Assignment #5: "树"算：概念、表示、解析、遍历

Updated 2124 GMT+8 March 17, 2024

2024 spring, Complied by ==同学的姓名、院系==

罗瑞哲 生命科学学院

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 10

Python编程环境：Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境：Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

1. 题目

27638: 求二叉树的高度和叶子数目

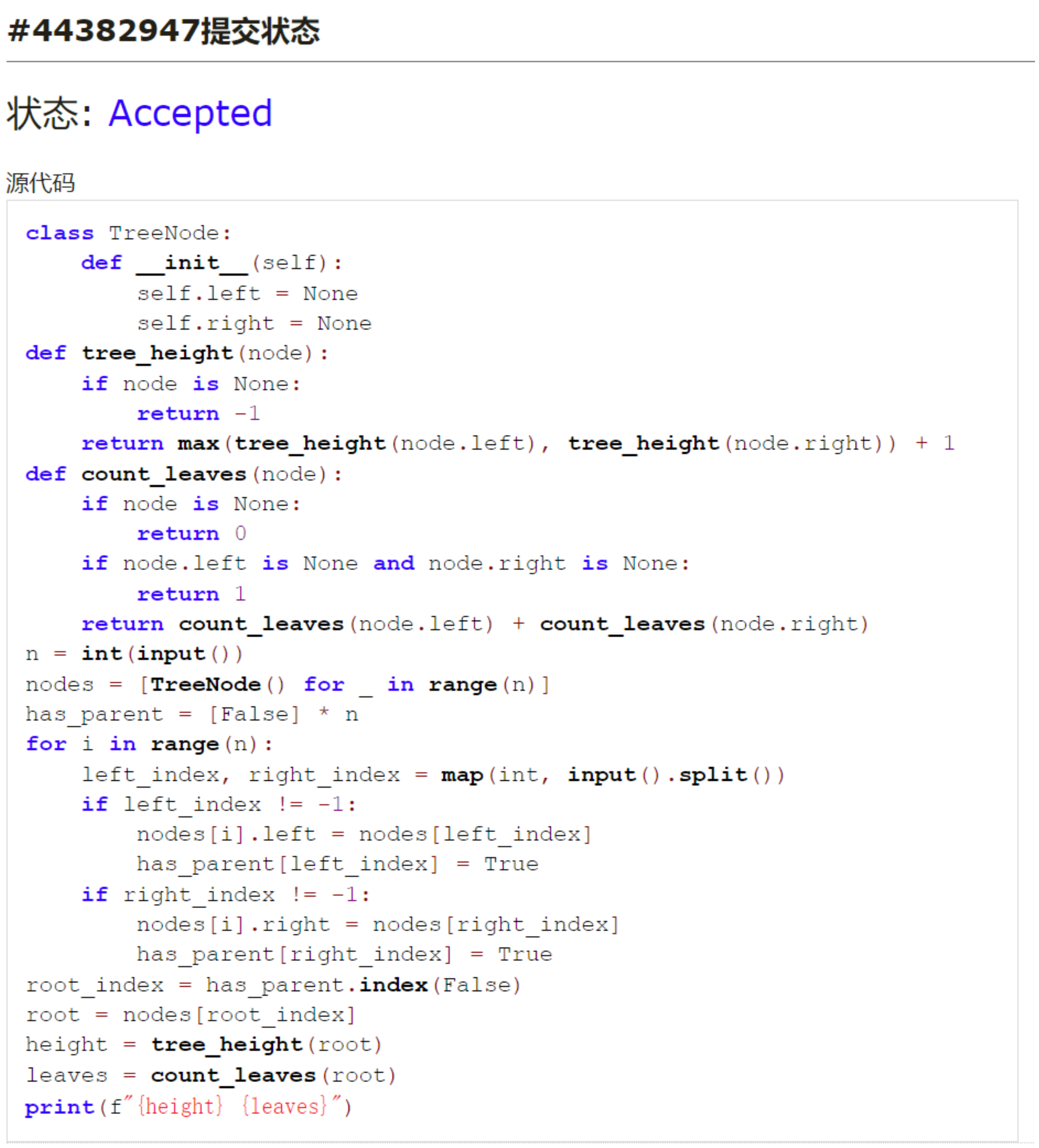
http://cs101.openjudge.cn/practice/27638/

