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==

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：macOS Ventura 13.4.1 (c)

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

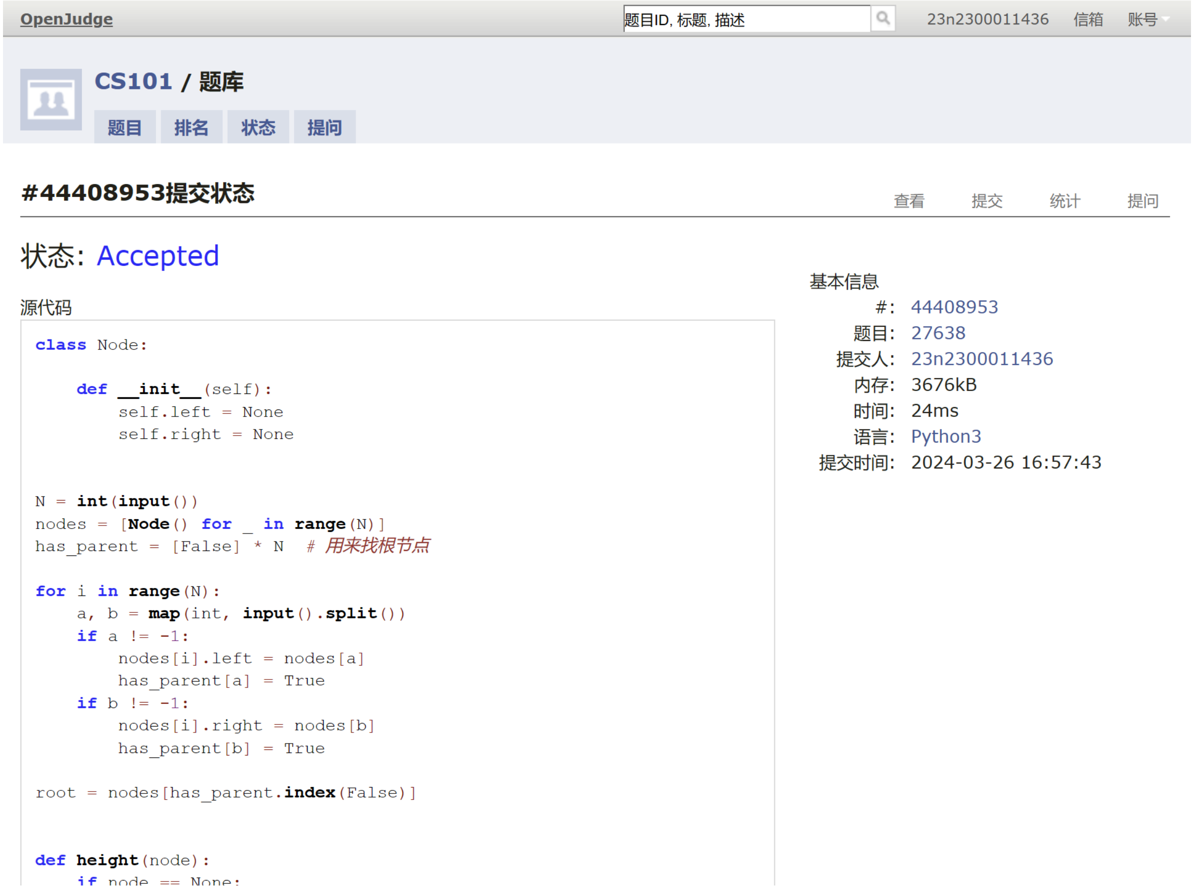
http://cs101.openjudge.cn/practice/27638/

思路：

代码

```
class Node:

    def __init__(self):
        self.left = None
        self.right = None


N = int(input())
nodes = [Node() for _ in range(N)]
has_parent = [False] * N  # 用来找根节点

for i in range(N):
    a, b = map(int, input().split())
    if a != -1:
        nodes[i].left = nodes[a]
        has_parent[a] = True
    if b != -1:
        nodes[i].right = nodes[b]
        has_parent[b] = True

root = nodes[has_parent.index(False)]


def height(node):
    if node == None:
        return -1
    return max(height(node.left), height(node.right)) + 1


def leaves(node):
    if node == None:
        return 0
    if node.left == None and node.right == None:
        return 1
    return leaves(node.left) + leaves(node.right)


print(height(root), leaves(root))
```

