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

Updated 2214 GMT+8 March 24, 2024

2024 spring, Complied by ==同学的姓名、院系==

赵云天 生命科学学院

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：win10

Python编程环境：Spyder IDE 5.2.2

C/C++编程环境：

1. 题目

22275: 二叉搜索树的遍历

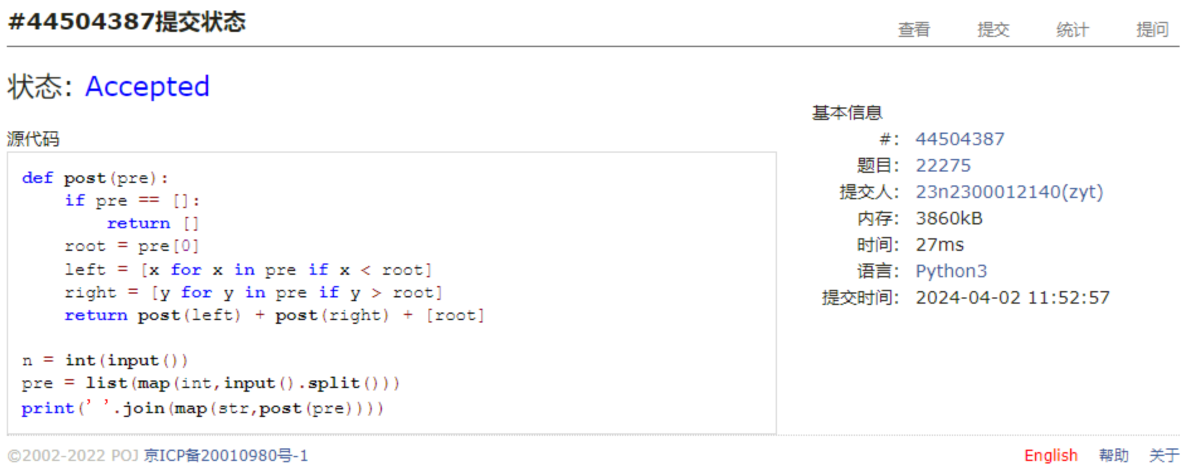
http://cs101.openjudge.cn/practice/22275/

思路：按照定义书写即可

代码

```python
def post(pre):
    if pre == []:
        return []
    root = pre[0]
    left = [x for x in pre if x < root]
    right = [y for y in pre if y > root]
    return post(left) + post(right) + [root]

n = int(input())
pre = list(map(int,input().split()))
print(' '.join(map(str,post(pre))))
```

代码运行截图 == （至少包含有"Accepted"） ==


#44504387提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```python
def post(pre):
    if pre == []:
        return []
    root = pre[0]
    left = [x for x in pre if x < root]
    right = [y for y in pre if y > root]
    return post(left) + post(right) + [root]