思路：

代码

```
#
class TreeNode:
    def __init__(self):
        self.left = None
        self.right = None
def tree_height(node):
    if node is None:
        return -1
    return max(tree_height(node.left), tree_height(node.right)) + 1
def count_leaves(node):
    if node is None:
        return 0
    if node.left is None and node.right is None:
        return 1
    return count_leaves(node.left) + count_leaves(node.right)
n = int(input())
nodes = [TreeNode() for _ in range(n)]
has_parent = [False] * n
for i in range(n):
    left_index, right_index = map(int, input().split())
    if left_index != -1:
        nodes[i].left = nodes[left_index]
        has_parent[left_index] = True
    if right_index != -1:
        nodes[i].right = nodes[right_index]
        has_parent[right_index] = True
root_index = has_parent.index(False)
root = nodes[root_index]
height = tree_height(root)
leaves = count_leaves(root)
print(f"{height} {leaves}")
```

代码运行截图 ==（至少包含有"Accepted"）==



#44382947提交状态

状态: Accepted

源代码

```python
class TreeNode:
    def __init__(self):
        self.left = None
        self.right = None
def tree_height(node):
    if node is None:
        return -1
    return max(tree_height(node.left), tree_height(node.right)) + 1
def count_leaves(node):
    if node is None:
        return 0
    if node.left is None and node.right is None:
        return 1
    return count_leaves(node.left) + count_leaves(node.right)
n = int(input())
nodes = [TreeNode() for _ in range(n)]
has_parent = [False] * n
for i in range(n):
    left_index, right_index = map(int, input().split())
    if left_index != -1:
        nodes[i].left = nodes[left_index]
        has_parent[left_index] = True
    if right_index != -1:
        nodes[i].right = nodes[right_index]
        has_parent[right_index] = True
root_index = has_parent.index(False)
root = nodes[root_index]
height = tree_height(root)
leaves = count_leaves(root)
print(f"{height} {leaves}")
```

24729: 括号嵌套树

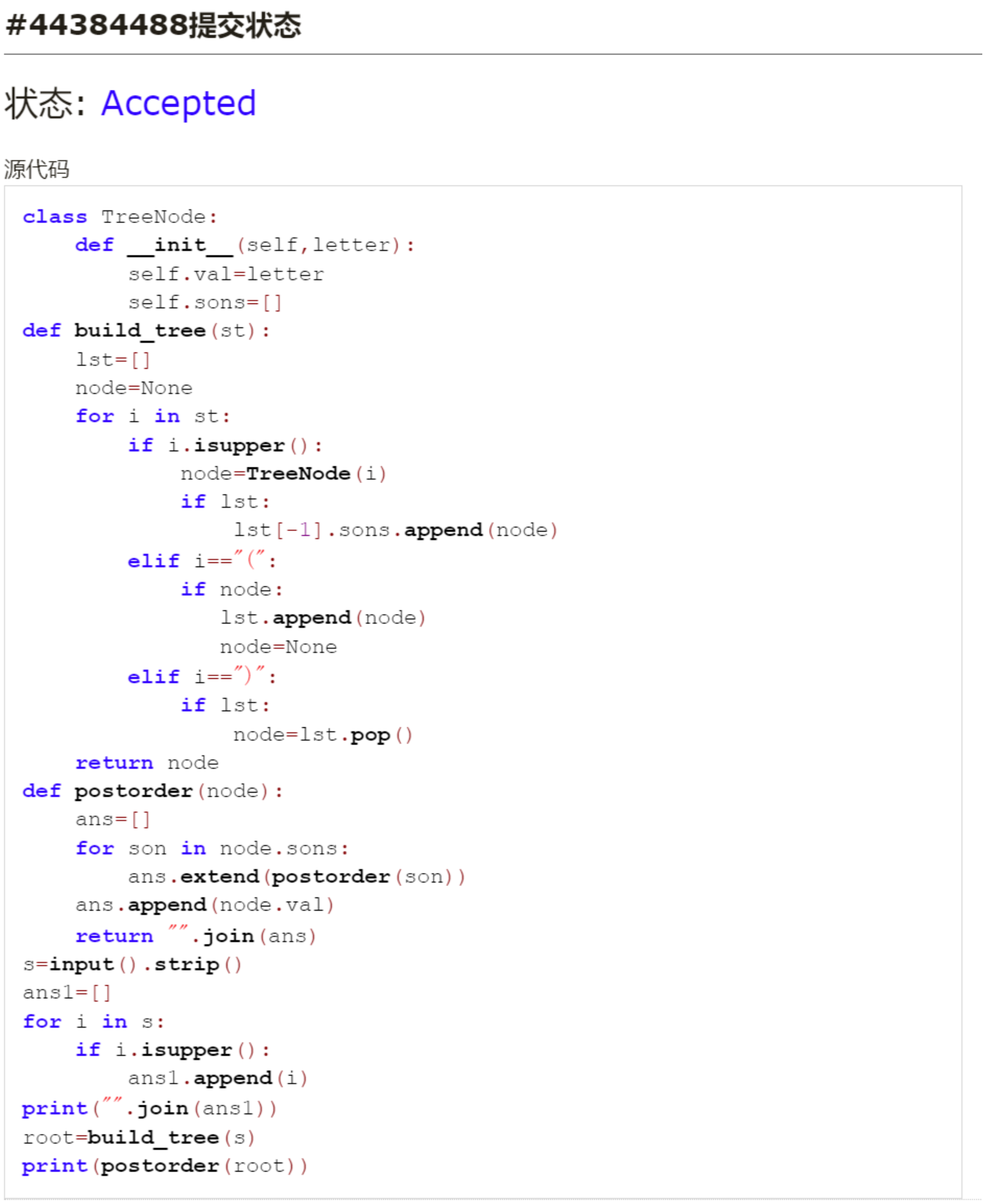
http://cs101.openjudge.cn/practice/24729/

