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：

逐词倒放

代码

```
print(' '.join(reversed(input().split())))
```

代码运行截图 ==（至少包含有"Accepted"）==

27951: 机器翻译

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

思路：

模拟，缓存内存中的数据。

代码

```
cache = []
m, n = map(int, input().split())
res = 0
for now in map(int, input().split()):
    if not now in cache:
        cache.append(now)
        res += 1
        if len(cache) > m:
            cache.pop(0)
print(res)
```

代码运行截图 ==（至少包含有"Accepted"）==

27932: Less or Equal

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

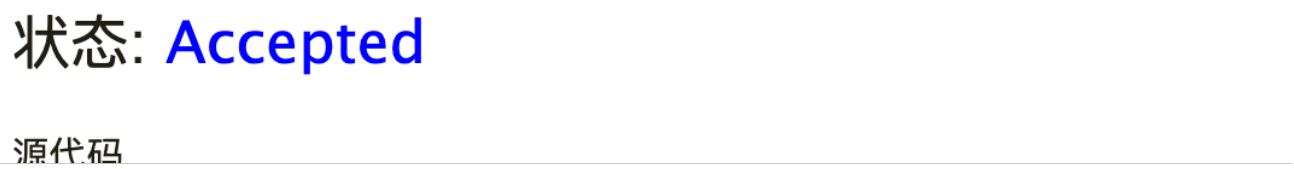
思路：

注意考虑 k = 0 的边界条件

代码

```
n, k = map(int, input().split())
l = sorted([int(x) for x in input().split()])
if k == 0:
    if l[0] > 1:
        print(1)
    else:
        print(-1)
else:
    if k < n and l[k - 1] == l[k]:
        print(-1)
    else:
        print(l[k - 1])
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



27948: FBI树

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

思路：

递归遍历。

代码

```
def find_type(s):
    zero = '0' in s
    one = '1' in s
    if zero and one:
        return 'F'
    if zero:
        return 'B'
    if one:
        return 'I'

class Node:
    def __init__(self, type, lchild=None, rchild=None):
        self.lchild = lchild
        self.rchild = rchild
        self.type = type
    def traverse(self):
        l = self.lchild.traverse() if self.lchild != None else ''
        r = self.rchild.traverse() if self.rchild != None else ''
        return l + r + self.type

def tree_parser(s, depth):
    l = len(s)
    if depth < 1:
        return Node(find_type(s))
    return Node(find_type(s), tree_parser(s[:l // 2], depth - 1), tree_parser(s[l // 2:],
depth - 1))

n = int(input())
string = input()
print(tree_parser(string, n).traverse())
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

状态: Accepted

源代码

```
#M27948:FBI树
def find_type(s):
    zero = '0' in s
    one = '1' in s
    if zero and one:
        return 'F'
    if zero:
        return 'B'
    if one:
        return 'I'

class Node:
    def __init__(self, type, lchild=None, rchild=None):
        self.lchild = lchild
        self.rchild = rchild
        self.type = type
    def traverse(self):
        l = self.lchild.traverse() if self.lchild != None else ''
        r = self.rchild.traverse() if self.rchild != None else ''
        return l + r + self.type

def tree_parser(s, depth):
    l = len(s)
    if depth < 1:
        return Node(find_type(s))
    return Node(find_type(s), tree_parser(s[:l // 2], depth - 1), tree_

n = int(input())
string = input()
print(tree_parser(string, n).traverse())
```

