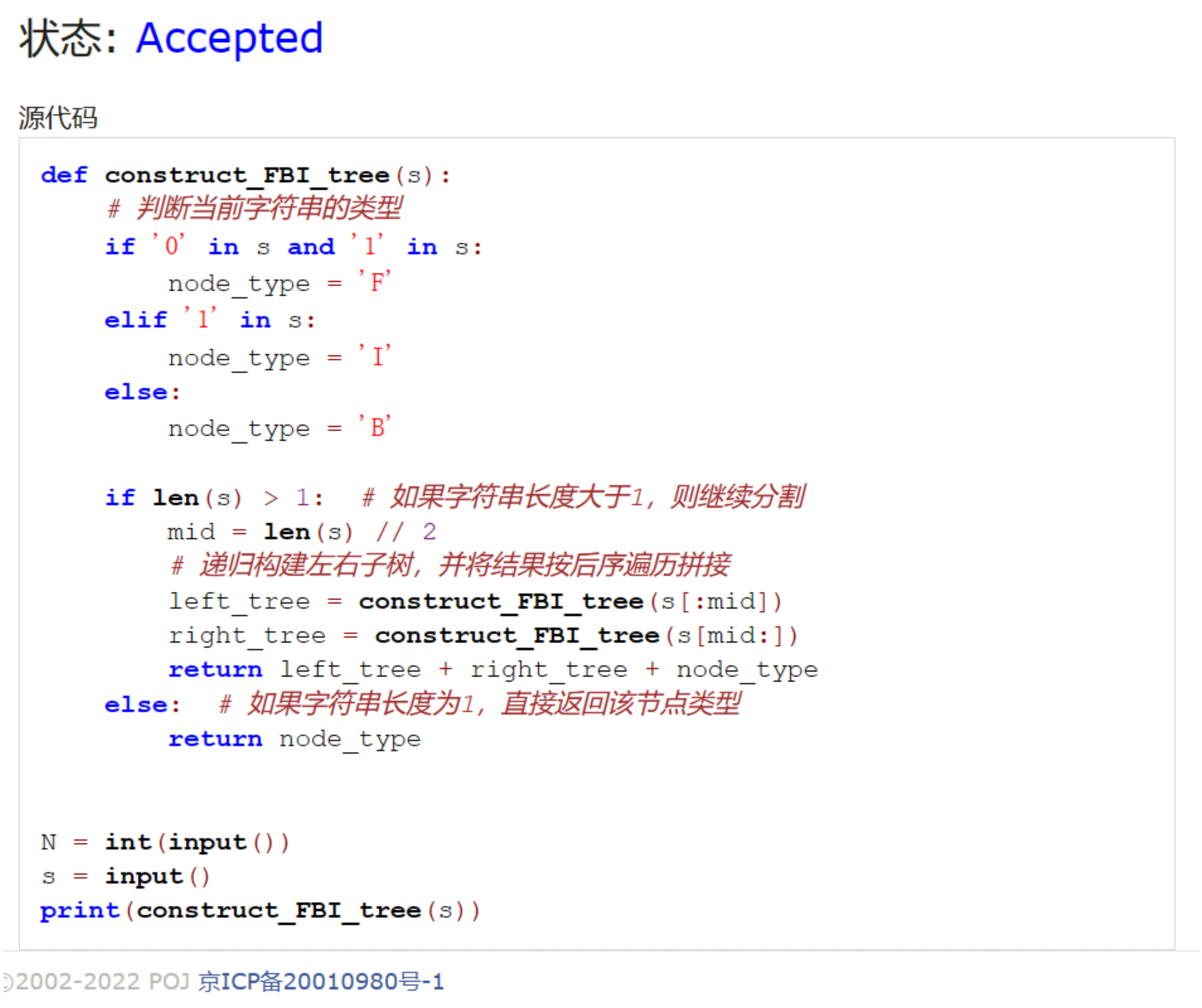
http://cs101.openjudge.cn/practice/27948/