思路:

代码

```python
#
class TreeNode:
    def __init__(self,letter):
        self.val=letter
        self.sons=[]
def build_tree(st):
    lst=[]
    node=None
    for i in st:
        if i.isupper():
            node=TreeNode(i)
            if lst:
                lst[-1].sons.append(node)
        elif i=="(":
            if node:
               lst.append(node)
               node=None
        elif i==")":
            if lst:
                node=lst.pop()
    return node
def postorder(node):
    ans=[]
    for son in node.sons:
        ans.extend(postorder(son))
    ans.append(node.val)
    return "".join(ans)
s=input().strip()
ans1=[]
for i in s:
    if i.isupper():
        ans1.append(i)
print("".join(ans1))
root=build_tree(s)
print(postorder(root))
```

代码运行截图 ==（至少包含有"Accepted"）==


#44384488提交状态

状态: Accepted

源代码

```
class TreeNode:
    def __init__(self,letter):
        self.val=letter
        self.sons=[]
def build_tree(st):
    lst=[]
    node=None
    for i in st:
        if i.isupper():
            node=TreeNode(i)
            if lst:
                lst[-1].sons.append(node)
        elif i=="(":
            if node:
                lst.append(node)
                node=None
        elif i==")":
            if lst:
                node=lst.pop()
    return node
def postorder(node):
    ans=[]
    for son in node.sons:
        ans.extend(postorder(son))
    ans.append(node.val)
    return "".join(ans)
s=input().strip()
ans1=[]
for i in s:
    if i.isupper():
        ans1.append(i)
print("".join(ans1))
root=build_tree(s)
print(postorder(root))
```

02775: 文件结构“图”