27925: 小组队列

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

思路：

使用列表和dict，存储当前小组排列状态。

代码

```
n = int(input())
queue = [[] for _ in range(n)]
```

```
queues = []
types = {}
for i in range(n):
    for j in input().split():
        types[j] = i

while 1:
    order = input().split()
    if order[0] == "STOP":
        break

    if order[0] == "ENQUEUE":
        queue[types[order[1]]].append(order[1])
        if not types[order[1]] in queues:
            queues.append(types[order[1]])

    if order[0] == "DEQUEUE":
        print(queue[queues[0]].pop(0))
        if queue[queues[0]] == []:
            queues.pop(0)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

27928: 遍历树

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

思路：

先找到根节点，然后递归，按值大小顺序遍历。

代码

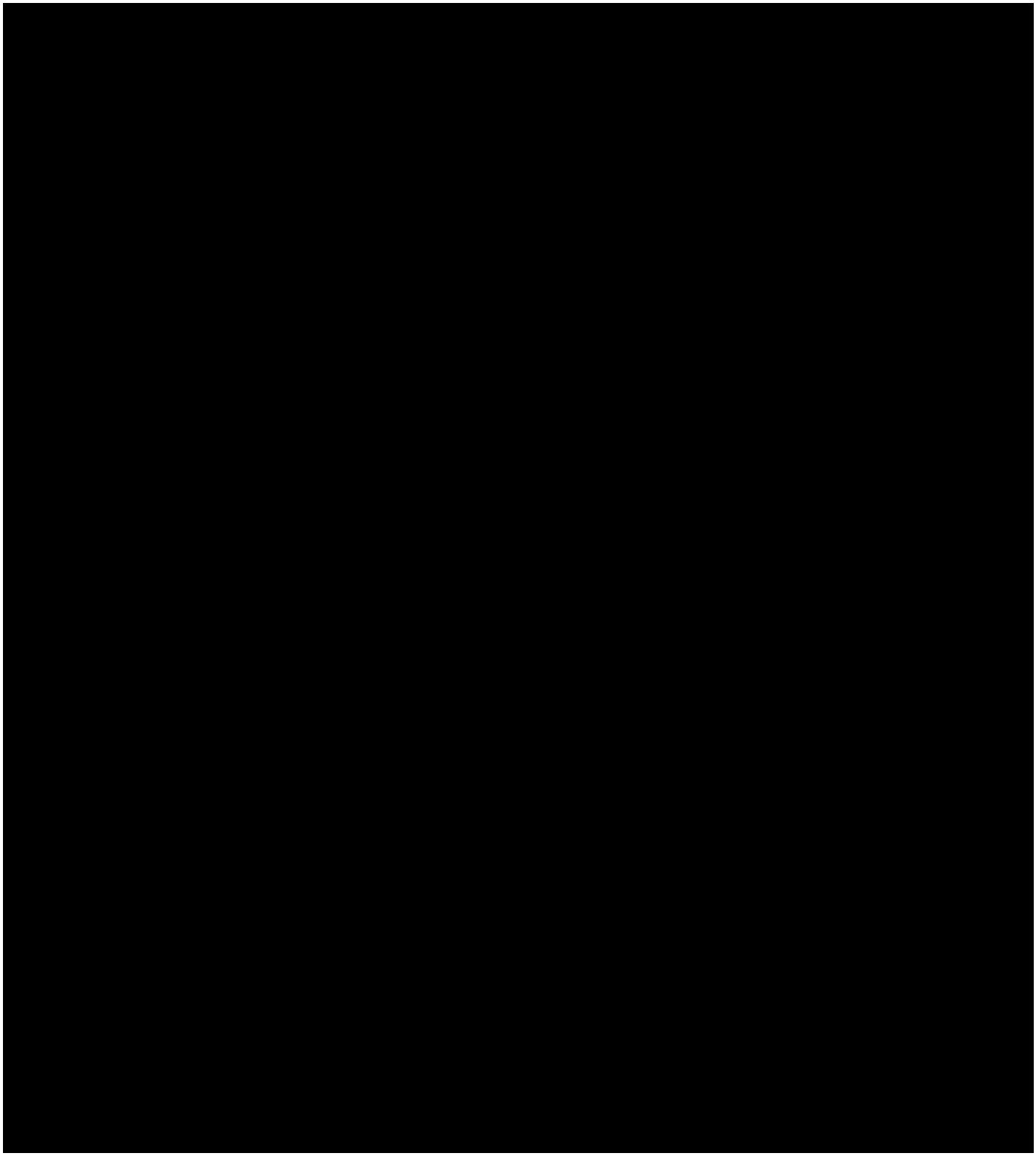
```
#T27928:遍历树
class Node:
    values = {}
    def __init__(self, id):
        self.id = id
        self.children = []
        self.father = None
    def set_value(self, value):
```

```
        self.value = value
        Node.values[value] = self

    def traverse(self):
        if self.children == []:
            print(self.value)
            return
        l = sorted([self.value] + self.children)
        for item in l:
            if item != self.value:
                Node.values[item].traverse()
            else:
                print(self.value)
        return
childrens = []
n = int(input())
nodes = {index: Node(index) for index in range(n)}
for i in range(n):
    iinput = [int(x) for x in input().split()]
    nodes[i].set_value(iinput[0])
    if len(iinput) > 1:
        nodes[i].children = iinput[1:]
        childrens += iinput[1:]

for i in range(n):
    if not nodes[i].value in childrens:
        nodes[i].traverse()
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



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。==

考场上ac5，小组队列考场上尝试自己实现类似链表的数据结构，结果debug较难，没有按时完成。考试结束后学到了比较好的思路。