代码运行截图



#44408953提交状态

状态: Accepted

源代码

```
class Node:

    def __init__(self):
        self.left = None
        self.right = None


N = int(input())
nodes = [Node() for _ in range(N)]
has_parent = [False] * N  # 用来找根节点

for i in range(N):
    a, b = map(int, input().split())
    if a != -1:
        nodes[i].left = nodes[a]
        has_parent[a] = True
    if b != -1:
        nodes[i].right = nodes[b]
        has_parent[b] = True

root = nodes[has_parent.index(False)]


def height(node):
    if node == None:
```

基本信息

#:	44408953
题目:	27638
提交人:	23n2300011436
内存:	3676kB
时间:	24ms
语言:	Python3
提交时间:	2024-03-26 16:57:43

24729: 括号嵌套树

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

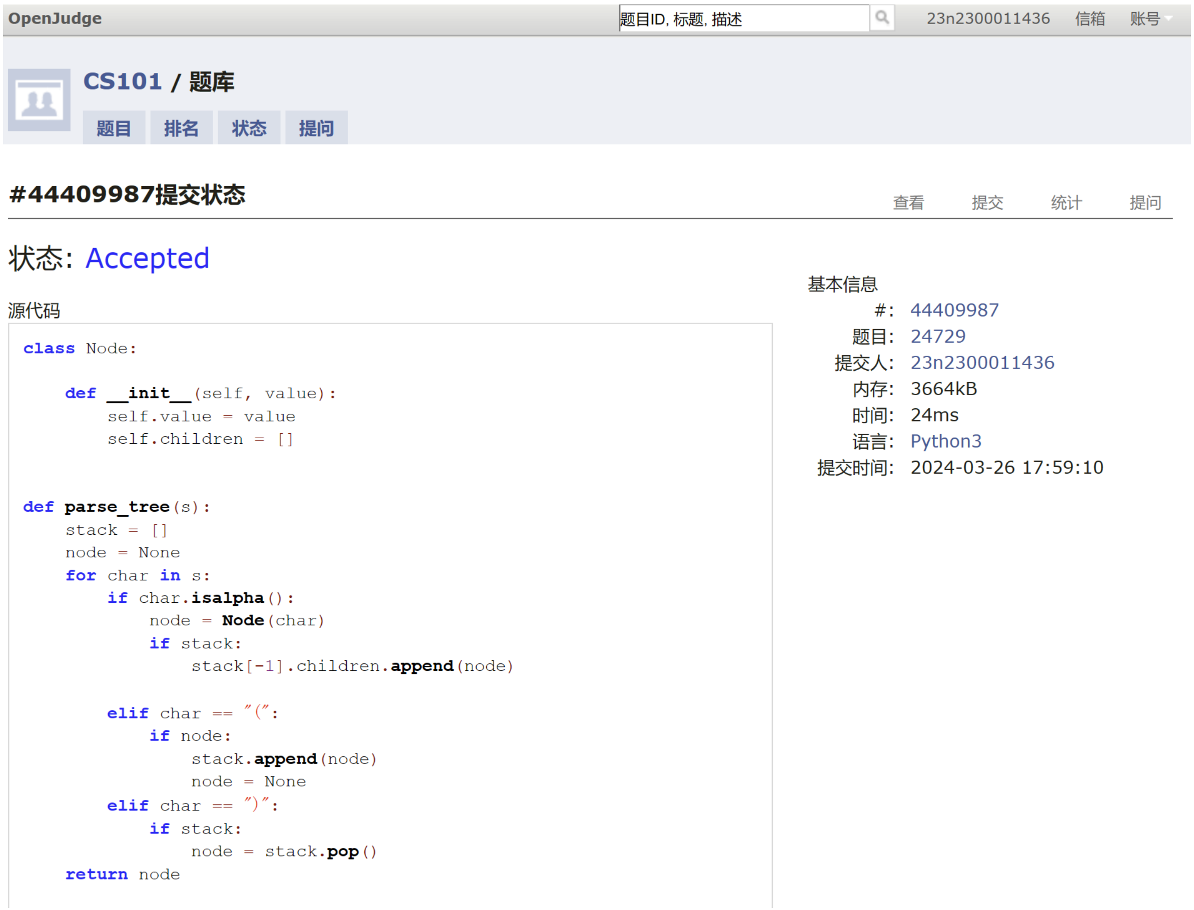
思路:

代码

```
class Node:

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


def parse_tree(s):
    stack = []
    node = None
    for char in s:
        if char.isalpha():
            node = Node(char)
            if stack:
                stack[-1].children.append(node)
```

```python
        elif char == "(":
            if node:
                stack.append(node)
                node = None
        elif char == ")":
            if stack:
                node = stack.pop()
    return node


def preorder(node):
    output = [node.value]
    for child in node.children:
        output += preorder(child)
    return ''.join(output)


def postorder(node):
    output = []
    for child in node.children:
        output += postorder(child)
    output += node.value
    return ''.join(output)


s = input()
print(preorder(parse_tree(s)))
print(postorder(parse_tree(s)))
```

代码运行截图



#44409987提交状态

状态: Accepted

源代码

```
class Node:

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


def parse_tree(s):
    stack = []
    node = None
    for char in s:
        if char.isalpha():
            node = Node(char)
            if stack:
                stack[-1].children.append(node)

        elif char == "(":
            if node:
                stack.append(node)
                node = None
        elif char == ")":
            if stack:
                node = stack.pop()
    return node
```

基本信息

#:	44409987
题目:	24729
提交人:	23n2300011436
内存:	3664kB
时间:	24ms
语言:	Python3
提交时间:	2024-03-26 17:59:10

02775: 文件结构“图”

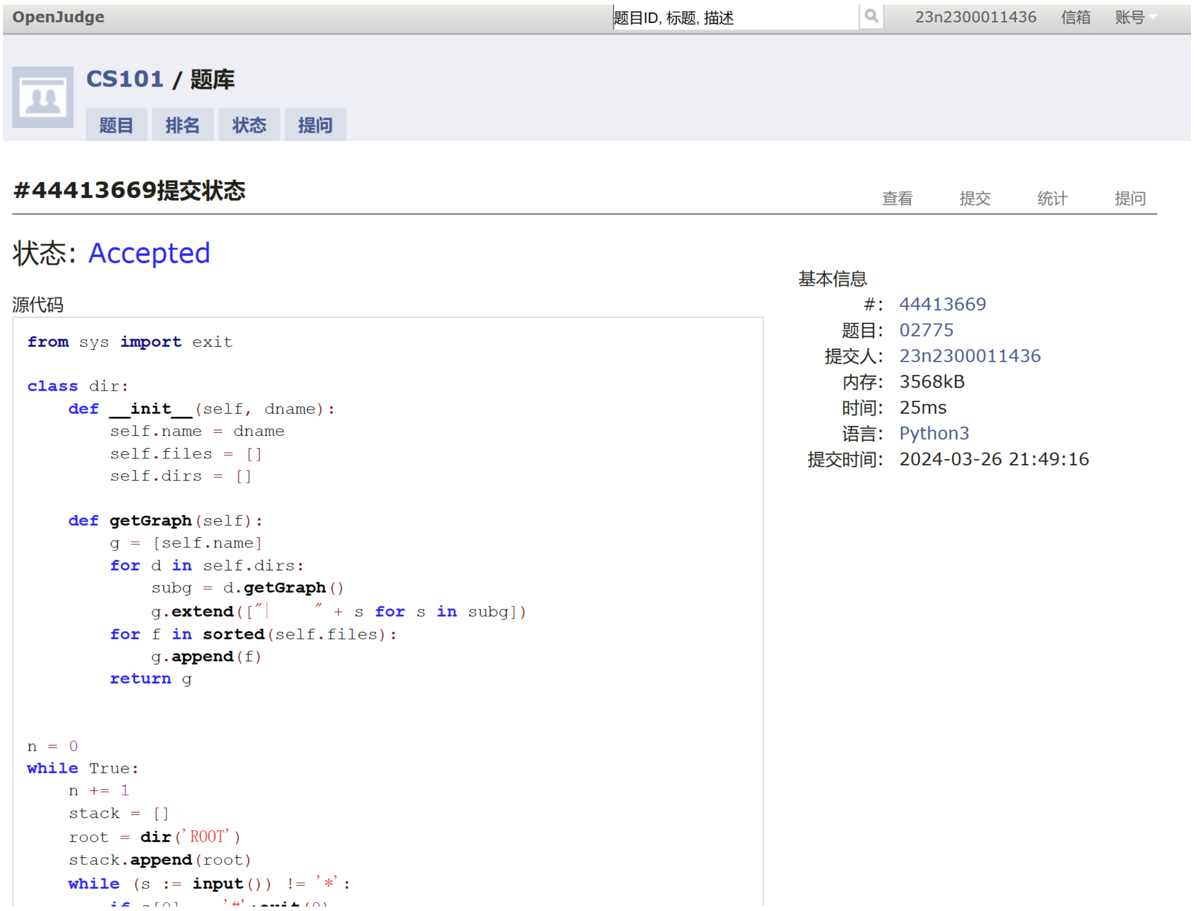
http://cs101.openjudge.cn/practice/02775/