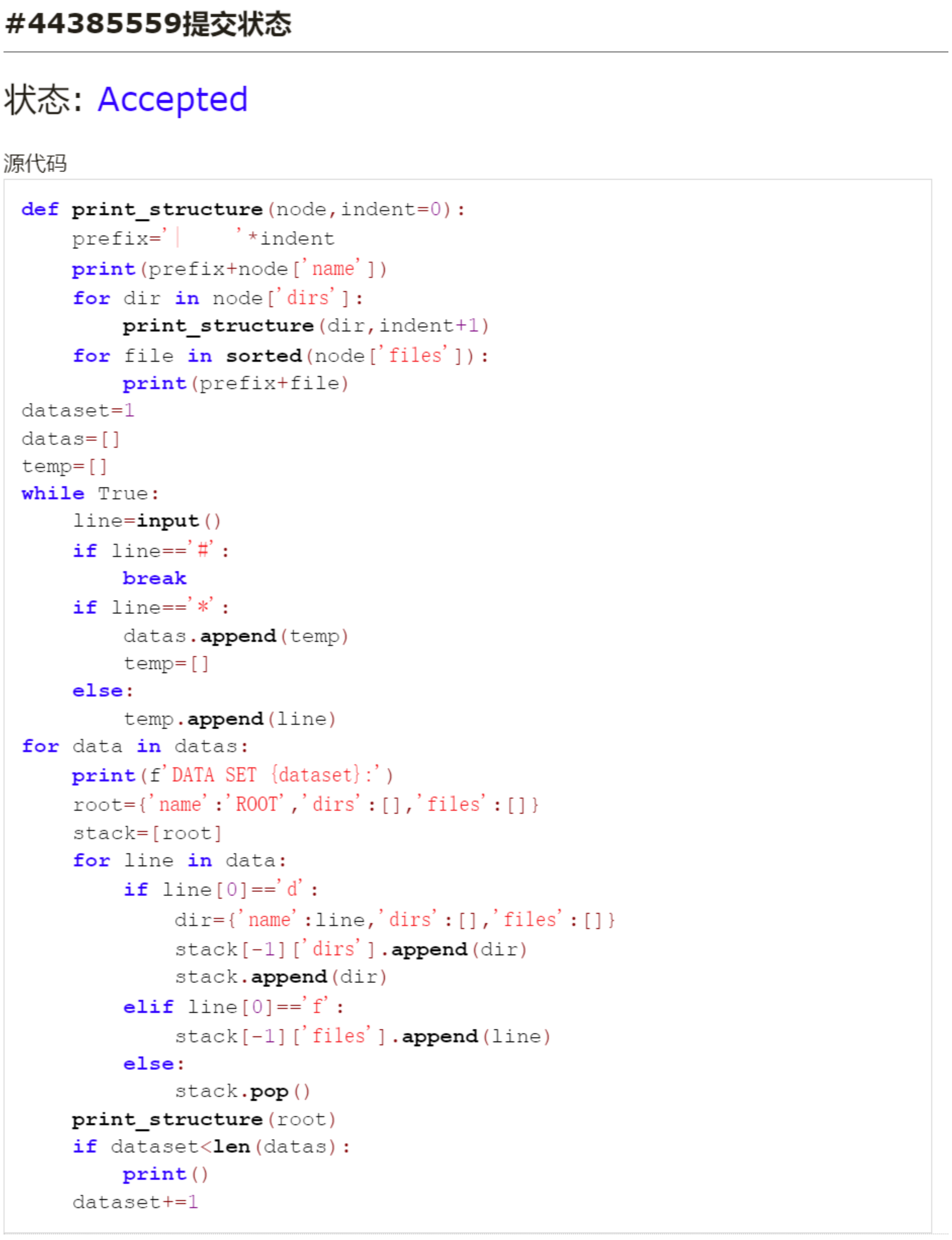
http://cs101.openjudge.cn/practice/02775/

思路:

代码

```
#
def print_structure(node,indent=0):
    prefix='|     '*indent
    print(prefix+node['name'])
    for dir in node['dirs']:
        print_structure(dir,indent+1)
    for file in sorted(node['files']):
        print(prefix+file)
dataset=1
datas=[]
temp=[]
while True:
    line=input()
    if line=='#':
        break
    if line=='*':
        datas.append(temp)
        temp=[]
    else:
        temp.append(line)
for data in datas:
    print(f'DATA SET {dataset}:')
    root={'name':'ROOT','dirs':[],'files':[]}
    stack=[root]
    for line in data:
        if line[0]=='d':
            dir={'name':line,'dirs':[],'files':[]}
            stack[-1]['dirs'].append(dir)
            stack.append(dir)
        elif line[0]=='f':
            stack[-1]['files'].append(line)
        else:
            stack.pop()
    print_structure(root)
    if dataset<len(datas):
        print()
    dataset+=1
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44385559提交状态

状态: Accepted

源代码