n = int(input())
pre = list(map(int,input().split()))
print(' '.join(map(str,post(pre))))
```

基本信息

#:	44504387
题目:	22275
提交人:	23n2300012140(zyt)
内存:	3860kB
时间:	27ms
语言:	Python3
提交时间:	2024-04-02 11:52:57

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

05455: 二叉搜索树的层次遍历

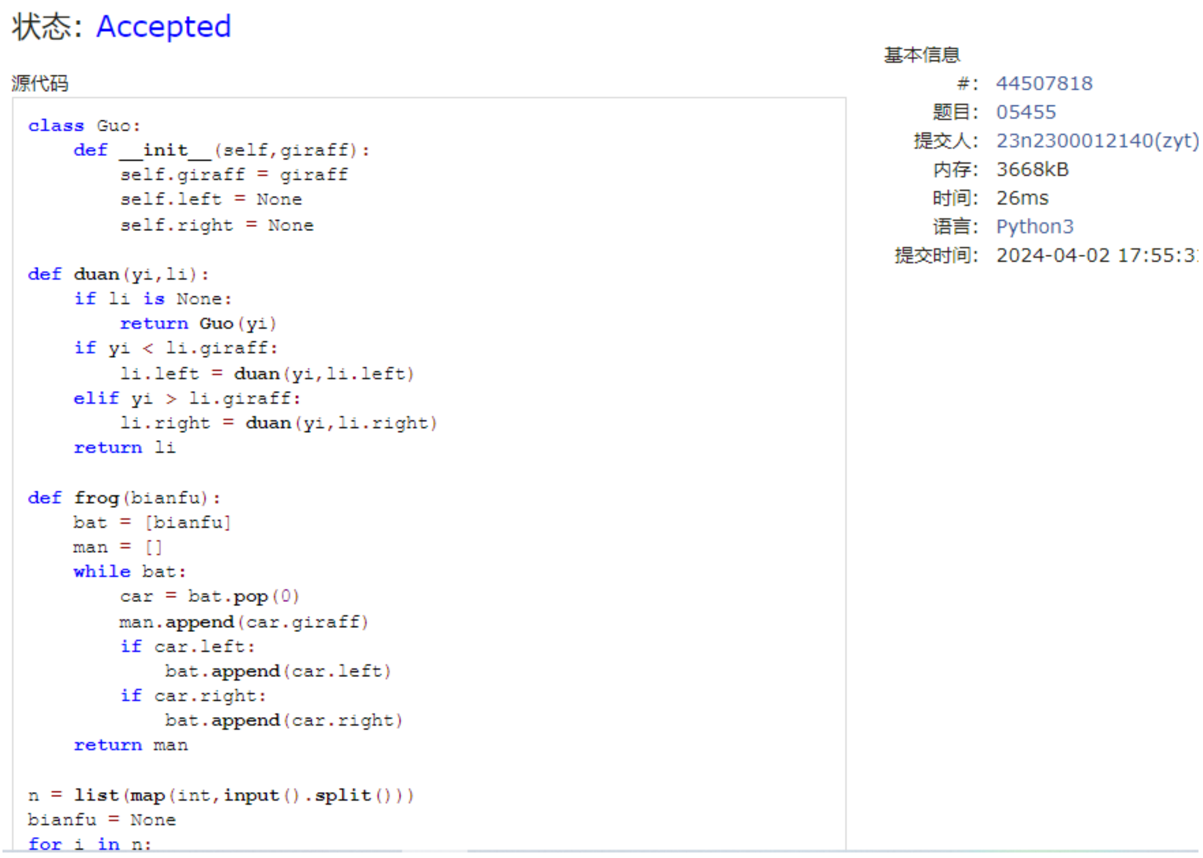
http://cs101.openjudge.cn/practice/05455/

思路：对类的方法的意义的理解多了一些

代码

```python
class Guo:
    def __init__(self,giraff):
        self.giraff = giraff
        self.left = None
        self.right = None

def duan(yi,li):
    if li is None:
        return Guo(yi)
    if yi < li.giraff:
        li.left = duan(yi,li.left)
    elif yi > li.giraff:
        li.right = duan(yi,li.right)
    return li

def frog(bianfu):
    bat = [bianfu]
    man = []
    while bat:
        car = bat.pop(0)
        man.append(car.giraff)
        if car.left:
            bat.append(car.left)
        if car.right:
            bat.append(car.right)
    return man

n = list(map(int,input().split()))
bianfu = None
for i in n:
    bianfu = duan(i,bianfu)
man = frog(bianfu)
print(' '.join(map(str,man)))
```

代码运行截图 ==（至少包含有"Accepted"）==



状态: Accepted

源代码

```
class Guo:
    def __init__(self,giraff):
        self.giraff = giraff
        self.left = None
        self.right = None

def duan(yi,li):
    if li is None:
        return Guo(yi)
    if yi < li.giraff:
        li.left = duan(yi,li.left)
    elif yi > li.giraff:
        li.right = duan(yi,li.right)
    return li

def frog(bianfu):
    bat = [bianfu]
    man = []
    while bat:
        car = bat.pop(0)
        man.append(car.giraff)
        if car.left:
            bat.append(car.left)
        if car.right:
            bat.append(car.right)
    return man

