题目 二叉搜索树与双向链表

考点 分解让复杂问题简单 热点指数 40641 通过率 27.70%

具体题目

输入一棵二叉搜索树，将该二叉搜索树转换成一个排序的双向链表。要求不能创建任何新的结点，只能调整树中结点指针的指向。

```
/** 递归：中序遍历 右 中 左 */
public class Solution {
    TreeNode list = null;
    public TreeNode Convert(TreeNode pRootOfTree) {
        if(pRootOfTree == null) return pRootOfTree;

        Convert(pRootOfTree.right);
        if(list == null){
            list = pRootOfTree;
        } else {
            list.left = pRootOfTree;
            pRootOfTree.right = list;
            list = pRootOfTree;
        }
        Convert(pRootOfTree.left);

        return list;
    }
}
```

```
/** 非递归 */
import java.util.Stack;
public class Solution {
    public TreeNode Convert(TreeNode pRootOfTree) {
        if(pRootOfTree == null) return pRootOfTree;

        TreeNode list = null;
        Stack<TreeNode> s = new Stack<>();
        while(pRootOfTree != null || !s.isEmpty()){
            if(pRootOfTree != null) {
                s.push(pRootOfTree);
                pRootOfTree = pRootOfTree.right;
            } else {
                pRootOfTree = s.pop();
                if(list == null)
                    list = pRootOfTree;
                else {
                    list.left = pRootOfTree;
                    pRootOfTree.right = list;
                    list = pRootOfTree;
                }
                pRootOfTree = pRootOfTree.left;
            }
        }

        return list;
    }
}
```

我认为最高赞答案的代码有些冗余，这里采用的是和剑指offer完全一样的思想。 明确Convert函数的功能。输入：输入一个二叉搜索树的根节点。过程：将其转化为一个有序的双向链表。输出：返回该链表的头节点。 明确成员变量pLast的功能。pLast用于记录当前链表的末尾节点。 明确递归过程。递归的过程就相当于按照中序遍历，将整个树分解成了无数的小树，然后将他们分别转化成了一小段一小段的双向链表。再利用pLast记录总的链表的末尾，然后将这些小段链表一个接一个地加到末尾。

```
private TreeNode pLast = null;
public TreeNode Convert(TreeNode root) {
 if (root == null)
     return null;
 // 如果左子树为空，那么根节点root为双向链表的头节点
 TreeNode head = Convert(root.left);
 if (head == null)
     head = root;
 // 连接当前节点root和当前链表的尾节点pLast
 root.left = pLast;
 if (pLast != null)
     pLast.right = root;
 pLast = root;
 Convert(root.right);
 return head;
}
```