Binary Tree Maximum Path Sum

Solution

Given a **non-empty** binary tree, find the maximum path sum.

For this problem, a path is defined as any sequence of nodes from some starting node to any node in the tree along the parent-child connections. The path must contain **at least one node** and does not need to go through the root.

Example 1:

```
Input: [1,2,3]

       1
      / \
     2   3

Output: 6
```

Example 2:

```
Input: [-10,9,20,null,null,15,7]

   -10
   / \
  9  20
    /  \
   15   7

Output: 42
```

Java

```java
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode() {}
 *     TreeNode(int val) { this.val = val; }
 *     TreeNode(int val, TreeNode left, TreeNode right) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
class Solution {
    int maxValue = Integer.MIN_VALUE;
    public int maxPathSum(TreeNode root) {
        helper(root);
        return maxValue;
    }

    private int helper(TreeNode root) {
        if(root == null) return Integer.MIN_VALUE;

        int total = root.val;
        int left = helper(root.left);
        int right = helper(root.right);

        int max = Math.max(left, right);
        if (max > 0) {
            total += max;
        }

        int maxPath = root.val + ((left > 0) ? left : 0) + ((right > 0) ? right : 0);

        maxValue = Math.max(maxValue, maxPath);
        return total;
    }
}
```

Custom Testcase (Contribute)

Run Code Submit

Submission Result: Accepted

More Details

Share your acceptance!

8