11 第四讲 作业记录（一）

笔记本： 浙江大学《数据结构》
创建时间： 2025/4/6 16:13 **更新时间：** 2025/4/6 16:39
作者： panhengye@163.com
URL： https://pintia.cn/problem-sets/1873565885118418944/exam/problems/type/7?...

04-树4 是否同一棵二叉搜索树

提交结果

题目	用户	提交时间
04-树4	飞翔的小师弟	2025/04/06 16:12:45
编译器	内存	用时
Python (python3)	3228 / 65536 KB	18 / 400 ms
状态	分数	评测时间
答案正确	25 / 25	2025/04/06 16:12:46

评测详情

测试点	提示	内存(KB)	用时(ms)	结果	得分
0	sample 换顺序。有Yes，有No：根不同，子树根不同。树有单边、有双子树	3008	15	答案正确	12 / 12
1	最大N，多组合	2952	16	答案正确	8 / 8
2	N=1，只有1个节点	3140	18	答案正确	3 / 3
3	卡只判断数字相对先后位置的错误算法	3228	15	答案正确	2 / 2

【代码记录】

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


class BinarySearchTree:
    def __init__(self):
        self.root = None


    def insert(self, value):
        if self.root is None:
            self.root = TreeNode(value)
        else:
            self._insert_recursive(self.root, value)
```

```
    def _insert_recursive(self, node, value):
        if value < node.value:
            if node.left is None:
                node.left = TreeNode(value)
            else:
                self._insert_recursive(node.left, value)
        else:
            if node.right is None:
                node.right = TreeNode(value)
            else:
                self._insert_recursive(node.right, value)


def is_same_tree(tree1, tree2):
    if tree1 is None and tree2 is None:
        return True
    if tree1 is None or tree2 is None:
        return False
    if tree1.value != tree2.value:
        return False
    return is_same_tree(tree1.left, tree2.left) and is_same_tree(tree1.right,
tree2.right)


def is_same_bst(seq1, seq2):
    # 创建两棵树
    tree1 = BinarySearchTree()
    tree2 = BinarySearchTree()

    # 按照给定的序列插入节点
    for value in seq1:
        tree1.insert(value)

    for value in seq2:
        tree2.insert(value)

    # 比较两棵树是否相同
    return is_same_tree(tree1.root, tree2.root)


def main():
    while True:
        first_line = input().strip().split()

        # 检查输入是否只有一个数字0
        if len(first_line) == 1 and int(first_line[0]) == 0:
            break

        n, l = int(first_line[0]), int(first_line[1])

        # 读取初始序列
        initial_seq = list(map(int, input().strip().split()))

        # 处理每个需要检查的序列
        for _ in range(l):
            test_seq = list(map(int, input().strip().split()))

            # 判断测试序列与初始序列是否生成相同的BST
            if is_same_bst(initial_seq, test_seq):
                print("Yes")
            else:
                print("No")


if __name__ == "__main__":
    main()
```

【整体思路】

在《小白专场:是否同一棵二叉搜索树- C实现》中，老师提到了对比树有三种方法：建两棵树、不建树、建一棵树。这里采用建两棵树的方法：

1. **数据结构设计**：
   - `TreeNode`类表示二叉树节点
   - `BinarySearchTree`类实现了BST的基本操作
2. **核心算法**：
   - `is_same_tree`函数：递归比较两个树结构是否完全一致
   - `is_same_bst`函数：根据输入序列构建树并比较结果

【关键点解析1】

```
        first_line = input().strip().split()

        # 检查输入是否只有一个数字0
        if len(first_line) == 1 and int(first_line[0]) == 0:
            break

        n, l = int(first_line[0]), int(first_line[1])
```

读入数据这一步容易错，该脚本的思路是：

- “主要信息”行有两种情况，一种是 N（结点数），L（待比较的数量）；另一种是0，即代表程序结束
- 我们用一个字符串来装，然后根据数量和值来判断它属于上述哪一种情况
  - 如果字符串长度为1且值为0，那么程序直接结束
  - 否则，将字符串中第一个项、第二个项存入变量