思路：利用分治思想完成，最开始没想到 in 判断，尝试先分支，利用n确定树的结构，然后自下而上的生成树。感觉上没啥问题，但是不好写树。作罢。

代码

```
#
def construct_FBI_tree(s):
    # 判断当前字符串的类型
    if '0' in s and '1' in s:
        node_type = 'F'
    elif '1' in s:
        node_type = 'I'
    else:
        node_type = 'B'

    if len(s) > 1:  # 如果字符串长度大于1，则继续分割
        mid = len(s) // 2
        # 递归构建左右子树，并将结果按后序遍历拼接
        left_tree = construct_FBI_tree(s[:mid])
        right_tree = construct_FBI_tree(s[mid:])
        return left_tree + right_tree + node_type
    else:  # 如果字符串长度为1，直接返回该节点类型
        return node_type


N = int(input())
s = input()
print(construct_FBI_tree(s))
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```
def construct_FBI_tree(s):
    # 判断当前字符串的类型
    if '0' in s and '1' in s:
        node_type = 'F'
    elif '1' in s:
        node_type = 'I'
    else:
        node_type = 'B'

    if len(s) > 1:  # 如果字符串长度大于1，则继续分割
        mid = len(s) // 2
        # 递归构建左右子树，并将结果按后序遍历拼接
        left_tree = construct_FBI_tree(s[:mid])
        right_tree = construct_FBI_tree(s[mid:])
        return left_tree + right_tree + node_type
    else:  # 如果字符串长度为1，直接返回该节点类型
        return node_type


N = int(input())
s = input()
print(construct_FBI_tree(s))
```

©2002-2022 POJ 京ICP备20010980号-1

27925: 小组队列

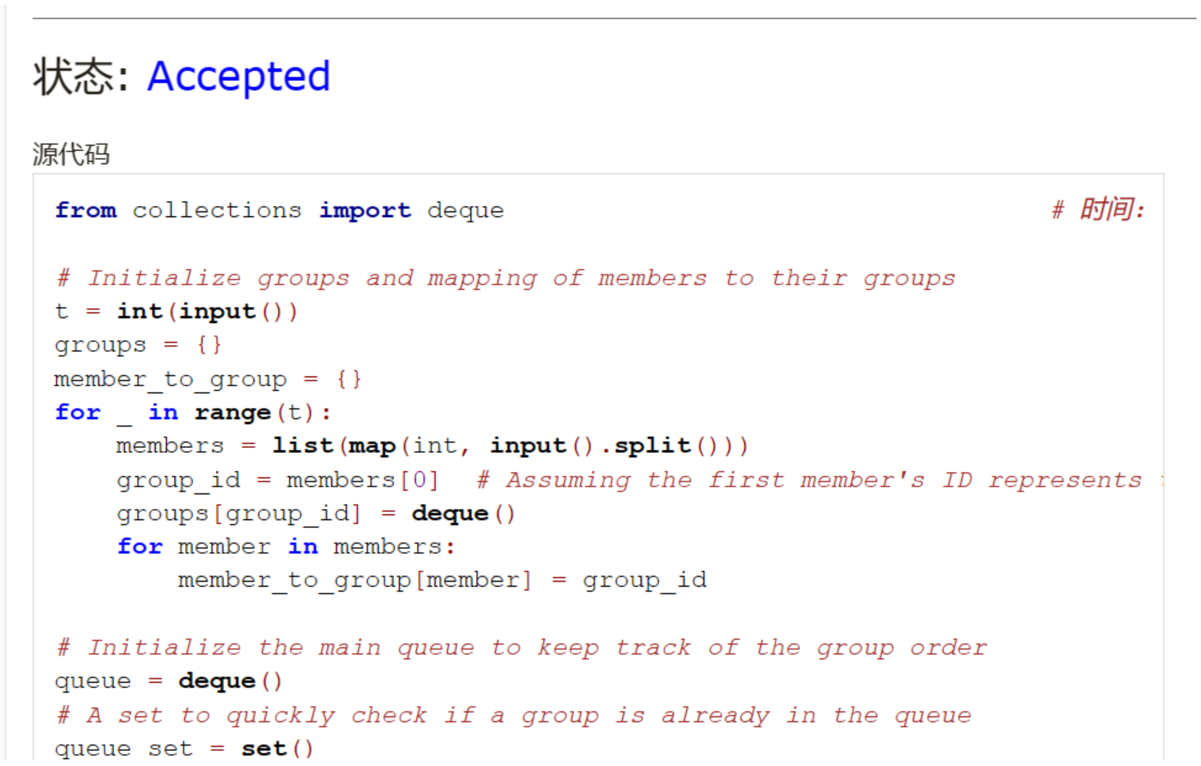
http://cs101.openjudge.cn/practice/27925/

