Assignment #5: "树"算：概念、表示、解析、遍历

Updated 2124 GMT+8 March 17, 2024

2024 spring, Complied by 李鹏辉，元培学院

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

Windows 10 Home, PyCharm 2022.3.2 (Community Edition)

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

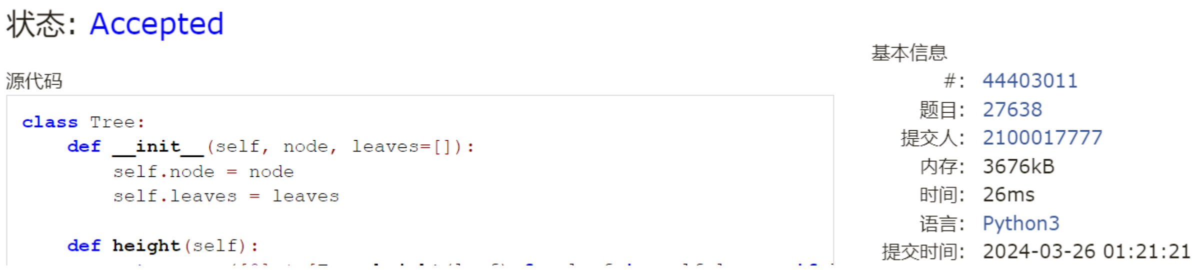
http://cs101.openjudge.cn/practice/27638/

思路：30分钟。主要是在纠结如何设置一个通用的Tree使得后面非二叉的、node值能自由定义的树也能用到，这就要求把求高度写成 a method of class Tree，这里耗费了一点时间。

代码

```python
class Tree:
    def __init__(self, node, leaves=[]):
        self.node = node
        self.leaves = leaves

    def height(self):
        return max([0] + [Tree.height(leaf) for leaf in self.leaves if leaf is not None]) + 1


def q1():
    n = int(input())
    if n == 0:
        print('-1 0')
        return
    leaf_nodes = 0
    parents = [False for _ in range(n)] + [True]
    nodes = [Tree(i) for i in range(n)] + [Tree(-1)]

    for i in range(n):
        left, right = map(int, input().split())
        leaves = [None, None]
        if (left, right) == (-1, -1):
            leaf_nodes += 1
        if left != -1:
            parents[left] = True
            leaves[0] = nodes[left]
        if right != -1:
            parents[right] = True
            leaves[1] = nodes[right]
        nodes[i].leaves = leaves

    root = parents.index(False)
    print(f'{Tree.height(nodes[root])-1} {leaf_nodes}')
    return


q1()
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```python
class Tree:
    def __init__(self, node, leaves=[]):
        self.node = node
        self.leaves = leaves

    def height(self):
```

基本信息

#:	44403011
题目:	27638
提交人:	2100017777
内存:	3676kB
时间:	26ms
语言:	Python3
提交时间:	2024-03-26 01:21:21

24729: 括号嵌套树

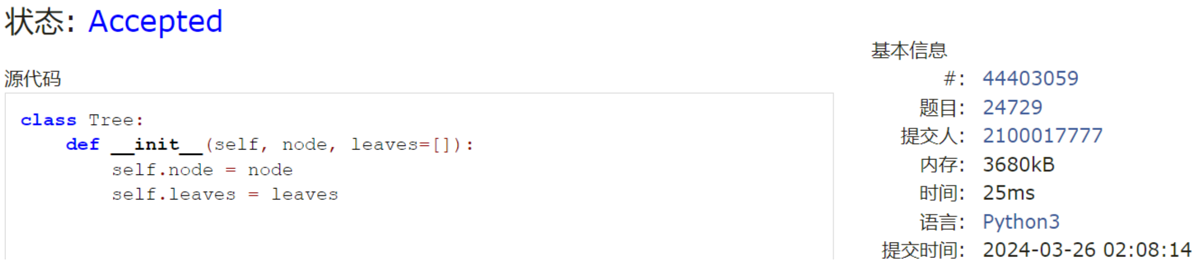
http://cs101.openjudge.cn/practice/24729/

