Assignment #6: "树"算：Huffman,BinHeap,BST,AVL,DisjointSet

Updated 2214 GMT+8 March 24, 2024

2024 spring, Complied by ==陈亚偲==

说明：

1）这次作业内容不简单，耗时长的话直接参考题解。

2）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。

3）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。

4）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：Windows

Python编程环境：Spyder IDE 5.2.2

1. 题目

22275: 二叉搜索树的遍历

http://cs101.openjudge.cn/practice/22275/

思路：

中序、前序转后序

代码

```python
#
# -*- coding: utf-8 -*-
"""
Created on Sat Mar 30 00:39:49 2024

@author: 陈亚偲 2300011106
"""
kk=int(input())
class node:
    def __init__(self,num):
        self.num=num
        self.r=None
        self.l=None
a=[int(i) for i in input().split()]
b=[node(i) for i in a]
ans=[]
def down(p):#p是node
    if p!=None:
        down(p.l)
        down(p.r)
        ans.append(p.num)
    return
def build(x,y,k):#x是节点的编号，y是顺序列表，找x的左右子节点
    ll=y[:k]
    rr=y[k+1:]
    lm=9999999
    rm=9999999
    for i in ll:
        lm=min(lm,a.index(i))
    for i in rr:
        rm=min(rm,a.index(i))
    if ll:
        b[a.index(x)].l=b[lm]
    if rr:
        b[a.index(x)].r=b[rm]
    return
def boza(x,y):
    k=y.index(x)
    build(x,y,k)
    if b[a.index(x)].l!=None:
        boza(b[a.index(x)].l.num,y[:k])
    if b[a.index(x)].r!=None:
        boza(b[a.index(x)].r.num,y[k+1:])
    return
boza(a[0],[i+1 for i in range(kk)])
down(b[0])
print(*ans)
```

代码运行截图 ==（至少包含有"Accepted"）==

状态: Accepted

源代码

```python
# -*- coding: utf-8 -*-
"""
Created on Sat Mar 30 00:39:49 2024

@author: 陈亚偲 2300011106
"""
kk=int(input())
class node:
    def __init__(self,num):
        self.num=num
        self.r=None
        self.l=None
a=[int(i) for i in input().split()]
b=[node(i) for i in a]
ans=[]
def down(p):#p是node
    if p!=None:
        down(p.l)
        down(p.r)
        ans.append(p.num)
    return
def build(x,y,k):#x是节点的编号, y是顺序列表, 找x的左右子节点
    ll=y[:k]
    rr=y[k+1:]
    lm=9999999
    rm=9999999
    for i in ll:
        lm=min(lm,a.index(i))
    for i in rr:
        rm=min(rm,a.index(i))
    if ll:
        b[a.index(x)].l=b[lm]
    if rr:
        b[a.index(x)].r=b[rm]
    return
def boza(x,y):
    k=y.index(x)
    build(x,y,k)
    if b[a.index(x)].l!=None:
        boza(b[a.index(x)].l.num,y[:k])
    if b[a.index(x)].r!=None:
        boza(b[a.index(x)].r.num,y[k+1:])
    return
boza(a[0],[i+1 for i in range(kk)])
down(b[0])
print(*ans)
```

05455: 二叉搜索树的层次遍历

http://cs101.openjudge.cn/practice/05455/

思路:

复制的答案，思路是一个一个往里加，加的时候左右分，最后用t表示出来

代码

```python
#
class TreeNode:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def insert(node, value):
    if node is None:
        return TreeNode(value)
    if value < node.value:
        node.left = insert(node.left, value)
    elif value > node.value:
        node.right = insert(node.right, value)
    return node

def t(root):
    queue = [root]
    traversal = []
    while queue:
        node = queue.pop(0)
        traversal.append(node.value)
        if node.left:
            queue.append(node.left)
        if node.right:
            queue.append(node.right)
    return traversal

numbers = list(map(int, input().strip().split()))
numbers = list(dict.fromkeys(numbers))
root = None
for number in numbers:
    root = insert(root, number)
v = t(root)
print(' '.join(map(str, v)))
```

代码运行截图 ==（至少包含有"Accepted"） ==

状态: Accepted

源代码

```
#
class TreeNode:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def insert(node, value):
    if node is None:
        return TreeNode(value)
    if value < node.value:
        node.left = insert(node.left, value)
    elif value > node.value:
        node.right = insert(node.right, value)
    return node

def t(root):
    queue = [root]
    traversal = []
    while queue:
        node = queue.pop(0)
        traversal.append(node.value)
        if node.left:
            queue.append(node.left)
        if node.right:
            queue.append(node.right)
    return traversal

numbers = list(map(int, input().strip().split()))
numbers = list(dict.fromkeys(numbers))
root = None
for number in numbers:
    root = insert(root, number)
v = t(root)
print(' '.join(map(str, v)))
```

