Binary Tree Maximum Path Sum

Try to solve the Binary Tree Maximum Path Sum problem.

We'll cover the following

Statement

Given the root of a binary tree, return the maximum sum of any non-empty path.

A path in a binary tree is defined as follows:

- A sequence of nodes in which each pair of adjacent nodes must have an edge connecting them.
  - A node can only be included in a path once at most.
  - Including the root in the path is not compulsory.

You can calculate the path sum by adding up all node values in the path. To solve this problem, calculate the maximum path sum given the root of a binary tree so that there won't be any greater path than it in the tree.

Constraints:

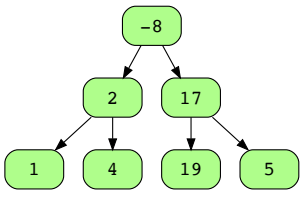
- $1 \leq$ Number of nodes in the tree $\leq 3 \times 10^4$.
- $-1000 \leq$ `Node.value` ≤ 1000

Examples

Sample example 1

Input: Level order

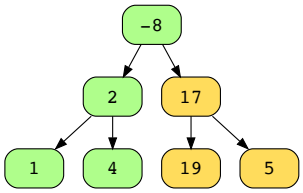
-8	2	17	1	4	19	5



Output

41

Maximum path sum is 19+17+5 = 41.



1 of 3

Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

Binary Tree Maximum Path Sum

1 What is the maximum path sum for the given binary tree?

level order = [-8, 2, 17, 1, 4, 24, 5]

A) 45

B) 46

C) 47

D) 48

Submit Answer

Question 1 of 3
0 attempted

Reset Quiz

Figure it out!

Here’s a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Drag and drop the cards to rearrange them in the correct sequence.

Initialize a maximum sum to negative infinity.

For a leaf node, determine its contribution equal to its value.

Otherwise, determine a node's contribution as its value plus the greater of the contributions of its left and right children.

Update maximum sum if the above is greater than previous maximum sum.

Reset | Show Solution | Submit

Try it yourself

Implement your solution in `main.java` in the following coding playground.

Java

usercode > main.java

```
// Definition of a binary tree node class
// class TreeNode<T> {
//     T data;
//     TreeNode<T> left;
//     TreeNode<T> right;
//
//     public TreeNode(T data) {
//         this.data = data;
//         this.left = null;
//         this.right = null;
//     }
// }

import java.util.*;
import ds_v1.BinaryTree.TreeNode;

public class Main{
  public static int maxPathSum(TreeNode<Integer> root) {

    // Replace this placeholder return statement with your code
    return -1;
  }
}
```

Powered by AI | Submit

Test Cases | Results

Case 1 | Case 2 | Case 3

Input #1

[-8,2,17,1,4,19,5]