

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:

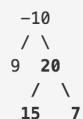
Input: [1,2,3]



Output: 6

Example 2:

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



Output: 42

Java

```

1 /**
2  * Definition for a binary tree node.
3  * public class TreeNode {
4  *     int val;
5  *     TreeNode left;
6  *     TreeNode right;
7  *     TreeNode() {}
8  *     TreeNode(int val) { this.val = val; }
9  *     TreeNode(int val, TreeNode left, TreeNode right) {
10 *         this.val = val;
11 *         this.left = left;
12 *         this.right = right;
13 *     }
14 * }
15 */
16 class Solution {
17     int maxValue = Integer.MIN_VALUE;
18     public int maxPathSum(TreeNode root) {
19         helper(root);
20         return maxValue;
21     }
22
23     private int helper(TreeNode root) {
24         if(root == null) return Integer.MIN_VALUE;
25
26         int total = root.val;
27         int left = helper(root.left);
28         int right = helper(root.right);
29
30         int max = Math.max(left, right);
31         if (max > 0) {
32             total += max;
33         }
34
35         int maxPath = root.val + ((left > 0) ? left : 0) + ((right > 0) ? right : 0);
36
37         maxValue = Math.max(maxValue, maxPath);
38         return total;
39     }
40 }
  
```

Submission Result: **Accepted** ⓘ

[More Details](#) >

Share your acceptance!