```
def print_structure(node,indent=0):
    prefix='|     '*indent
    print(prefix+node['name'])
    for dir in node['dirs']:
        print_structure(dir,indent+1)
    for file in sorted(node['files']):
        print(prefix+file)
dataset=1
datas=[]
temp=[]
while True:
    line=input()
    if line=='#':
        break
    if line=='*':
        datas.append(temp)
        temp=[]
    else:
        temp.append(line)
for data in datas:
    print(f'DATA SET {dataset}:')
    root={'name':'ROOT','dirs':[],'files':[]}
    stack=[root]
    for line in data:
        if line[0]=='d':
            dir={'name':line,'dirs':[],'files':[]}
            stack[-1]['dirs'].append(dir)
            stack.append(dir)
        elif line[0]=='f':
            stack[-1]['files'].append(line)
        else:
            stack.pop()
    print_structure(root)
    if dataset<len(datas):
        print()
    dataset+=1
```

25140: 根据后序表达式建立队列表达式

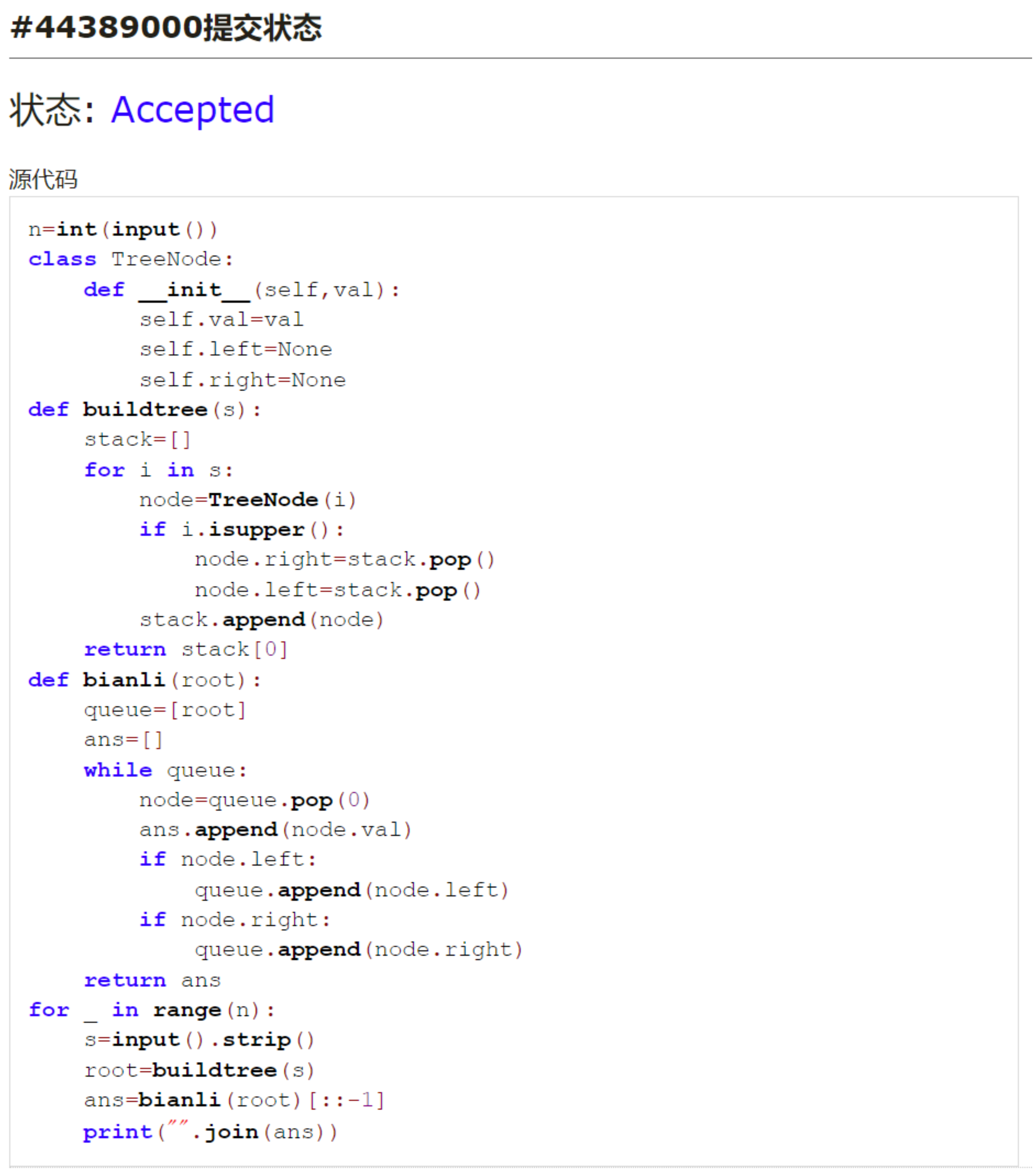
http://cs101.openjudge.cn/practice/25140/

思路：

代码