n = list(map(int,input().split()))
bianfu = None
for i in n:
```

基本信息

#: 44507818
题目: 05455
提交人: 23n2300012140(zyt)
内存: 3668kB
时间: 26ms
语言: Python3
提交时间: 2024-04-02 17:55:3

04078: 实现堆结构

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

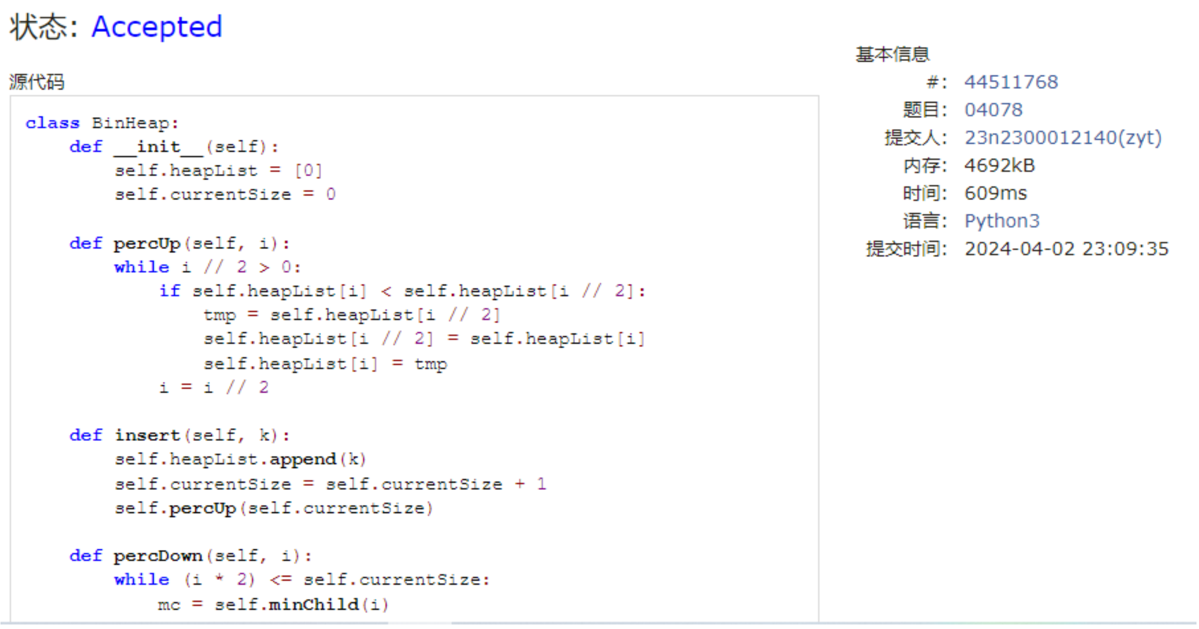
练习自己写个BinHeap。当然机考时候，如果遇到这样题目，直接import heapq。手搓栈、队列、堆、AVL等，考试前需要搓个遍。

思路：用了一定时间来理解题解

代码

```python
class BinHeap:
    def __init__(self):
        self.heapList = [0]
        self.currentSize = 0

    def percUp(self, i):
        while i // 2 > 0:
            if self.heapList[i] < self.heapList[i // 2]:
                tmp = self.heapList[i // 2]
                self.heapList[i // 2] = self.heapList[i]
                self.heapList[i] = tmp
            i = i // 2

    def insert(self, k):
        self.heapList.append(k)
        self.currentSize = self.currentSize + 1
        self.percUp(self.currentSize)

    def percDown(self, i):
        while (i * 2) <= self.currentSize:
            mc = self.minChild(i)
            if self.heapList[i] > self.heapList[mc]:
                tmp = self.heapList[i]
                self.heapList[i] = self.heapList[mc]
                self.heapList[mc] = tmp
            i = mc

    def minChild(self, i):
        if i * 2 + 1 > self.currentSize:
            return i * 2
        else:
            if self.heapList[i * 2] < self.heapList[i * 2 + 1]:
                return i * 2
            else:
                return i * 2 + 1