思路：双端队列典型题目。

代码

```python
#
from collections import deque                    # 时间: 105ms

# Initialize groups and mapping of members to their groups
t = int(input())
groups = {}
member_to_group = {}
for _ in range(t):
    members = list(map(int, input().split()))
    group_id = members[0]  # Assuming the first member's ID represents the group
ID
    groups[group_id] = deque()
    for member in members:
        member_to_group[member] = group_id

# Initialize the main queue to keep track of the group order
queue = deque()
# A set to quickly check if a group is already in the queue
queue_set = set()


while True:
    command = input().split()
    if command[0] == 'STOP':
        break
    elif command[0] == 'ENQUEUE':
        x = int(command[1])
        group = member_to_group.get(x, None)
        # Create a new group if it's a new member not in the initial list
        if group is None:
            group = x
            groups[group] = deque([x])
            member_to_group[x] = group
        else:
            groups[group].append(x)
        if group not in queue_set:
            queue.append(group)
            queue_set.add(group)
    elif command[0] == 'DEQUEUE':
        if queue:
            group = queue[0]
            x = groups[group].popleft()
            print(x)
            if not groups[group]:  # If the group's queue is empty, remove it
from the main queue
                queue.popleft()
                queue_set.remove(group)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```
from collections import deque                                          # 时间:

# Initialize groups and mapping of members to their groups
t = int(input())
groups = {}
member_to_group = {}
for _ in range(t):
    members = list(map(int, input().split()))
    group_id = members[0]  # Assuming the first member's ID represents
    groups[group_id] = deque()
    for member in members:
        member_to_group[member] = group_id

# Initialize the main queue to keep track of the group order
queue = deque()
# A set to quickly check if a group is already in the queue
queue_set = set()
```