```python
#
n=int(input())
class TreeNode:
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
def buildtree(s):
    stack=[]
    for i in s:
        node=TreeNode(i)
        if i.isupper():
            node.right=stack.pop()
            node.left=stack.pop()
        stack.append(node)
    return stack[0]
def bianli(root):
    queue=[root]
    ans=[]
    while queue:
        node=queue.pop(0)
        ans.append(node.val)
        if node.left:
            queue.append(node.left)
        if node.right:
            queue.append(node.right)
    return ans
for _ in range(n):
    s=input().strip()
    root=buildtree(s)
    ans=bianli(root)[::-1]
    print("".join(ans))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44389000提交状态

状态: Accepted

源代码

```python
n=int(input())
class TreeNode:
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
def buildtree(s):
    stack=[]
    for i in s:
        node=TreeNode(i)
        if i.isupper():
            node.right=stack.pop()
            node.left=stack.pop()
        stack.append(node)
    return stack[0]
def bianli(root):
    queue=[root]
    ans=[]
    while queue:
        node=queue.pop(0)
        ans.append(node.val)
        if node.left:
            queue.append(node.left)
        if node.right:
            queue.append(node.right)
    return ans
for _ in range(n):
    s=input().strip()
    root=buildtree(s)
    ans=bianli(root)[::-1]
    print("".join(ans))
```

24750: 根据二叉树中后序序列建树

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

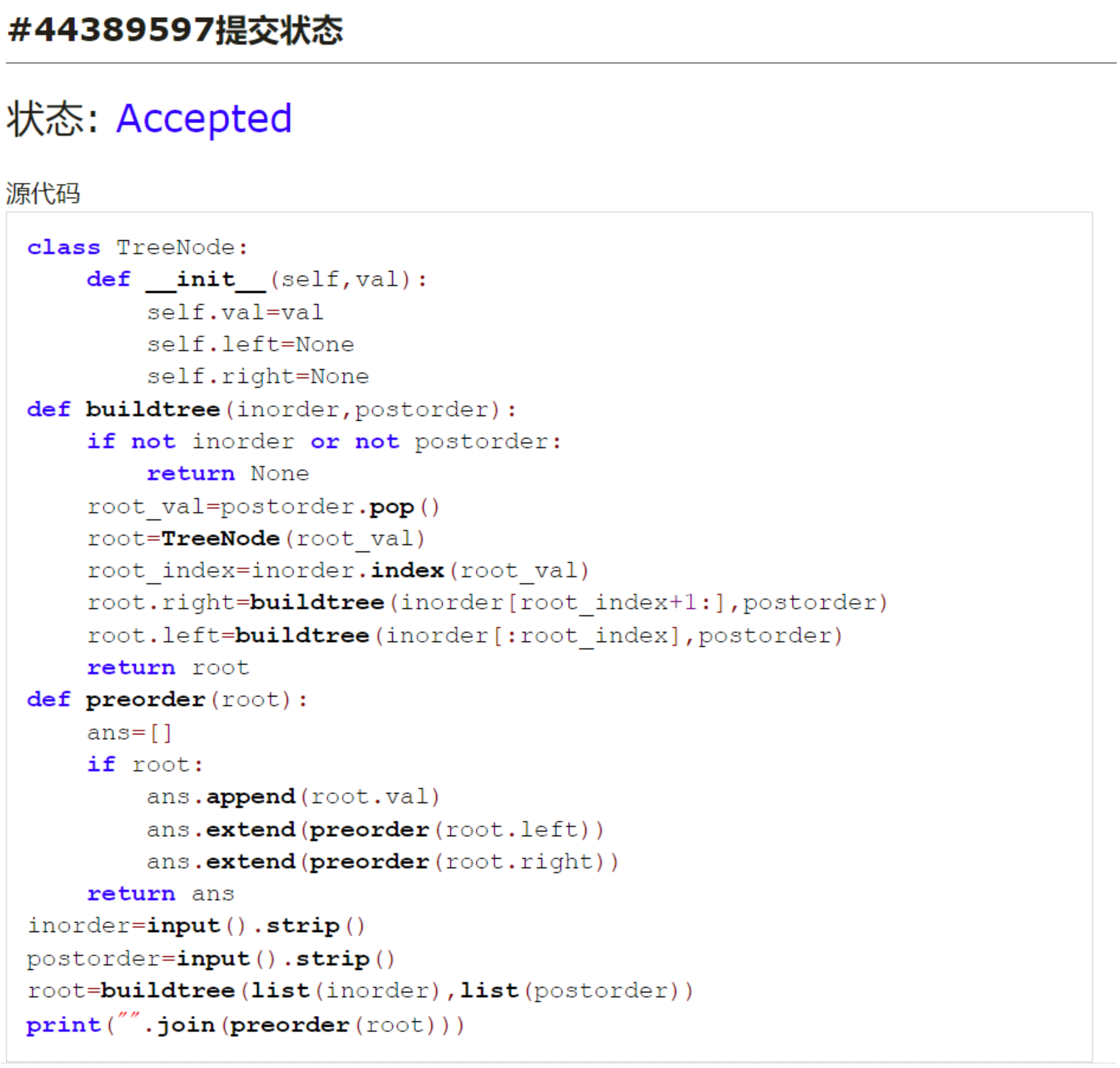
思路:

