

Assignment #7: April 月考

Updated 1557 GMT+8 Apr 3, 2024

2024 spring, Compiled 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:

字符串切片

代码

```
1  '''
2  2100017810 刘思瑞
3  '''
4  s = input().split()
5  print(' '.join(s[::-1]))
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''

s = input().split()
print(' '.join(s[::-1]))
```

27951: 机器翻译

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

思路:

FIFO

代码

```
1  '''
2  2100017810 刘思瑞
3  '''
4  from collections import deque
5  M,N = map(int,input().split())
6  l = list(map(int,input().split()))
7  m = deque()
8  count = 0
9  for i in l:
10     if i in m:
11         continue
12     if len(m) == M:
13         m.popleft()
14     m.append(i)
15     count +=1
16  print(count)
```

代码运行截图

状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''

from collections import deque
M,N = map(int,input().split())
l = list(map(int,input().split()))
m = deque()
count = 0
for i in l:
    if i in m:
        continue
    if len(m) == M:
        m.popleft()
    m.append(i)
    count +=1
print(count)
```

27932: Less or Equal

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

思路:

排序

代码

```
1  '''
2  2100017810 刘思瑞
3  '''
4  n,k = map(int,input().split())
5  l = list(map(int,input().split()))
6  if k == 0:
7      if 1 in l:
8          print(-1)
9      else:
10         print(1)
```

```

11 elif k == n:
12     print(max(l))
13 else:
14     l.sort(reverse=-1)
15     if l[n-k] != l[n-k-1]:
16         print(l[n-k])
17     else:
18         print(-1)

```

代码运行截图

状态: Accepted

源代码

```

'''
2100017810 刘思瑞
'''
n,k = map(int,input().split())
l = list(map(int,input().split()))
if k == 0:
    if 1 in l:
        print(-1)
    else:
        print(1)
elif k == n:
    print(max(l))
else:
    l.sort(reverse=-1)
    if l[n-k] != l[n-k-1]:
        print(l[n-k])
    else:
        print(-1)

```

27948: FBI树

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

思路:

递归建树

代码

```

1 '''
2 2100017810 刘思瑞
3 '''
4 class treenode():
5     def __init__(self,value):
6         self.value = value
7         self.left = None

```

```

8         self.right = None
9
10    def decide(node):
11        if node.left:
12            if node.right.value == node.left.value:
13                return node.right.value
14            else:
15                return 'F'
16
17    def build(N,l):
18        if N == 0:
19            return treenode(['B','I'][l[0]])
20        o = treenode(0)
21        ll = l[:2*(N-1)]
22        lr = l[2*(N-1):]
23        o.left = build(N-1,ll)
24        o.right = build(N-1,lr)
25        o.value = decide(o)
26        return o
27
28    def postorder(tree):
29        if tree != None:
30            postorder(tree.left)
31            postorder(tree.right)
32            print(tree.value,end='')
33
34
35    N = int(input())
36    s = list(map(int,list(input()))))
37    postorder(build(N,s))

```

代码运行截图

```
'''
2100017810 刘思瑞
'''

class treeNode():
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def decide(node):
    if node.left:
        if node.right.value == node.left.value:
            return node.right.value
        else:
            return 'F'

def build(N, l):
    if N == 0:
        return treeNode(['B', 'I'][l[0]])
    o = treeNode(0)
    ll = l[:2**(N-1)]
    lr = l[2**(N-1):]
    o.left = build(N-1, ll)
    o.right = build(N-1, lr)
    o.value = decide(o)
    return o

def postorder(tree):
    if tree.left:
        postorder(tree.left)
```

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

代码

```
1 from collections import deque
2
3 t = int(input())
4 groups_dict = {}
5 member_to_group = {}
6
7 for _ in range(t):
8     members_list = list(map(int, input().split()))
9     group_id = members_list[0]
10    groups_dict[group_id] = deque()
11    for member in members_list:
```

```
12         member_to_group[member] = group_id
13
14     queue = deque()
15     queue_set = set()
16
17     while True:
18         command = input().split()
19         if command[0] == 'STOP':
20             break
21         elif command[0] == 'ENQUEUE':
22             x = int(command[1])
23             group = member_to_group.get(x, None)
24             if group is None:
25                 group = x
26                 groups_dict[group] = deque([x])
27                 member_to_group[x] = group
28             else:
29                 groups_dict[group].append(x)
30             if group not in queue_set:
31                 queue.append(group)
32                 queue_set.add(group)
33         elif command[0] == 'DEQUEUE':
34             if queue:
35                 group = queue[0]
36                 x = groups_dict[group].popleft()
37                 print(x)
38                 if not groups_dict[group]:
39                     queue.popleft()
40                     queue_set.remove(group)
```

代码运行截图

状态: Accepted

源代码

```
from collections import deque

t = int(input())
groups_dict = {}
member_to_group = {}

for _ in range(t):
    members_list = list(map(int, input().split()))
    group_id = members_list[0]
    groups_dict[group_id] = deque()
    for member in members_list:
        member_to_group[member] = group_id

queue = deque()
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)
        if group is None:
            group = x
            groups_dict[group] = deque([x])
            member_to_group[x] = group
        else:
            groups_dict[group].append(x)
```

27928: 遍历树

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

思路:

先判断父节点再递归

代码

```
1 '''
2 2100017810 刘思瑞
3 '''
4 class treenode():
5     def __init__(self,value,child):
6         self.value = value
7         self.child = child
```



```

8
9 def sortorder(tree):
10     global hasvis,node
11     while True:
12         if tree != None:
13             temp = []
14             for i in [tree.value]+tree.child:
15                 if i not in hasvis:
16                     temp.append(i)
17             if not temp :
18                 return
19             m = min(temp)
20             print(m)
21             hasvis.add(m)
22             sortorder(node[m])
23
24 n = int(input())
25 node = dict()
26 for i in range(n):
27     s = list(map(int,input().split()))
28     v,l = s[0],s[1:]
29     node[v] = treenode(v,l)
30     if i == 0:
31         origin = node[v]
32     if origin.value in l:
33         origin = node[v]
34 hasvis = set()
35 sortorder(origin)

```

代码运行截图

状态: Accepted

源代码

```

'''
2100017810 刘思瑞
'''

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

def traverse print(root, nodes):

```

2. 学习总结和收获

对树有了进一步的认识，并且感觉对于遍历和建树的递归操作更加熟练了