Assignment #7: April 月考

Updated 1557 GMT+8 Apr 3, 2024

2024 spring, Complied by 夏天明 元培学院

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

操作系统：Windows 10 | 22H2

Python编程环境：Spyder IDE 5.4.3 | Python 3.11.4 64-bit

1. 题目

27706: 逐词倒放

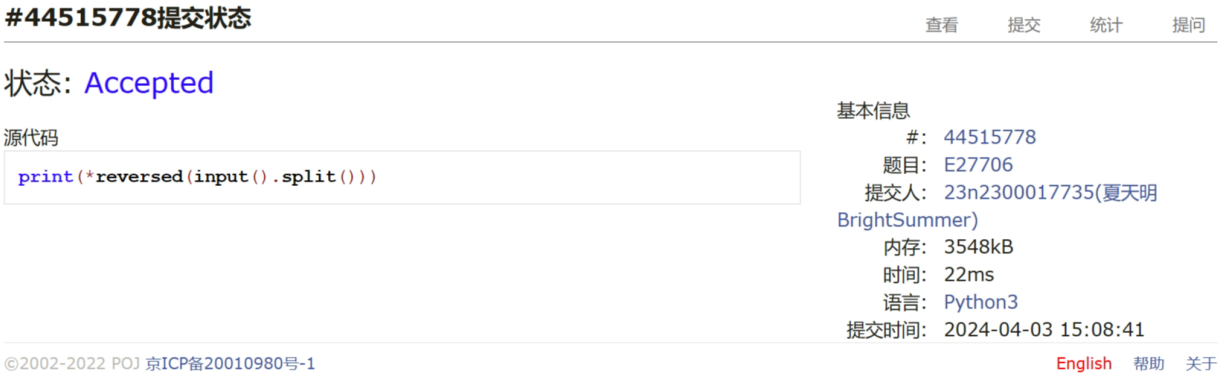
http://cs101.openjudge.cn/practice/27706/

思路：直接reversed。其实写麻烦了，更好的代码是[::-1]

代码

```
print(*reversed(input().split()))
```

代码运行截图



27951: 机器翻译

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

思路：用队列直接实现

代码

```
from collections import deque
m, n = map(int, input().split())
word = [int(i) for i in input().split()]
memory = deque()
ans = 0
for w in word:
    if w not in memory:
        memory.append(w)
        ans += 1
        if len(memory) > m:
            memory.popleft()
print(ans)
```

代码运行截图

#44515885提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque
m, n = map(int, input().split())
word = [int(i) for i in input().split()]
memory = deque()
ans = 0
for w in word:
    if w not in memory:
        memory.append(w)
        ans += 1
        if len(memory) > m:
            memory.popleft()
print(ans)
```

基本信息

#: 44515885
题目: E27951
提交人: 23n2300017735(夏天明 BrightSummer)
内存: 3664kB
时间: 25ms
语言: Python3
提交时间: 2024-04-03 15:11:42

English 帮助 关于

27932: Less or Equal

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

思路：排序之后直接索引。但0和n是两个特殊情况，需要单独讨论

代码

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

代码运行截图

#44516034提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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

基本信息

#: 44516034
题目: M27932
提交人: 23n2300017735(夏天明 BrightSummer)
内存: 9928kB
时间: 44ms
语言: Python3
提交时间: 2024-04-03 15:17:27

English 帮助 关于

27948: FBI树

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

思路：递归地实现

代码

```
class FBI:
    def __init__(self, val, l, r):
        self.val = val
        self.left = l
        self.right = r

def generateFBI(s):
    val = 'B'
    if '1' in s:
        if '0' in s:
            val = 'F'
        else:
            val = 'I'
    if len(s) == 1:
        return FBI(val, None, None)
    return FBI(val, generateFBI(s[:len(s)//2]), generateFBI(s[len(s)//2:]))

def postSeq(fbi):
    if not fbi:
        return ''
    return postSeq(fbi.left) + postSeq(fbi.right) + fbi.val