    def delMin(self):
        retval = self.heapList[1]
        self.heapList[1] = self.heapList[self.currentSize]
        self.currentSize = self.currentSize - 1
        self.heapList.pop()
        self.percDown(1)
        return retval

    def buildHeap(self, alist):
        i = len(alist) // 2
        self.currentSize = len(alist)
        self.heapList = [0] + alist[:]
        while (i > 0):
            #print(f'i = {i}, {self.heapList}')
            self.percDown(i)
            i = i - 1
        #print(f'i = {i}, {self.heapList}')


n = int(input().strip())
bh = BinHeap()
for _ in range(n):
    inp = input().strip()
    if inp[0] == '1':
        bh.insert(int(inp.split()[1]))
    else:
        print(bh.delMin())
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```
class BinHeap:
    def __init__(self):
        self.heapList = [0]
        self.currentSize = 0

    def percUp(self, i):
        while i // 2 > 0:
            if self.heapList[i] < self.heapList[i // 2]:
                tmp = self.heapList[i // 2]
                self.heapList[i // 2] = self.heapList[i]
                self.heapList[i] = tmp
            i = i // 2

    def insert(self, k):
        self.heapList.append(k)
        self.currentSize = self.currentSize + 1
        self.percUp(self.currentSize)

    def percDown(self, i):
        while (i * 2) <= self.currentSize:
            mc = self.minChild(i)
```

基本信息

#:	44511768
题目:	04078
提交人:	23n2300012140(zyt)
内存:	4692kB
时间:	609ms
语言:	Python3
提交时间:	2024-04-02 23:09:35

22161: 哈夫曼编码树

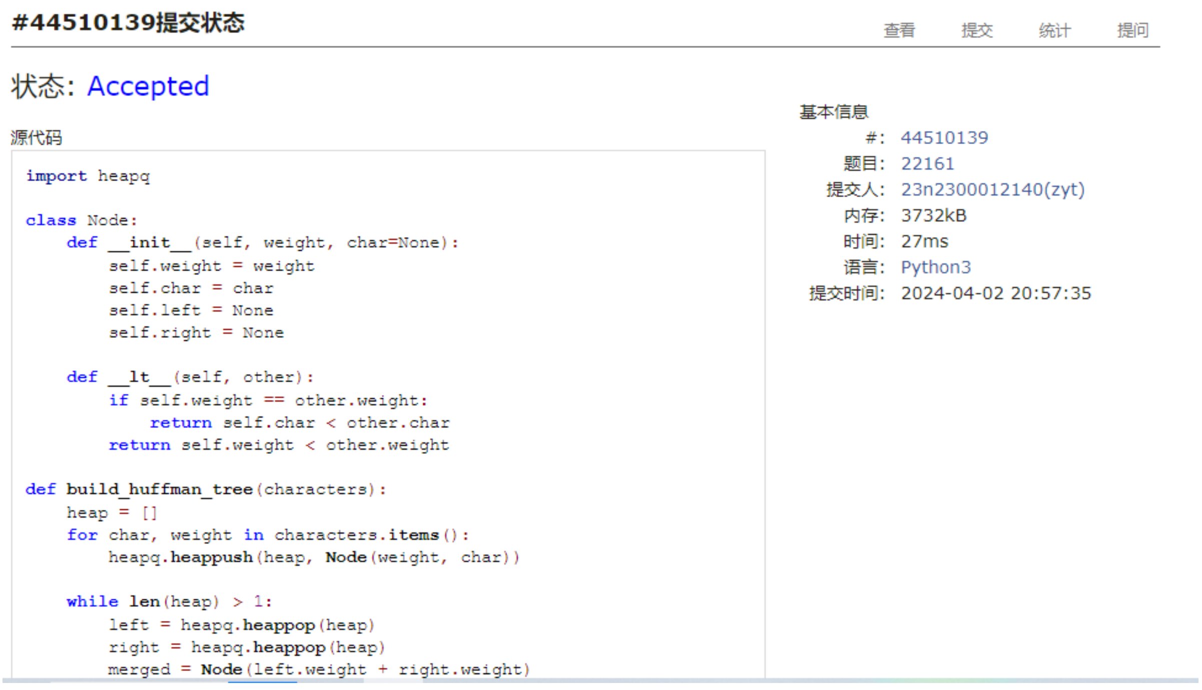
http://cs101.openjudge.cn/practice/22161/

思路：相加后创建节点并入堆，直至只剩一个

代码

