剑指 Offer 68 - I. 二叉搜索树的最近公共祖先

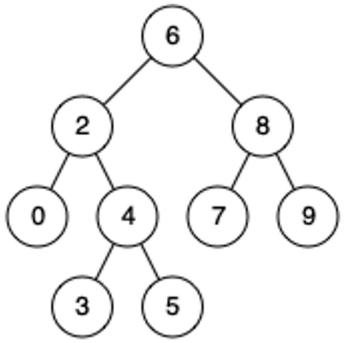
Wednesday, June 29, 2022 12:19 PM

https://leetcode.cn/problems/er-cha-sou-suo-shu-de-zui-jin-gong-gong-zu-xian-lcof/solution/mian-shi-ti-68-i-er-cha-sou-suo-shu-de-zui-jin-g-7/

简单

给定一个二叉搜索树, 找到该树中两个指定节点的最近公共祖先。

例如，给定如下二叉搜索树: root = [6,2,8,0,4,7,9,null,null,3,5]



示例 1:

输入: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 8

输出: 6

解释: 节点 2 和节点 8 的最近公共祖先是 6。

示例 2:

输入: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 4

输出: 2

解释: 节点 2 和节点 4 的最近公共祖先是 2, 因为根据定义最近公共祖先节点可以为节点本身。

思路:

二叉搜索数特点是。中序遍历为有序序列，其中每个节点值小于其右节点的值，大于其左节点的值，且所有节点均唯一。

确定pq在不在当前节点的同一子树下即可，若在则继续遍历，不在则返回此时节点。

解法一：递归 o(N) o(N)

```
class Solution {
    public TreeNode lowestCommonAncestor(TreeNode root, TreeNode p, TreeNode
q) {
        // 此时pq均在root右子树中 以右子节点为父继续遍历
        if(root.val < p.val && root.val < q.val)
            return lowestCommonAncestor(root.right, p, q);
        // 此时pq均在root左子树中 以左子节点为父继续遍历
        if(root.val > p.val && root.val > q.val)
```

```
            return lowestCommonAncestor(root.left, p, q);
        // 此时pq分别在root左右子树或与root重合，返回root为最近公共祖先
        return root;
    }
}
```

解法二：迭代 o(N) o(1)

```
class Solution {
    public TreeNode lowestCommonAncestor(TreeNode root, TreeNode p, TreeNode
q) {
        while(root != null) {
            if(root.val < p.val && root.val < q.val) // p,q 都在 root 的右子树中
                root = root.right; // 遍历至右子节点
            else if(root.val > p.val && root.val > q.val) // p,q 都在 root 的左子树中
                root = root.left; // 遍历至左子节点
            else break;
        }
        return root;
    }
}
```