04078: 实现堆结构

http://cs101.openjudge.cn/practice/04078/

练习自己写个BinHeap。当然机考时候，如果遇到这样题目，直接import heapq。手搓栈、队列、堆、AVL等，考试前需要搓个遍。

思路:

堆和哈夫曼还完全不会，打算清明补

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

22161: 哈夫曼编码树

http://cs101.openjudge.cn/practice/22161/

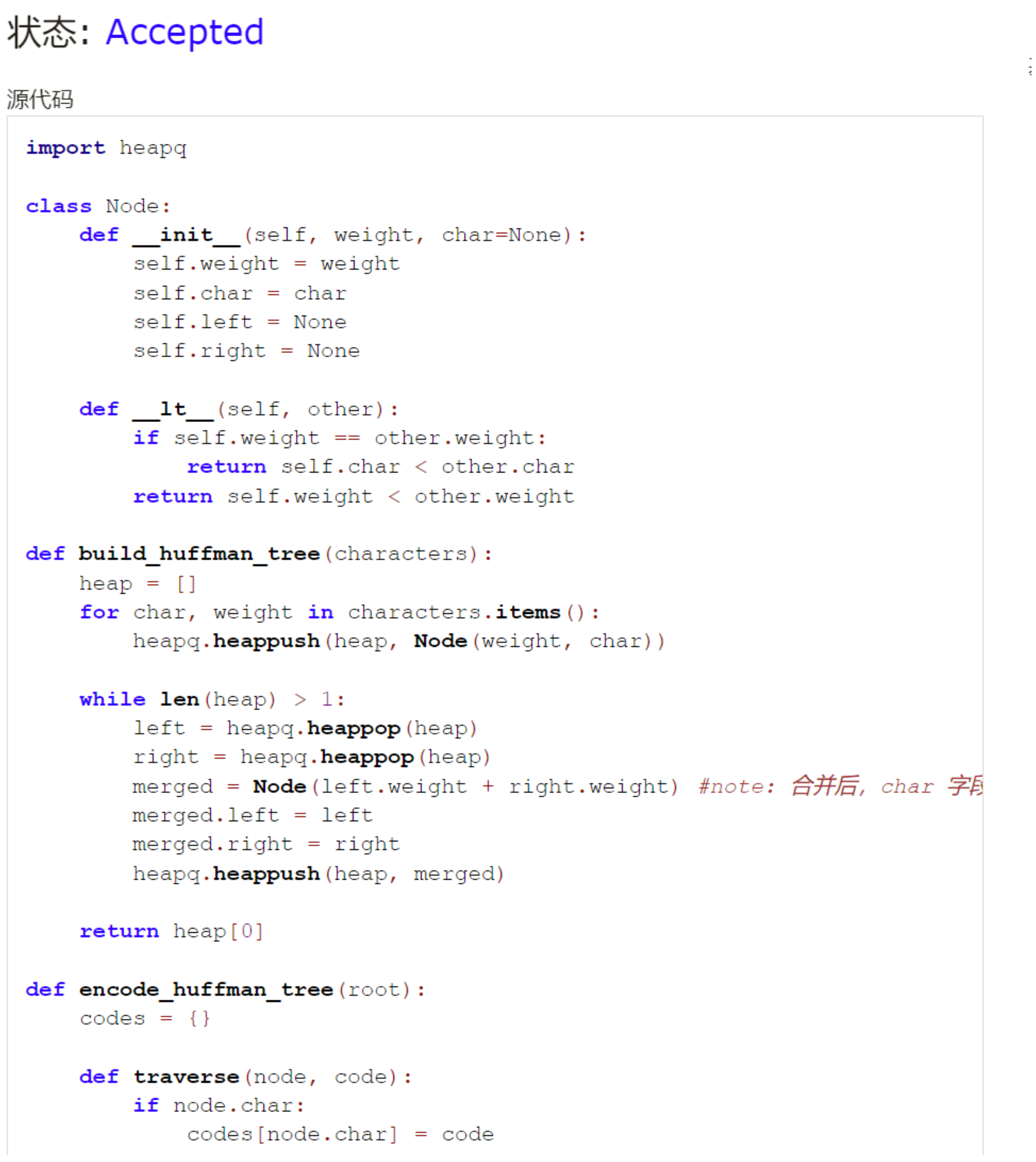
思路：抄的答案，哈夫曼还没搞懂

代码

```
#
import heapq

class Node:
    def __init__(self, weight, char=None):
        self.weight = weight
        self.char = char
        self.left = None
        self.right = None

    def __lt__(self, other):
        if self.weight == other.weight:
            return self.char < other.char
        return self.weight < other.weight

def build_huffman_tree(characters):
    heap = []
    for char, weight in characters.items():
        heapq.heappush(heap, Node(weight, char))

    while len(heap) > 1:
        left = heapq.heappop(heap)
        right = heapq.heappop(heap)
        merged = Node(left.weight + right.weight) #note: 合并后，char 字段默认值是空
        merged.left = left
        merged.right = right
        heapq.heappush(heap, merged)

    return heap[0]

def encode_huffman_tree(root):
    codes = {}

    def traverse(node, code):
        if node.char:
            codes[node.char] = code
        else:
            traverse(node.left, code + '0')
            traverse(node.right, code + '1')

    traverse(root, '')
    return codes

def huffman_encoding(codes, string):
    encoded = ''
    for char in string:
        encoded += codes[char]
    return encoded

def huffman_decoding(root, encoded_string):
    decoded = ''
    node = root
    for bit in encoded_string:
        if bit == '0':
            node = node.left
        else:
            node = node.right

        if node.char:
            decoded += node.char