```python
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)
        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

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

huffman_tree = build_huffman_tree(characters)

codes = encode_huffman_tree(huffman_tree)

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

results = []
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"）==


#44510139提交状态

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)
```

基本信息

#:	44510139
题目:	22161
提交人:	23n2300012140(zyt)
内存:	3732kB
时间:	27ms
语言:	Python3
提交时间:	2024-04-02 20:57:35

晴问9.5: 平衡二叉树的建立

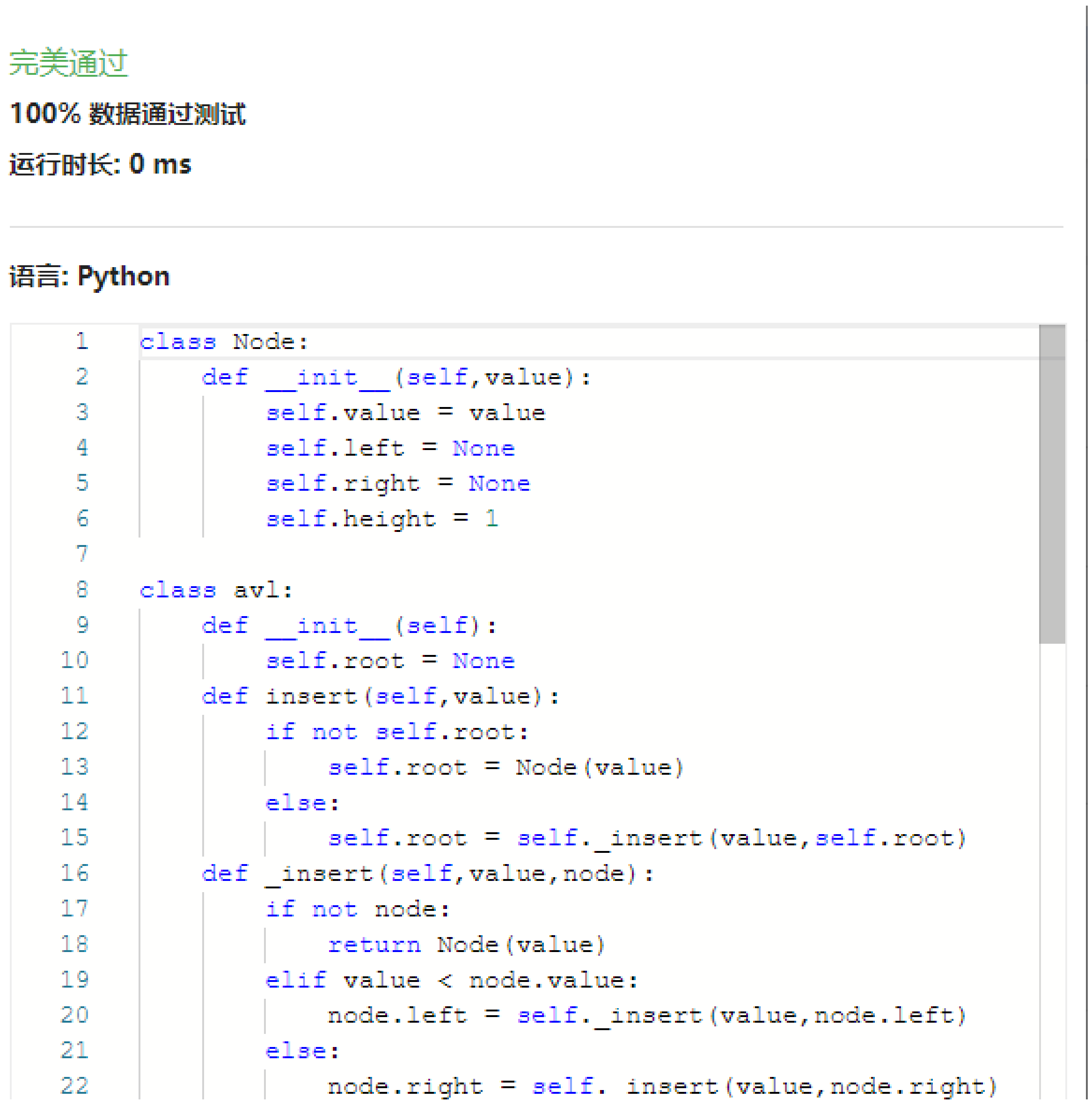
https://sunnywhy.com/sfbj/9/5/359

思路：自己对着题解试着手搓了一遍

代码