思路:

代码

```
from sys import exit

class dir:
    def __init__(self, dname):
        self.name = dname
        self.files = []
        self.dirs = []

    def getGraph(self):
        g = [self.name]
        for d in self.dirs:
            subg = d.getGraph()
            g.extend(["|     " + s for s in subg])
        for f in sorted(self.files):
            g.append(f)
        return g


n = 0
while True:
    n += 1
    stack = []
    root = dir('ROOT')
    stack.append(root)
    while (s := input()) != '*':
        if s[0] == '#':exit(0)
        if s[0] == 'f':
            stack[-1].files.append(s)
        elif s[0] == 'd':
            node = dir(s)
            if stack:
                stack[-1].dirs.append(node)
                stack.append(node)
        elif s == ']':
            stack.pop()
    print(f"DATA SET {n}:")
    print(*stack[0].getGraph(), sep='\n')
    print()
```

代码运行截图



#44413669提交状态

状态: Accepted

源代码

```
from sys import exit

class dir:
    def __init__(self, dname):
        self.name = dname
        self.files = []
        self.dirs = []

    def getGraph(self):
        g = [self.name]
        for d in self.dirs:
            subg = d.getGraph()
            g.extend(["|     " + s for s in subg])
        for f in sorted(self.files):
            g.append(f)
        return g


n = 0
while True:
    n += 1
    stack = []
    root = dir('ROOT')
    stack.append(root)
    while (s := input()) != '*':
        if s[0] == '#':exit(0)
```

基本信息

#: 44413669
题目: 02775
提交人: 23n2300011436
内存: 3568kB
时间: 25ms
语言: Python3
提交时间: 2024-03-26 21:49:16