思路：40分钟。

代码

```python
class Tree:
    def __init__(self, node, leaves=[]):
        self.node = node
        self.leaves = leaves


def q2():
    def parse():
        s = []
        nodes = []
        raw = input()
        for c in raw:  # character
            if c not in ['(', ')', ',']:
                s.append(Tree(c))
            elif c != ')':
                s.append(c)
            else:
                l = []  # leaves
                while len(s) > 2 and s[-1] != '(':
                    top = s.pop()
                    if top != ',':
                        l.append(top)
                s.pop()  # '('
                node = s.pop()
                node.leaves = l[::-1]
                s.append(node)
        return s[0]

    def preorder(t):
        pre = t.node
        for leaf in t.leaves:
            pre += preorder(leaf)
        return pre

    def postorder(t):
        post = ''
        for leaf in t.leaves:
            post += postorder(leaf)
        post += t.node
        return post

    t = parse()
    print(preorder(t))
    print(postorder(t))
    return


q2()
```

代码运行截图（至少包含有"Accepted"）



状态: Accepted

源代码

```
class Tree:
    def __init__(self, node, leaves=[]):
        self.node = node
        self.leaves = leaves
```

基本信息

#: 44403059
题目: 24729
提交人: 2100017777
内存: 3680kB
时间: 25ms
语言: Python3
提交时间: 2024-03-26 02:08:14

02775: 文件结构“图”

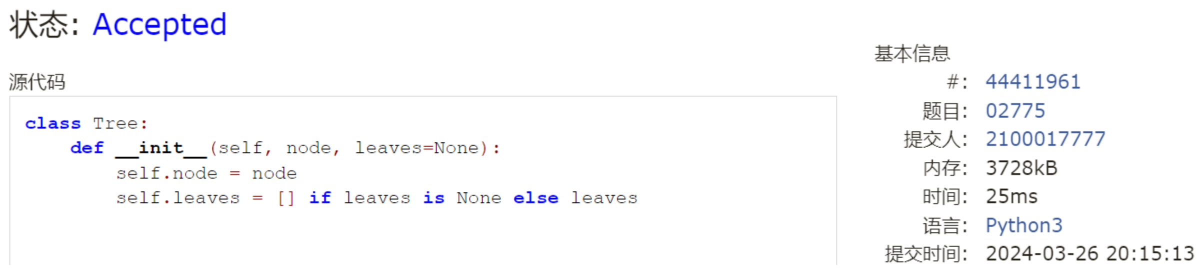
http://cs101.openjudge.cn/practice/02775/

思路：不记得做了多久了……大概三个小时吧，反正每个函数都出过错，而且由于不方便拆开单独检查，所以只能先全部写完再一步步检查，尤其是检查出来class Tree不能将空列表作为默认参数输入花了很久时间。

代码

```python
class Tree:
    def __init__(self, node, leaves=None):
        self.node = node
        self.leaves = [] if leaves is None else leaves


class File(Tree):
    def __init__(self, node, leaves=None):
        super().__init__(node, leaves)
        self.depth = 0


class Dir(Tree):
    def __init__(self, node, leaves=None):
        super().__init__(node, leaves)
        self.depth = 0


def q3():
    re = ''  # result
    ds_num = 1
    s = [Dir('ROOT')]

    def d_c(t, p_d=0):  # depth_calculate; parent_depth
        for leaf in t.leaves:
            c_d = 1 + p_d  # current_depth
            leaf.depth += c_d
            d_c(leaf, c_d)

    def printer(t):
        nonlocal re
        re += '|     ' * t.depth + t.node + '\n'
        for leaf in t.leaves:
            printer(leaf)

    while True:
        inp = input()
        if inp == '#':
            print(re[:-1])
            return

        if inp == '*':
            re += f'DATA SET {ds_num}:\n'
            l_d = [leaf for leaf in s if isinstance(leaf, Dir)][::-1]
            l_f = [leaf for leaf in s if isinstance(leaf, File)][::-1]
            s = l_d + sorted(l_f, key=lambda x: x.node)
            for t in s:
                d_c(t)
            for t in s:
                printer(t)
            re += '\n'
            ds_num += 1
            s = [Dir('ROOT')]

        else:
            if inp[0] == 'f':
                s.append(File(inp))
            if inp[0] == 'd':
                s.append(Dir(inp))
            if inp == ']':
                l = []
                while not isinstance(s[-1], Dir):
                    l.append(s.pop())
                l.append(s.pop())
                for i in range(len(s)-1, -1, -1):
                    if isinstance(s[i], Dir):
                        l_d = [leaf for leaf in l if isinstance(leaf, Dir)][::-1]
                        l_f = [leaf for leaf in l if isinstance(leaf, File)][::-1]
                        s[i].leaves += l_d + sorted(l_f, key=lambda x: x.node)
                        break


q3()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```
class Tree:
    def __init__(self, node, leaves=None):
        self.node = node
        self.leaves = [] if leaves is None else leaves
```

