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

Updated 2214 GMT+8 March 24, 2024

2024 spring, Complied by 陈奕好 工学院

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：macOS Sonoma 14.4 (23E214)

Python编程环境：PyCharm 2023.3.1 (Professional Edition)

1. 题目

22275: 二叉搜索树的遍历

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

思路：先建树，加入一个insert模块，用于每次放置新数字，postOrder用于后序输出。

代码

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

```

```python

def insert(root, value):
    if root is None:
        return TreeNode(value)
    elif value > root.val:
        root.right = insert(root.right, value)
    else:
        root.left = insert(root.left, value)
    return root


def postOrder(root):
    if root is None:
        return []
    return postOrder(root.left) + postOrder(root.right) + [root.val]


N = int(input())
arr = list(map(int, input().split()))
root = None
for value in arr:
    root = insert(root, value)
print(*postOrder(root))

```

代码运行截图 （至少包含有"Accepted"）

#44402292提交状态

查看 提交

状态: Accepted

源代码

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


def insert(root, value):
    if root is None:
        return TreeNode(value)
    elif value > root.val:
        root.right = insert(root.right, value)  #
    else:
        root.left = insert(root.left, value)
    return root


def postOrder(root):
    if root is None:
        return []
    return postOrder(root.left) + postOrder(root.right) + [root.val]
```

基本信息

#: 44402292
题目: 22275
提交人: 23n23000110
内存: 4100kB
时间: 29ms
语言: Python3
提交时间: 2024-03-25 2

05455: 二叉搜索树的层次遍历

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

思路：和上面一样，但追加了levelOrder板块

代码

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


def insert(root, value):
    if root is None:
        return TreeNode(value)
    elif value > root.val:
        root.right = insert(root.right, value)  #
    else:
        root.left = insert(root.left, value)
    return root


def postOrder(root):
    if root is None:
        return []
    return postOrder(root.left) + postOrder(root.right) + [root.val]


def levelOrder(root):
    queue = [root]
    result = []
    while queue:
        current = queue.pop(0)
        result.append(current.val)
        if current.left is not None:
            queue.append(current.left)
        if current.right is not None:
            queue.append(current.right)
    return result


# N = int(input())
arr = list(map(int, input().split()))
vaild = set()
```

```
root = None
for value in arr:
    if value in vaild:
        continue
    vaild.add(value)
    root = insert(root, value)
print(*levelOrder(root))
```

代码运行截图 （至少包含有"Accepted"）

#44402407提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44402407
题目: 05455
提交人: 23n2300011030(陈奕好)
内存: 3676kB
时间: 24ms
语言: Python3
提交时间: 2024-03-25 23:11:31

源代码

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


def insert(root, value):
    if root is None:
        return TreeNode(value)
    elif value > root.val:
        root.right = insert(root.right, value)   #
    else:
        root.left = insert(root.left, value)
    return root


def postOrder(root):
    if root is None:
        return []
```

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：insert_key不断比较parent和本身
extract_min最为关键，把最大项移动到最开始，重新建树

min_heapify则是把列表归位

代码

```
class MinHeap:
    def __init__(self):
        self.heap = []

```

```
    def parent(self, i):
        return (i - 1) // 2

    def left_child(self, i):
        return 2 * i + 1

    def right_child(self, i):
        return 2 * i + 2

    def get_min(self):
        return self.heap[0]

    def insert_key(self, k):
        self.heap.append(k)
        i = len(self.heap) - 1
        while i != 0 and self.heap[self.parent(i)] > self.heap[i]:
            self.heap[i], self.heap[self.parent(i)] = self.heap[self.parent(i)], self.heap[i]
            i = self.parent(i)

    def decrease_key(self, i, new_val):
        self.heap[i] = new_val
        while i != 0 and self.heap[self.parent(i)] > self.heap[i]:
            self.heap[i], self.heap[self.parent(i)] = self.heap[self.parent(i)], self.heap[i]
            i = self.parent(i)

    def extract_min(self):
        if len(self.heap) <= 0:
            return float('inf')
        if len(self.heap) == 1:
            return self.heap.pop()
        root = self.heap[0]
        self.heap[0] = self.heap.pop()
        self.min_heapify(0)
        return root

    def min_heapify(self, i):
        l = self.left_child(i)
        r = self.right_child(i)
        smallest = i
        if l < len(self.heap) and self.heap[l] < self.heap[i]:
            smallest = l
        if r < len(self.heap) and self.heap[r] < self.heap[smallest]:
            smallest = r
        if smallest != i:
            self.heap[i], self.heap[smallest] = self.heap[smallest], self.heap[i]
            self.min_heapify(smallest)


heap = MinHeap()
for i in range(int(input())):
```

```
    opt = list(map(int, input().split()))
    if opt[0] == 1:
        heap.insert_key(opt[1])
    else:
        print(heap.extract_min())