25140: 根据后序表达式建立队列表达式

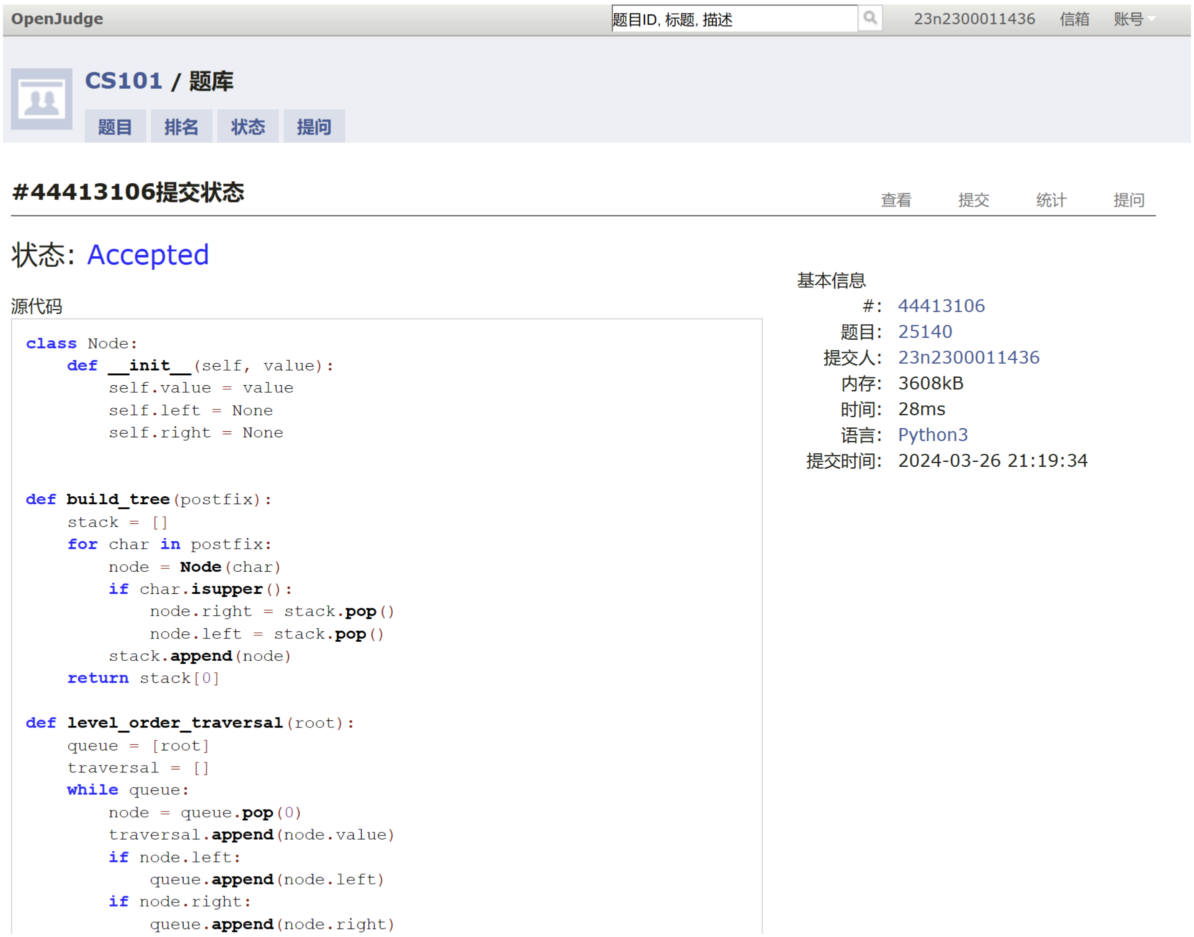
http://cs101.openjudge.cn/practice/25140/

思路：

代码

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


def build_tree(postfix):
    stack = []
    for char in postfix:
        node = Node(char)
        if char.isupper():
            node.right = stack.pop()
            node.left = stack.pop()
        stack.append(node)
    return stack[0]

def level_order_traversal(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

n = int(input())
for _ in range(n):
    s = input()
    root = build_tree(s)
    traversal = level_order_traversal(root)
    print(''.join(reversed(traversal)))
```

代码运行截图



#44413106提交状态

状态: Accepted

源代码

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


def build_tree(postfix):
    stack = []
    for char in postfix:
        node = Node(char)
        if char.isupper():
            node.right = stack.pop()
            node.left = stack.pop()
        stack.append(node)
    return stack[0]

def level_order_traversal(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)
```

基本信息

#:	44413106
题目:	25140
提交人:	23n2300011436
内存:	3608kB
时间:	28ms
语言:	Python3
提交时间:	2024-03-26 21:19:34

24750: 根据二叉树中后序序列建树

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

思路:

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

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

.isalpha()可以判断是不是字母

.isupper()和.islower()可以判断大小写

剩下两道看了答案的代码，之后再自己写一遍