基本信息

#: 44411961
题目: 02775
提交人: 2100017777
内存: 3728kB
时间: 25ms
语言: Python3
提交时间: 2024-03-26 20:15:13

25140: 根据后序表达式建立队列表达式

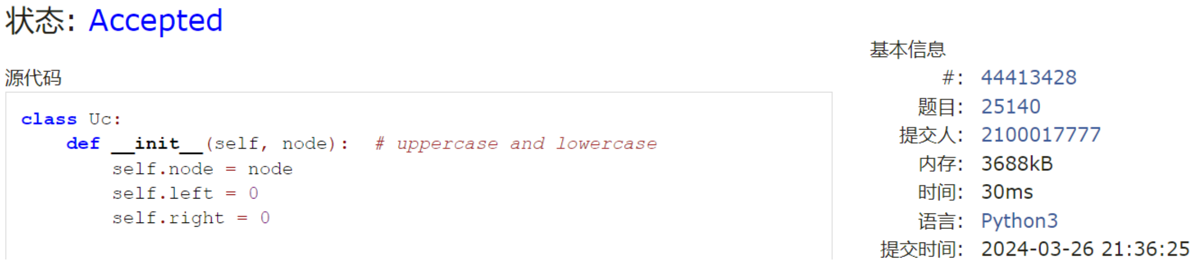
http://cs101.openjudge.cn/practice/25140/

思路：

代码

```python
class Uc:
    def __init__(self, node):  # uppercase and lowercase
        self.node = node
        self.left = 0
        self.right = 0

    @property
    def filled(self):
        return self.left != 0 and self.right != 0


class Lc:
    def __init__(self, node):
        self.node = node
        self.filled = True


def q4():
    def convert(pi):  # post_input
        pi = pi[::-1]
        s = []
        for i in pi:
            if 'A' <= i <= 'Z':
                s.append(Uc(i))
            else:
                s.append(Lc(i))
                while len(s) > 1 and s[-1].filled and s[-2].filled:
                    second = s.pop()
                    first = s.pop()
                    operator = s.pop()
                    operator.left, operator.right = first, second
                    s.append(operator)

        t = s[0]
        results = []
        leaves = [t]
        while leaves:
            next_lvs = []
            re = ''
            for c_t in leaves:
                re += c_t.node
                if isinstance(c_t, Uc):
                    next_lvs.append(c_t.left)
                    next_lvs.append(c_t.right)
            results.append(re)
            leaves = next_lvs[:]
        print(''.join(results[::-1]))

    for _ in range(int(input())):
        convert(input())


q4()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```
class Uc:
    def __init__(self, node):  # uppercase and lowercase
        self.node = node
        self.left = 0
        self.right = 0
```