```
class Node:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None
        self.height = 1

class avl:
    def __init__(self):
        self.root = None

    def insert(self, value):
        if not self.root:
            self.root = Node(value)
        else:
            self.root = self._insert(value, self.root)

    def _insert(self, value, node):
        if not node:
            return Node(value)
        elif value < node.value:
            node.left = self._insert(value, node.left)
        else:
            node.right = self._insert(value, node.right)
        node.height = 1 + max(self._get_height(node.left),
self._get_height(node.right))

        balance = self._get_balance(node)

        if balance > 1:
            if value < node.left.value: # 树形是 LL
                return self._rotate_right(node)
            else:   # 树形是 LR
                node.left = self._rotate_left(node.left)
                return self._rotate_right(node)

        if balance < -1:
            if value > node.right.value:    # 树形是 RR
                return self._rotate_left(node)
            else:   # 树形是 RL
                node.right = self._rotate_right(node.right)
                return self._rotate_left(node)

        return node

    def _get_height(self, node):
        if not node:
            return 0
        return node.height

    def _get_balance(self, node):
        if not node:
            return 0
        return self._get_height(node.left) - self._get_height(node.right)

    def _rotate_left(self, z):
        y = z.right
        T2 = y.left
        y.left = z
        z.right = T2
        z.height = 1 + max(self._get_height(z.left), self._get_height(z.right))
        y.height = 1 + max(self._get_height(y.left), self._get_height(y.right))
        return y


    def _rotate_right(self, y):
        x = y.left
        T2 = x.right
        x.right = y
        y.left = T2
        y.height = 1 + max(self._get_height(y.left), self._get_height(y.right))
        x.height = 1 + max(self._get_height(x.left), self._get_height(x.right))
        return x

    def preorder(self):
        return self._preorder(self.root)

    def _preorder(self, node):
        if not node:
            return []
        return [node.value] + self._preorder(node.left) +
self._preorder(node.right)

n = int(input().strip())
seq = list(map(int, input().strip().split()))

av = avl()
for value in seq:
    avl.insert(value)

print(' '.join(map(str, avl.preorder())))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"） ==



完美通过

100% 数据通过测试

运行时长: 0 ms

语言: Python

```
class Node:
    def __init__(self,value):
        self.value = value
        self.left = None
        self.right = None
        self.height = 1

class avl:
    def __init__(self):
        self.root = None
    def insert(self,value):
        if not self.root:
            self.root = Node(value)
        else:
            self.root = self._insert(value,self.root)
    def _insert(self,value,node):
        if not node:
            return Node(value)
        elif value < node.value:
            node.left = self._insert(value,node.left)
        else:
            node.right = self._insert(value,node.right)
```

