Assignment #5: "树"算：概念、表示、解析、遍历

Updated 2124 GMT+8 March 17, 2024

2024 spring, Complied by 尹柚鑫 光华管理学院 2100015878

说明：

1）The complete process to learn DSA from scratch can be broken into 4 parts:

Learn about Time complexities, learn the basics of individual Data Structures, learn the basics of Algorithms, and practice Problems.

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 11

Python编程环境：jupter notebook

1. 题目

27638: 求二叉树的高度和叶子数目

http://cs101.openjudge.cn/practice/27638/

耗时：20mins

思路：建立孩子列表和爸爸列表，先遍历，让每一个孩子找到爸爸，同时添加叶节点，从而确定叶子数目，之后对于每一个叶节点，不断向上找爸爸，从而确定二叉树的高度

代码

```
#
leaf=[]
father=[-1]*105
for i in range(int(input())):
    l,r=map(int,input().split())
    if l==-1 and r==-1:
```

```
            leaf.append(i)# 添加叶节点
        #孩子找爸爸
        father[l]=i
        father[r]=i
ans=0
for x in leaf:
        #每个叶节点不断向上找爸爸
        cnt=0
        while father[x]>=0:
            cnt+=1
            x=father[x]
        ans=max(ans,cnt)
print(ans,len(leaf))
```

代码运行截图 ==（至少包含有"Accepted"）==

#44395865提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
leaf=[]
father=[-1]*105
for i in range(int(input())):
    l,r=map(int,input().split())
    if l==-1 and r==-1:
        leaf.append(i)
    father[l]=i
    father[r]=i
ans=0
for x in leaf:
    cnt=0
    while father[x]>=0:
        cnt+=1
        x=father[x]
    ans=max(ans,cnt)
print(ans,len(leaf))
```

基本信息

#: 44395865
题目: 27638
提交人: 尹柚鑫(2100015878)
内存: 3600kB
时间: 23ms
语言: Python3
提交时间: 2024-03-25 15:57:58

24729: 括号嵌套树

http://cs101.openjudge.cn/practice/24729/

耗时：40mins

思路：先根据规则建树，之后前序，后序循环

代码

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

def parse_tree(s):
    stack=[]      # 记录父节点
    node=None
    for char in s:    #读入节点
        if char.isalpha():
```

```
            node=TreeNode(char)
            if stack:    #孩子找到爸爸
                stack[-1].children.append(node)
        elif char=='(':     #表明节点还有孩子，孩子变成爸爸，进栈
            if node:
                stack.append(node)
                node=None
        elif char==')':  #节点的孩子读完了，爸爸出栈
            if stack:
                node=stack.pop()    #返回爸爸节点
    return node  # 返回根节点

def preorder(node):
    output=[node.value]   #根节点在第一个，爸爸节点在子节点前面
    for child in node.children:   #连接子子孙孙
        output.extend(preorder(child))
    return ''.join(output)

def postorder(node):
    output=[]
    for child in node.children:   #连接子子孙孙
        output.extend(postorder(child))
    output.append(node.value)   #根节点在最后，爸爸节点在子节点后面
    return ''.join(output)

def main():
    s=input().strip()
    s=''.join(s.split())
    root=parse_tree(s)  # 解析树
    if root:
        print(preorder(root))
        print(postorder(root))
    else:
        print("input error")
if __name__=="__main__":
    main()
```

代码运行截图 ==（至少包含有"Accepted"）==

#44396825提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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

def parse_tree(s):
    stack=[]      # 记录父节点
    node=None
    for char in s:    #读入节点
        if char.isalpha():
            node=TreeNode(char)
            if stack:     #孩子找到爸爸
                stack[-1].children.append(node)
        elif char=='(':      #表明节点还有孩子，孩子变成爸爸，进栈
            if node:
                stack.append(node)
                node=None
        elif char==')':  #节点的孩子读完了，爸爸出栈
            if stack:
                node=stack.pop()     #返回爸爸节点
    return node  # 返回根节点
```

基本信息

#: 44396825
题目: 24729
提交人: 尹柚鑫(2100015878)
内存: 3688kB
时间: 25ms
语言: Python3
提交时间: 2024-03-25 16:54:43

02775: 文件结构“图”

http://cs101.openjudge.cn/practice/02775/

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

25140: 根据后序表达式建立队列表达式

http://cs101.openjudge.cn/practice/25140/

耗时：20mins

思路：先建树，对栈后序表达式建树，之后再一行一行逆序输出就是队列表达式

代码

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