基本信息

#: 44413428

题目: 25140

提交人: 2100017777

内存: 3688kB

时间: 30ms

语言: Python3

提交时间: 2024-03-26 21:36:25

24750: 根据二叉树中后序序列建树

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

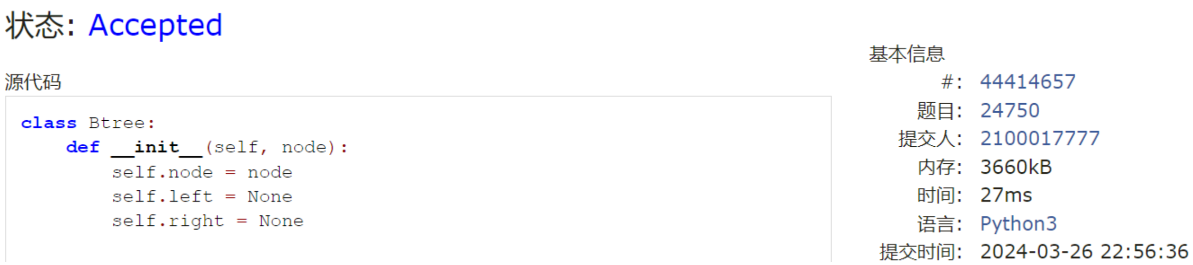
思路：30分钟。

代码

```
class Btree:
    def __init__(self, node):
        self.node = node
        self.left = None
        self.right = None


def q5():
    def build(ino, post):
        if not ino and not post:
            return None

        root = post[-1]
        root_idx = ino.index(root)
        tree = Btree(root)
        left_ino = ino[:root_idx]
        tree.left = build(left_ino, post[:len(left_ino)])
        tree.right = build(ino[root_idx+1:], post[len(left_ino):-1])
        return tree

```

```python
    re = ''

    def pre(t):
        nonlocal re
        if t:
            re += t.node
            pre(t.left)
            pre(t.right)

    ino = input()
    post = input()
    pre(build(ino, post))
    print(re)
    return


q5()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```python
class Btree:
    def __init__(self, node):
        self.node = node
        self.left = None
        self.right = None
```

基本信息

#: 44414657
题目: 24750
提交人: 2100017777
内存: 3660kB
时间: 27ms
语言: Python3
提交时间: 2024-03-26 22:56:36

22158: 根据二叉树前中序序列建树

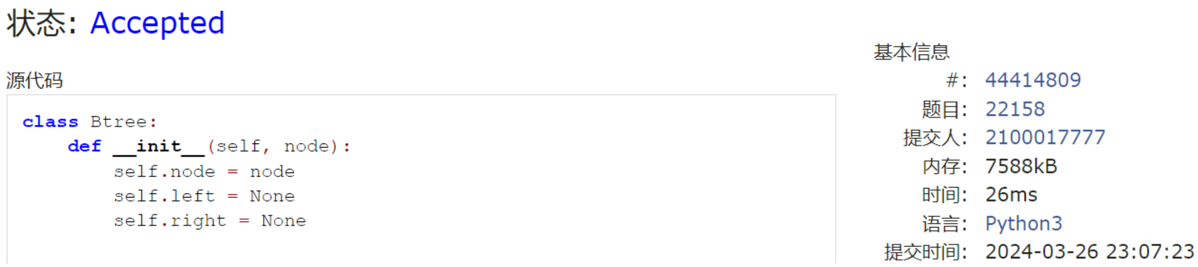
http://cs101.openjudge.cn/practice/22158/

思路:

代码

```python
class Btree:
    def __init__(self, node):
        self.node = node
        self.left = None
        self.right = None


def q6():
    def build(ino, pre):
        if not ino and not pre:
            return None

        root = pre[0]
        root_idx = ino.index(root)
        tree = Btree(root)
        left_ino = ino[:root_idx]
        tree.left = build(left_ino, pre[1:len(left_ino)+1])
        tree.right = build(ino[root_idx+1:], pre[len(left_ino)+1:])
        return tree

    re = ''

    def post(t):
        nonlocal re
        if t:
            post(t.left)
            post(t.right)
            re += t.node


    while True:
        try:
            pre = input()
            if not pre: return
            ino = input()
            post(build(ino, pre))
            print(re)
            re = ''
        except EOFError:
            return


q6()
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）



状态: Accepted

源代码

```
class Btree:
    def __init__(self, node):
        self.node = node
        self.left = None
        self.right = None
```

基本信息

#: 44414809
题目: 22158
提交人: 2100017777
内存: 7588kB
时间: 26ms
语言: Python3
提交时间: 2024-03-26 23:07:23

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。

文件结构图一题对处理实际应用问题时的抽象能力有相当高的要求，解读出dir与file的不同地位并设置成不同的类属性，而非照搬全部设置成子节点的思维旧式，才是最合适的方法。