Problem Challenge 2

We'll cover the following

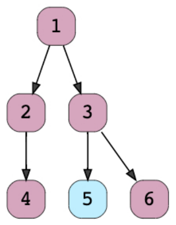
Path with Maximum Sum (hard)

Find the path with the maximum sum in a given binary tree. Write a function that returns the maximum sum.

A path can be defined as a **sequence of nodes between any two nodes** and doesn't necessarily pass through the root. The path must contain at least one node.

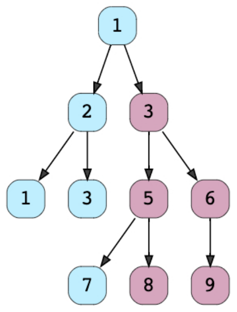
Example 1:

Output: 16
Explanation: The path with maximum sum is: [4, 2, 1, 3, 6]



Example 2:

Output: 31
Explanation: The path with maximum sum is: [8, 5, 3, 6, 9]



Try it yourself

Try solving this question here:

Java | Python3 | JS | C++

```
class TreeNode {
  int val;
  TreeNode left;
  TreeNode right;

  TreeNode(int x) {
    val = x;
  }
};

class MaximumPathSum {

  public static int findMaximumPathSum(TreeNode root) {
    // TODO: Write your code here
    return -1;
  }

  public static void main(String[] args) {
    TreeNode root = new TreeNode(1);
    root.left = new TreeNode(2);
    root.right = new TreeNode(3);
    System.out.println("Maximum Path Sum: " + MaximumPathSum.findMaximumPathSum(root));

    root.left.left = new TreeNode(1);
    root.left.right = new TreeNode(3);
    root.right.left = new TreeNode(5);
    root.right.right = new TreeNode(6);
    root.right.left.left = new TreeNode(7);
```

Run Save Reset

Back
Solution Review: Problem Challenge 1

Next
Solution Review: Problem Challenge 2

Mark as Completed

Report an Issue Ask a Question