```
        self.right=None

def parse_tree(s):
    stack=[]
    for char in s:
        node=TreeNode(char) #对每一个元素都建立为结点
        if char.isupper():
            node.right=stack.pop()     # 给运算符找运算数
            node.left=stack.pop()
        stack.append(node)       #把运算树放到栈中
    return stack[0]

    # 下面其实只需要一行一行地把树输出就行

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

n=int(input().strip())
for _ in range(n):
    s=input().strip()
    root=parse_tree(s)
    queue=level_output(root)
    print(''.join(queue[::-1]))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44411785提交状态

查看　提交　统计　提问

状态: Accepted

源代码

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

def parse_tree(s):
    stack=[]
    for char in s:
        node=TreeNode(char) #对每一个元素都建立为结点
        if char.isupper():
            node.right=stack.pop()     # 给运算符找运算数
            node.left=stack.pop()
        stack.append(node)       #把运算树放到栈中
    return stack[0]

    # 下面其实只需要一行一行地把树输出就行

def level_output(root):
    queue=[root]
```

基本信息

#: 44411785
题目: 25140
提交人: 尹柚鑫(2100015878)
内存: 3684kB
时间: 28ms
语言: Python3
提交时间: 2024-03-26 20:05:36

24750: 根据二叉树中后序序列建树

http://cs101.openjudge.cn/practice/24750/

耗时：40mins

思路：后序表达式最后一个一定是跟，从中序中找到这个根，就可以找到左右子树，从而递归

代码

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

#树和递归密不可分
def buildTree(inorder, postorder):
    if not inorder or not postorder:
        return None

    # 后序遍历的最后一个元素是当前的根节点
    root_val = postorder.pop()
    root = TreeNode(root_val)

    # 在中序遍历中找到根节点的位置
    root_index = inorder.index(root_val)

    # 构建右子树和左子树
    root.right = buildTree(inorder[root_index + 1:], postorder)
    root.left = buildTree(inorder[:root_index], postorder)

    return root


def preorderTraversal(root):
    result = []
    if root:
        result.append(root.val)
        result.extend(preorderTraversal(root.left))
        result.extend(preorderTraversal(root.right))
    return result


# 读取输入
inorder = input().strip()
postorder = input().strip()

# 构建树
root = buildTree(list(inorder), list(postorder))

# 输出前序遍历序列
print(''.join(preorderTraversal(root)))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44414439提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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


def buildTree(inorder, postorder):
    if not inorder or not postorder:
        return None

    # 后序遍历的最后一个元素是当前的根节点
    root_val = postorder.pop()
    root = TreeNode(root_val)

    # 在中序遍历中找到根节点的位置
    root_index = inorder.index(root_val)

    # 构建右子树和左子树
    root.right = buildTree(inorder[root_index + 1:], postorder)
    root.left = buildTree(inorder[:root_index], postorder)

    return root
```

基本信息

#: 44414439
题目: 24750
提交人: 尹柚鑫(2100015878)
内存: 7376kB
时间: 25ms
语言: Python3
提交时间: 2024-03-26 22:39:11

22158: 根据二叉树前中序序列建树

http://cs101.openjudge.cn/practice/22158/

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

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。==

1.树的题目经常和递归的想法有关，要多多回顾这些树的题目，同时加强练习

2.栈，队列，树这些其实都代表一种想法，可以帮助解题，在遇到一个题目的时候，可以先考虑每一步需要用到怎么样的数据结构

3.注意区分node.value和node的区别