            node = root
    return decoded

# 读取输入
n = int(input())
characters = {}
for _ in range(n):
    char, weight = input().split()
    characters[char] = int(weight)

#string = input().strip()
#encoded_string = input().strip()

# 构建哈夫曼编码树
huffman_tree = build_huffman_tree(characters)

# 编码和解码
codes = encode_huffman_tree(huffman_tree)

strings = []
while True:
    try:
        line = input()
        if line:
            strings.append(line)
        else:
            break
    except EOFError:
        break
```

```
results = []
#print(strings)
for string in strings:
    if string[0] in ('0','1'):
        results.append(huffman_decoding(huffman_tree, string))
    else:
        results.append(huffman_encoding(codes, string))

for result in results:
    print(result)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```
import heapq

class Node:
    def __init__(self, weight, char=None):
        self.weight = weight
        self.char = char
        self.left = None
        self.right = None

    def __lt__(self, other):
        if self.weight == other.weight:
            return self.char < other.char
        return self.weight < other.weight

def build_huffman_tree(characters):
    heap = []
    for char, weight in characters.items():
        heapq.heappush(heap, Node(weight, char))

    while len(heap) > 1:
        left = heapq.heappop(heap)
        right = heapq.heappop(heap)
        merged = Node(left.weight + right.weight)  #note: 合并后，char 字
        merged.left = left
        merged.right = right
        heapq.heappush(heap, merged)

    return heap[0]

def encode_huffman_tree(root):
    codes = {}

    def traverse(node, code):
        if node.char:
            codes[node.char] = code
```