```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44402782提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class MinHeap:
    def __init__(self):
        self.heap = []

    def parent(self, i):
        return (i - 1) // 2

    def left_child(self, i):
        return 2 * i + 1

    def right_child(self, i):
        return 2 * i + 2

    def get_min(self):
        return self.heap[0]

    def insert_key(self, k):
        self.heap.append(k)
        i = len(self.heap) - 1
```

基本信息

#: 44402782
题目: 04078
提交人: 23n2300011030(陈奕好)
内存: 4132kB
时间: 858ms
语言: Python3
提交时间: 2024-03-26 00:02:23

22161: 哈夫曼编码树

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

思路：抄了题解，确实题解代码好

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)
        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)

#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"）

#44403914提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44403914
题目: 22161
提交人: 23n2300011030(陈奕好)
内存: 3696kB
时间: 26ms
语言: Python3
提交时间: 2024-03-26 09:54:22

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):
```

晴问9.5: 平衡二叉树的建立

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

思路：这里询问了copilot，了解了“旋转”

代码

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


def insert(root, value):
    if root is None:
        return TreeNode(value)
    elif value > root.val:
        root.right = insert(root.right, value)  #
    else:
        root.left = insert(root.left, value)

    root.height = 1 + max(get_height(root.left), get_height(root.right))

    balance = get_balance(root)

    # 如果节点失衡，那么有4种情况需要处理
    # Case 1 - 左左
```

```
    if balance > 1 and value < root.left.val:
        return right_rotate(root)

    # Case 2 - 右右
    if balance < -1 and value > root.right.val:
        return left_rotate(root)

    # Case 3 - 左右
    if balance > 1 and value > root.left.val:
        root.left = left_rotate(root.left)
        return right_rotate(root)

    # Case 4 - 右左
    if balance < -1 and value < root.right.val:
        root.right = right_rotate(root.right)
        return left_rotate(root)
    return root

def get_height(root):
    if root is None:
        return 0
    else:
        return root.height


def get_balance(root):
    if root is None:
        return 0
    return get_height(root.left) - get_height(root.right)


def left_rotate(z):
    y = z.right
    T2 = y.left
    y.left = z
    z.right = T2
    z.height = 1 + max(get_height(z.left), get_height(z.right))
    y.height = 1 + max(get_height(y.left), get_height(y.right))
    return y


def right_rotate(y):
    x = y.left
    T2 = x.right
    x.right = y
    y.left = T2
    y.height = 1 + max(get_height(y.left), get_height(y.right))
    x.height = 1 + max(get_height(x.left), get_height(x.right))
    return x


def preOrder(root):
```

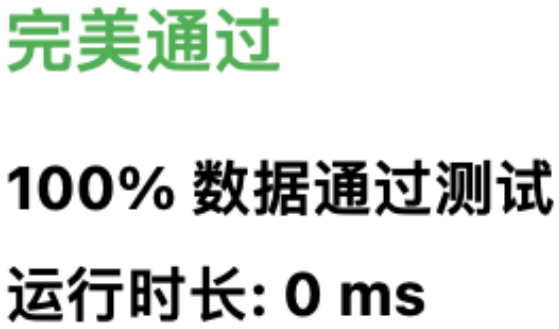
```
    if root is None:
        return []
    return [root.val] + preOrder(root.left) + preOrder(root.right)


N = int(input())
arr = list(map(int, input().split()))

root = None
for value in arr:
    root = insert(root, value)
# print(get_height(root))
print(*preOrder(root))
# print(*ans)


```

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：一开始没想到用并查集，确实好用

代码

```
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
```

```
    parent[find(x)] = find(y)

case = 0
while True:
    n, m = map(int, input().split())
    if n == 0 and m == 0:
        break
    parent = list(range(n+1))
    for _ in range(m):
        i, j = map(int, input().split())
        union(i, j)
    religions = len(set(find(i) for i in range(1, n+1)))
    case += 1
    print("Case %d: %d" % (case, religions))

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#44406889提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
    parent[find(x)] = find(y)

case = 0
while True:
    n, m = map(int, input().split())
    if n == 0 and m == 0:
        break
    parent = list(range(n+1))
    for _ in range(m):
        i, j = map(int, input().split())
        union(i, j)
    religions = len(set(find(i) for i in range(1, n+1)))
    case += 1
    print("Case %d: %d" % (case, religions))
```

基本信息

#: 44406889
题目: 02524
提交人: 23n2300011030(陈奕好)
内存: 11020kB
时间: 1224ms
语言: Python3
提交时间: 2024-03-26 15:10:52

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。

并查集大法

```
def find(x):
    if parent[x] != x:
        parent[x] = find(parent[x])
    return parent[x]

def union(x, y):
    parent[find(x)] = find(y)
```