代码

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

```python
def buildtree(inorder,postorder):
    if not inorder or not postorder:
        return None
    root_val=postorder.pop()
    root=TreeNode(root_val)
    root_index=inorder.index(root_val)
    root.right=buildtree(inorder[root_index+1:],postorder)
    root.left=buildtree(inorder[:root_index],postorder)
    return root
def preorder(root):
    ans=[]
    if root:
        ans.append(root.val)
        ans.extend(preorder(root.left))
        ans.extend(preorder(root.right))
    return ans
inorder=input().strip()
postorder=input().strip()
root=buildtree(list(inorder),list(postorder))
print("".join(preorder(root)))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44389597提交状态

状态: Accepted

源代码

```python
class TreeNode:
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
def buildtree(inorder,postorder):
    if not inorder or not postorder:
        return None
    root_val=postorder.pop()
    root=TreeNode(root_val)
    root_index=inorder.index(root_val)
    root.right=buildtree(inorder[root_index+1:],postorder)
    root.left=buildtree(inorder[:root_index],postorder)
    return root
def preorder(root):
    ans=[]
    if root:
        ans.append(root.val)
        ans.extend(preorder(root.left))
        ans.extend(preorder(root.right))
    return ans
inorder=input().strip()
postorder=input().strip()
root=buildtree(list(inorder),list(postorder))
print("".join(preorder(root)))
```

22158: 根据二叉树前中序序列建树

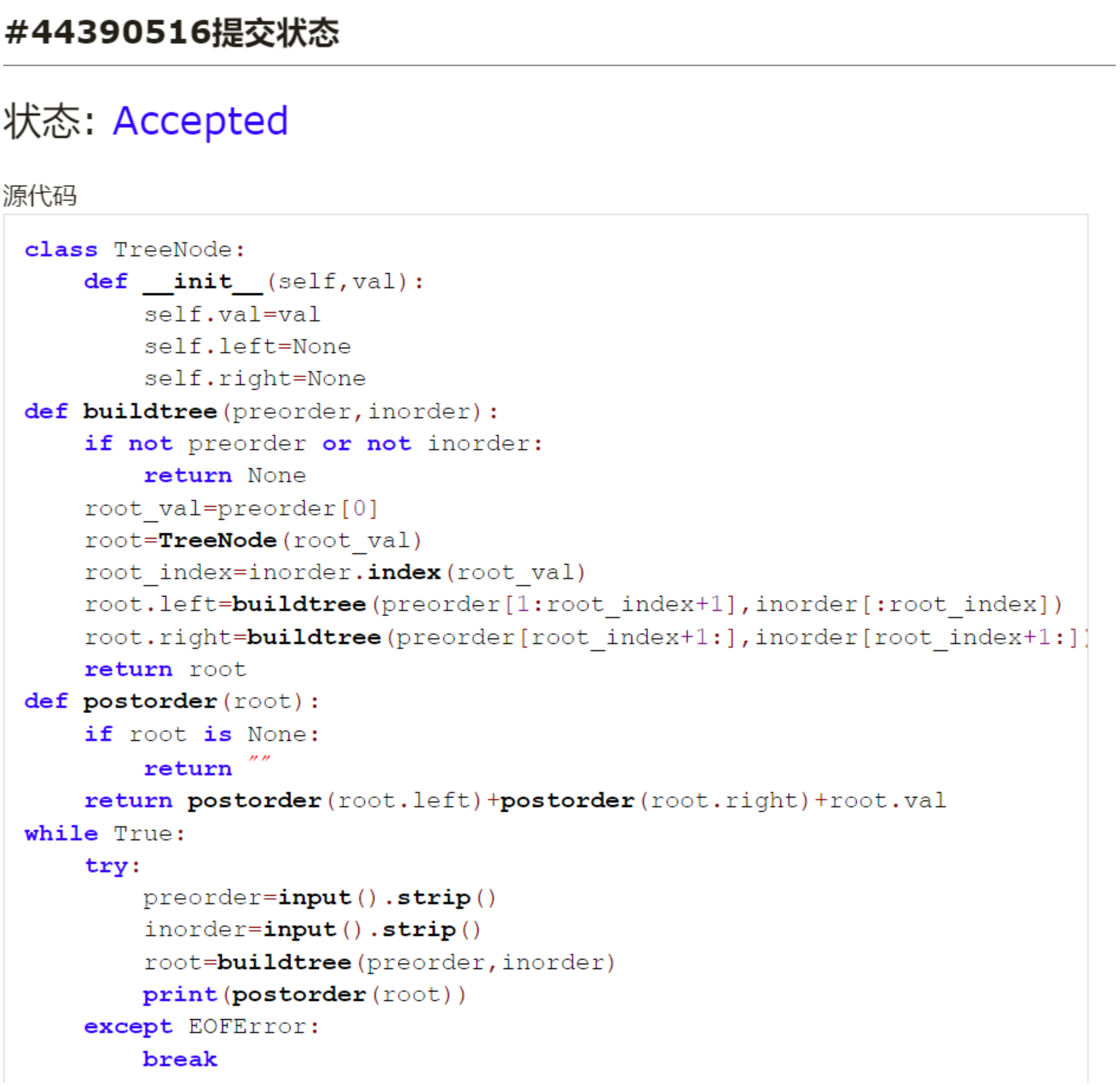
http://cs101.openjudge.cn/practice/22158/

思路：

代码

```python
#
class TreeNode:
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