晴问9.5: 平衡二叉树的建立

https://sunnywhy.com/sfbj/9/5/359

思路：

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

02524: 宗教信仰

http://cs101.openjudge.cn/practice/02524/

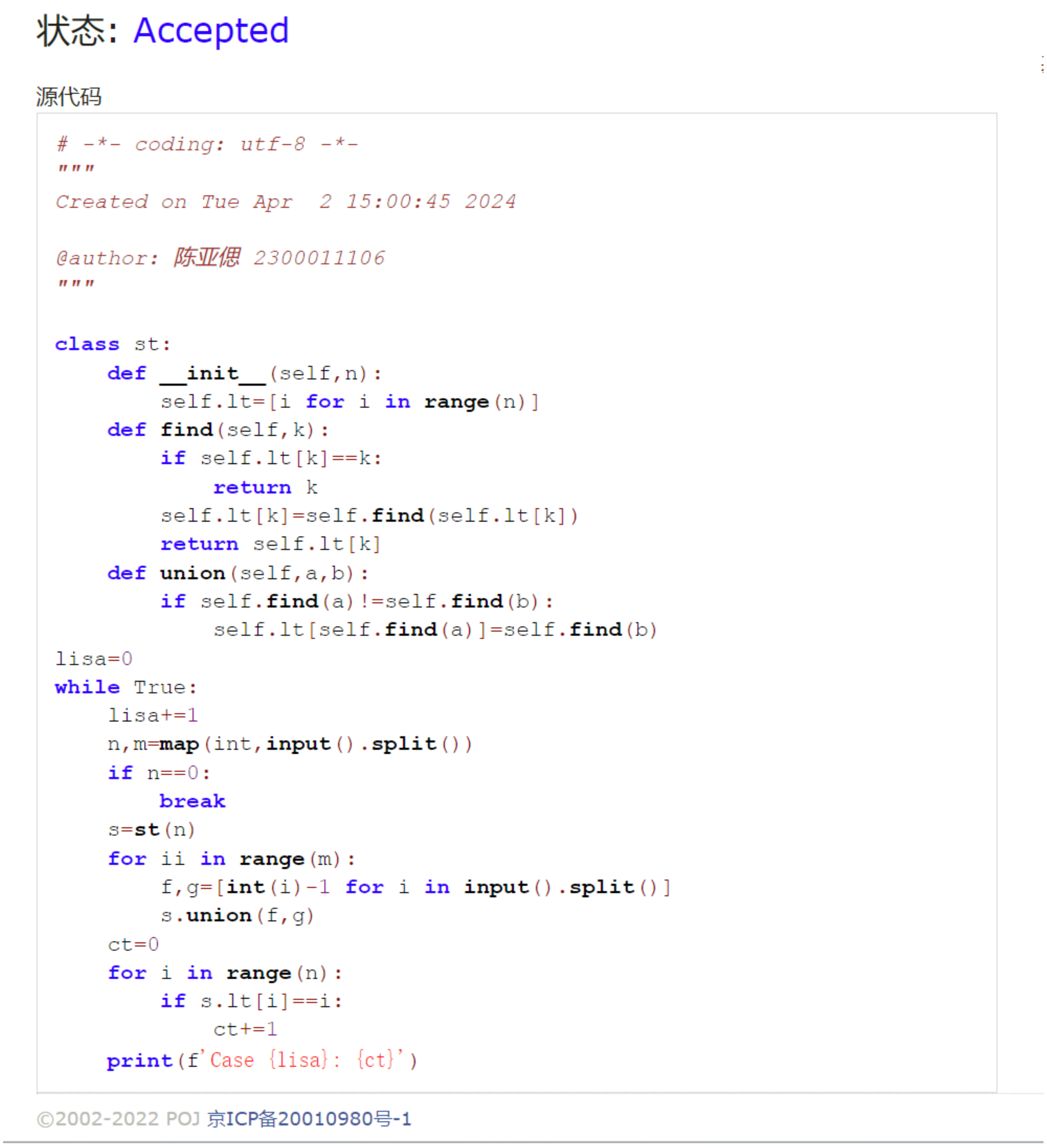
思路：

直接并查集做

代码

```
#
# -*- coding: utf-8 -*-
"""
Created on Tue Apr  2 15:00:45 2024

@author: 陈亚偲 2300011106
"""

class st:
    def __init__(self,n):
        self.lt=[i for i in range(n)]
    def find(self,k):
        if self.lt[k]==k:
            return k
        self.lt[k]=self.find(self.lt[k])
        return self.lt[k]
    def union(self,a,b):
        if self.find(a)!=self.find(b):
            self.lt[self.find(a)]=self.find(b)
lisa=0
while True:
    lisa+=1
    n,m=map(int,input().split())
    if n==0:
        break
    s=st(n)
    for ii in range(m):
        f,g=[int(i)-1 for i in input().split()]
        s.union(f,g)
    ct=0
    for i in range(n):
        if s.lt[i]==i:
            ct+=1
    print(f'Case {lisa}: {ct}')
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```
# -*- coding: utf-8 -*-
"""
Created on Tue Apr  2 15:00:45 2024

@author: 陈亚偲 2300011106
"""

class st:
    def __init__(self,n):
        self.lt=[i for i in range(n)]
    def find(self,k):
        if self.lt[k]==k:
            return k
        self.lt[k]=self.find(self.lt[k])
        return self.lt[k]
    def union(self,a,b):
        if self.find(a)!=self.find(b):
            self.lt[self.find(a)]=self.find(b)
lisa=0
while True:
    lisa+=1
    n,m=map(int,input().split())
    if n==0:
        break
    s=st(n)
    for ii in range(m):
        f,g=[int(i)-1 for i in input().split()]
        s.union(f,g)
    ct=0
    for i in range(n):
        if s.lt[i]==i:
            ct+=1
    print(f'Case {lisa}: {ct}')
```

©2002-2022 POJ 京ICP备20010980号-1

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。==

宗教信仰那道题卡了一整天，自己写永远爆栈，要么就莫名其妙地WA，后来套模板才过的，还是不懂union那里