N = int(input())
print(postSeq(generateFBI(input())))
```

代码运行截图

#44516252提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class FBI:
    def __init__(self, val, l, r):
        self.val = val
        self.left = l
        self.right = r

def generateFBI(s):
    val = 'B'
    if '1' in s:
        if '0' in s:
            val = 'F'
        else:
            val = 'I'
    if len(s) == 1:
        return FBI(val, None, None)
    return FBI(val, generateFBI(s[:len(s)//2]), generateFBI(s[len(s)//2:

def postSeq(fbi):
    if not fbi:
        return ''
    return postSeq(fbi.left) + postSeq(fbi.right) + fbi.val

N = int(input())
print(postSeq(generateFBI(input())))
```

基本信息

#: 44516252
题目: M27948
提交人: 23n2300017735(夏天明BrightSummer)
内存: 3920kB
时间: 24ms
语言: Python3
提交时间: 2024-04-03 15:25:36

27925: 小组队列

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

思路：首先用桶来确定每个成员所在的组号（group）。我用堆（q）来根据每个成员插入的组的优先级（pos）以及成员本身在组内的优先级（sub_pos）每次弹出第一个元素。用桶来存储队尾的优先级以确定新插入元素的优先级（sub_pos和newPos）。用桶来更新出光的队的优先级回到0（in_q_num）。值得注意的是，这种算法并非最高效。

代码

```
from heapq import heappop, heappush

group = dict()
q = []
t = int(input())
for i in range(t):
    for member in input().split():
        group[member] = i
newPos = 0
pos = [0]*t
sub_pos = [0]*t
in_q_num = [0]*t
while (s:=input()) != 'STOP':
    token = s.split()
```

```
    if token[0] == 'DEQUEUE':
        p, sp, g, idx = heappop(q)
        print(idx)
        in_q_num[g] -= 1
        if not in_q_num[g]:
            pos[g] = 0
    else:
        g = group[token[1]]
        if not pos[g]:
            newPos += 1
            pos[g] = newPos
        sub_pos[g] += 1
        heappush(q, (pos[g], sub_pos[g], g, token[1]))
```

代码运行截图

27928: 遍历树

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

思路：正常建树，在遍历的时候先对数值进行排序，然后对于子节点递归地调用遍历。这里节点与节点的数值常常混合处理，因此使用正常的类写法不太方便，这里采取了child桶来模拟树节点。notRoot桶来寻找没有父节点的根节点的数值。

代码

```python
from collections import defaultdict

child = defaultdict(list)
notRoot = defaultdict(int)

def getSeq(root):
    if not child[root]:
        return [root]
    return [num for item in [getSeq(node) if node != root else [node] for node in
sorted([root] + child[root])] for num in item]

for o in range(int(input())):
    s = [int(i) for i in input().split()]
    child[s[0]] = s[1:]
    notRoot[s[0]]
    for node in s[1:]:
        notRoot[node] = 1
for root, i in notRoot.items():
    if i == 0:
        break
for val in getSeq(root):
    print(val)
```

代码运行截图

#44516838提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```python
from collections import defaultdict

child = defaultdict(list)
notRoot = defaultdict(int)

def getSeq(root):
    if not child[root]:
        return [root]
    return [num for item in [getSeq(node) if node != root else [node] fo

for o in range(int(input())):
    s = [int(i) for i in input().split()]
    child[s[0]] = s[1:]
    notRoot[s[0]]
    for node in s[1:]:
        notRoot[node] = 1
for root, i in notRoot.items():
    if i == 0:
        break
for val in getSeq(root):
    print(val)
```

基本信息

#: 44516838
题目: T27928
提交人: 23n2300017735(夏天明 BrightSummer)
内存: 3676kB
时间: 23ms
语言: Python3
提交时间: 2024-04-03 15:42:48

English 帮助 关于

2. 学习总结和收获

本次考试题目较为灵活，不是传统的模板题。灵活的题目也有灵活的处理方式，对于并不直接和要使用的数据结构相对应的输入数据，采用不同的桶来处理往往可以高效解决。

在 遍历树 一题中也可看出树和桶的密切联系。涉及到数值、排序等问题，桶可以很好地解决，此时恰好可以用字典来实现树结构的存储（即存储桶中数值之间的父子关系），递归的过程即为沿着桶存储的子元素反复代入直至叶元素。进而使得代码非常精简。另一方面，传统的类实现树，也可以看作桶，其中每个元素的数据集就是子元素的信息。