def buildtree(preorder,inorder):
    if not preorder or not inorder:
        return None
    root_val=preorder[0]
    root=TreeNode(root_val)
    root_index=inorder.index(root_val)
    root.left=buildtree(preorder[1:root_index+1],inorder[:root_index])
    root.right=buildtree(preorder[root_index+1:],inorder[root_index+1:])
    return root
def postorder(root):
    if root is None:
        return ""
    return postorder(root.left)+postorder(root.right)+root.val
while True:
    try:
        preorder=input().strip()
        inorder=input().strip()
        root=buildtree(preorder,inorder)
        print(postorder(root))
    except EOFError:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==


#44390516提交状态

状态: Accepted

源代码

```
class TreeNode:
    def __init__(self,val):
        self.val=val
        self.left=None
        self.right=None
def buildtree(preorder,inorder):
    if not preorder or not inorder:
        return None
    root_val=preorder[0]
    root=TreeNode(root_val)
    root_index=inorder.index(root_val)
    root.left=buildtree(preorder[1:root_index+1],inorder[:root_index])
    root.right=buildtree(preorder[root_index+1:],inorder[root_index+1:])
    return root
def postorder(root):
    if root is None:
        return ""
    return postorder(root.left)+postorder(root.right)+root.val
while True:
    try:
        preorder=input().strip()
        inorder=input().strip()
        root=buildtree(preorder,inorder)
        print(postorder(root))
    except EOFError:
        break
```

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。==

经过这些题目的训练，我对于树（尤其是二叉树）相关的算法有了一些概念，后面做题应该会相对顺一些。