02524: 宗教信仰

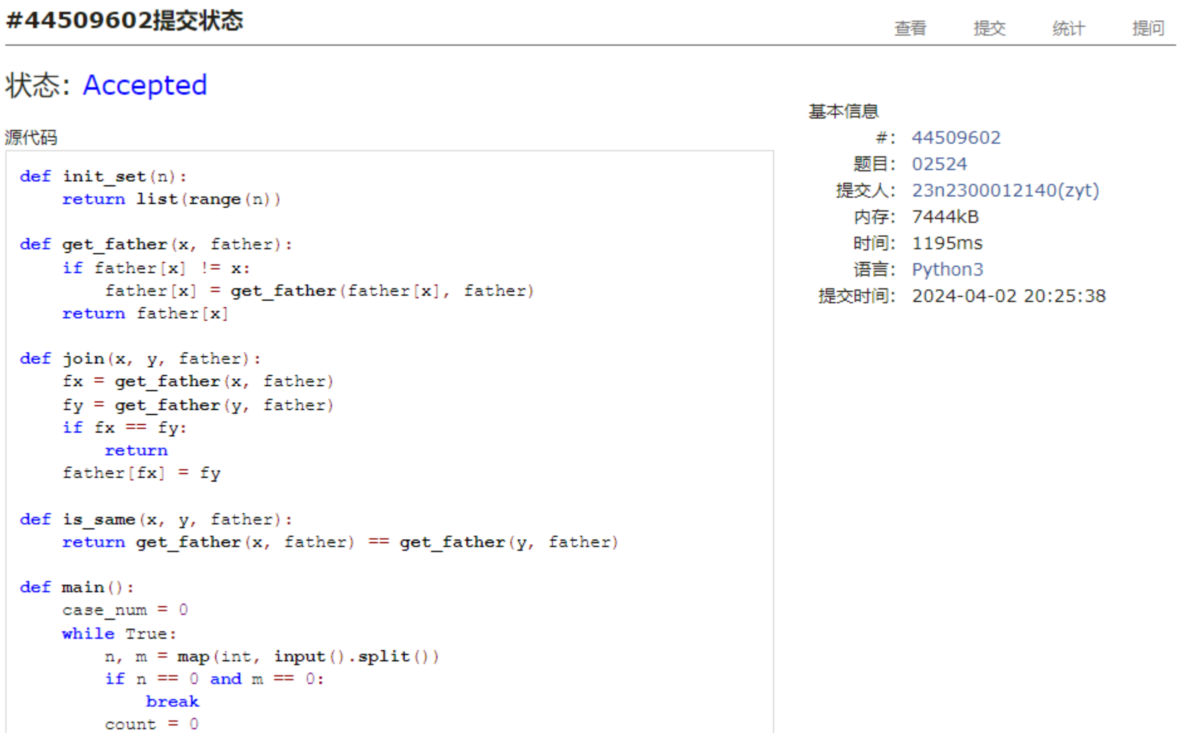
http://cs101.openjudge.cn/practice/02524/

思路：看的题解

代码

```python
def init_set(n):
    return list(range(n))

def get_father(x, father):
    if father[x] != x:
        father[x] = get_father(father[x], father)
    return father[x]

def join(x, y, father):
    fx = get_father(x, father)
    fy = get_father(y, father)
    if fx == fy:
        return
    father[fx] = fy

def is_same(x, y, father):
    return get_father(x, father) == get_father(y, father)

def main():
    case_num = 0
    while True:
        n, m = map(int, input().split())
        if n == 0 and m == 0:
            break
        count = 0
        father = init_set(n)
        for _ in range(m):
            s1, s2 = map(int, input().split())
            join(s1 - 1, s2 - 1, father)
        for i in range(n):
            if father[i] == i:
                count += 1
        case_num += 1
        print(f"Case {case_num}: {count}")

if __name__ == "__main__":
    main()
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```python
def init_set(n):
    return list(range(n))

def get_father(x, father):
    if father[x] != x:
        father[x] = get_father(father[x], father)
    return father[x]

def join(x, y, father):
    fx = get_father(x, father)
    fy = get_father(y, father)
    if fx == fy:
        return
    father[fx] = fy

def is_same(x, y, father):
    return get_father(x, father) == get_father(y, father)

def main():
    case_num = 0
    while True:
        n, m = map(int, input().split())
        if n == 0 and m == 0:
            break
        count = 0
```

基本信息

#:	44509602
题目:	02524
提交人:	23n2300012140(zyt)
内存:	7444kB
时间:	1195ms
语言:	Python3
提交时间:	2024-04-02 20:25:38

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。==

一半的题可以靠自己或看一点题解做出，另一半需要反复看题解才能较好地理解。