27928: 遍历树

http://cs101.openjudge.cn/practice/27928/

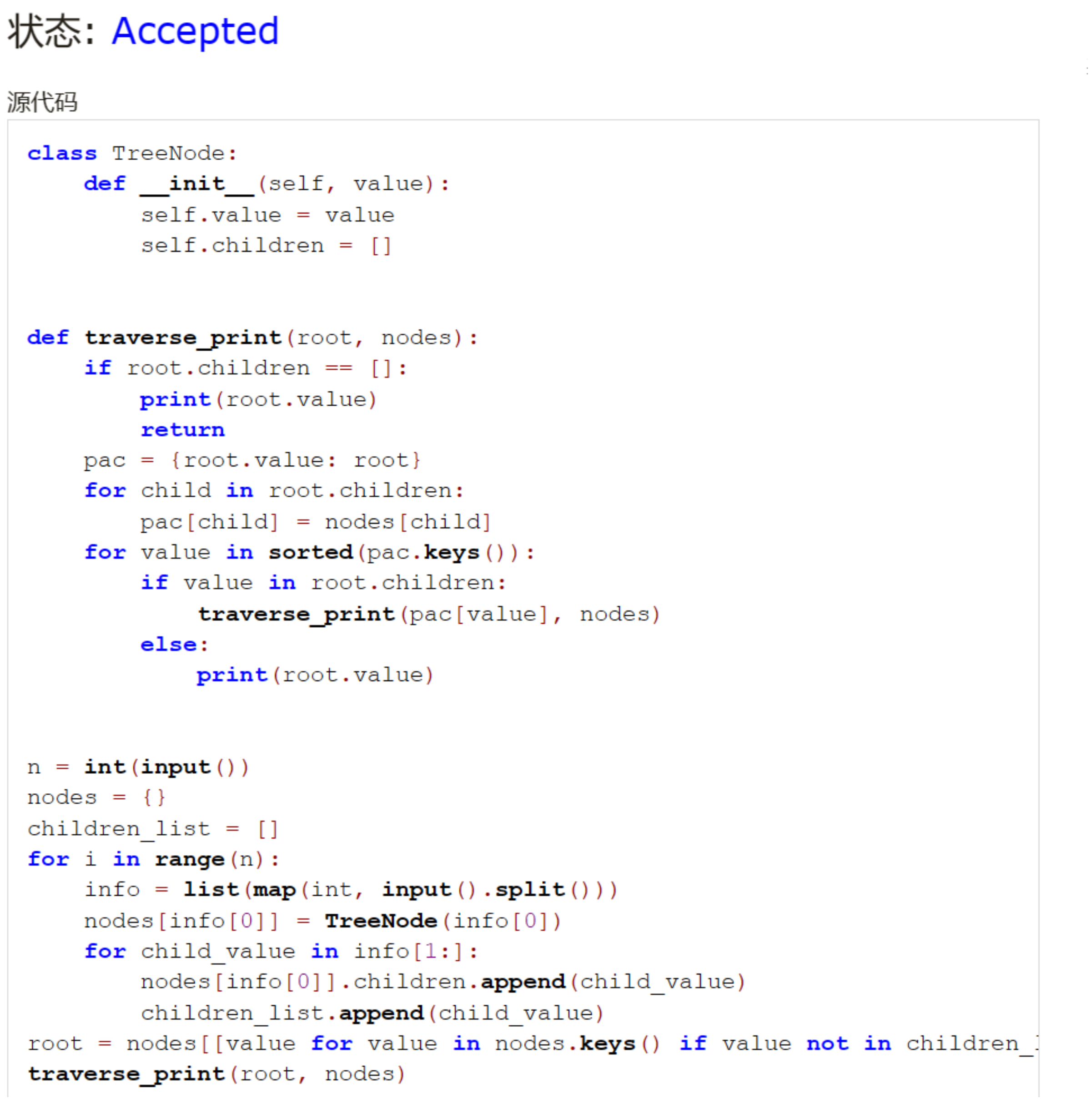
思路：不是很会这道题的理解，也没看到同学的方法。用可视化看看。

代码

```
#
class TreeNode:
    def __init__(self, value):
        self.value = value
        self.children = []


def traverse_print(root, nodes):
    if root.children == []:
        print(root.value)
        return
    pac = {root.value: root}
    for child in root.children:
        pac[child] = nodes[child]
    for value in sorted(pac.keys()):
        if value in root.children:
            traverse_print(pac[value], nodes)
        else:
            print(root.value)


n = int(input())
```

```python
nodes = {}
children_list = []
for i in range(n):
    info = list(map(int, input().split()))
    nodes[info[0]] = TreeNode(info[0])
    for child_value in info[1:]:
        nodes[info[0]].children.append(child_value)
        children_list.append(child_value)
root = nodes[[value for value in nodes.keys() if value not in children_list][0]]
traverse_print(root, nodes)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）


状态: Accepted

源代码

```python
class TreeNode:
    def __init__(self, value):
        self.value = value
        self.children = []


def traverse_print(root, nodes):
    if root.children == []:
        print(root.value)
        return
    pac = {root.value: root}
    for child in root.children:
        pac[child] = nodes[child]
    for value in sorted(pac.keys()):
        if value in root.children:
            traverse_print(pac[value], nodes)
        else:
            print(root.value)


n = int(input())
nodes = {}
children_list = []
for i in range(n):
    info = list(map(int, input().split()))
    nodes[info[0]] = TreeNode(info[0])
    for child_value in info[1:]:
        nodes[info[0]].children.append(child_value)
        children_list.append(child_value)
root = nodes[[value for value in nodes.keys() if value not in children_
traverse_print(root, nodes)
```

2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。

这周考试题目难度适中，能够做出来5题左右，但是还应该提升速度和熟练度。特别是一些常规题目的应用，如果期末考试出现的双端队列和排序题目，我们都应该快速完成，我打算在4月下旬突击一波书本概念，然后在复习一次排序算法和双端队列，5月开始吭硬骨头树图。加油。