






// <https://leetcode.com/problems/binary-tree-maximum-path-sum/>

 Explore **Problems** Mock **Contest** Discuss  Store

Description

 **Solution**

 Discuss (999+)

 Submissions

124. Binary Tree Maximum Path Sum

Hard 4284 319 Add to List Share

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
```

Accepted 407,452 Submissions 1,176,550

Seen this question in a real interview before?

Companies  

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

```
1  /**
2   * Definition for a binary tree node.
3   * public class TreeNode {
4   *     public int val;
5   *     public TreeNode left;
6   *     public TreeNode right;
7   *     public TreeNode(int val=0, TreeNode left=null, TreeNode right=null) {
8   *         this.val = val;
9   *         this.left = left;
10 *         this.right = right;
11 *     }
12 * }
13 */
14 public class Solution {
15     public int MaxPathSum(TreeNode root) {
16         int max=0;
17         MaxPathSum(root,ref max);
18         return max;
19     }
20     public int MaxPathSum(TreeNode root, ref int maxSum)
21     {
22         if(root==null) return 0;
23
24         int leftTreeSum=Int32.MinValue,rtTreeSum=Int32.MinValue;
25         int leftSingleLongestSum = MaxPathSum(root.left,ref leftTreeSum);
26         int rtSingleLongestSum = MaxPathSum(root.right,ref rtTreeSum);
27
28         leftSingleLongestSum = leftSingleLongestSum>0 ? leftSingleLongestSum : 0;
29         rtSingleLongestSum = rtSingleLongestSum>0 ? rtSingleLongestSum : 0;
30
31         int rootPathSum = leftSingleLongestSum+rtSingleLongestSum+root.val;
32         maxSum = Math.Max(Math.Max(leftTreeSum,rtTreeSum),rootPathSum);
33         return Math.Max(leftSingleLongestSum,rtSingleLongestSum)+root.val;
34     }
35 }
```

Your previous code was restored from your local storage. [Reset to default](#)

```
public class Solution {  
  
    public int MaxPathSum(TreeNode root) {  
  
        int max=0;  
  
        MaxPathSum(root,ref max);  
  
        return max;  
    }  
  
    public int MaxPathSum(TreeNode root, ref int maxSum)  
    {  
  
        if(root==null) return 0;  
  
  
        int leftTreeSum=Int32.MinValue,rtTreeSum=Int32.MinValue;  
  
        int leftSingleLongestSum = MaxPathSum(root.left,ref leftTreeSum);  
  
        int rtSingleLongestSum = MaxPathSum(root.right,ref rtTreeSum);  
  

```

```
leftSingleLongestSum = leftSingleLongestSum>0 ? leftSingleLongestSum : 0;
rtSingleLongestSum = rtSingleLongestSum>0 ? rtSingleLongestSum : 0;

int rootPathSum = leftSingleLongestSum+rtSingleLongestSum+root.val;
maxSum = Math.Max(Math.Max(leftTreeSum,rtTreeSum),rootPathSum);
return Math.Max(leftSingleLongestSum,rtSingleLongestSum)+root.val;
}
}
```

Binary Tree Maximum Path Sum

Submission Detail

93 / 93 test cases passed.

Runtime: 108 ms

Memory Usage: 30.3 MB

Status: Accepted

Submitted: 14 minutes ago