Binary Tree Maximum Path Sum

This is a classic maximum path sum problem that requires computing the highest sum of any path in a binary tree. The path can start and end at any node, and it does not have to pass through the root.

Key Idea:

For each node, two values are important:

1. Max path sum including this node as the highest point:
   node.val + max(0, left_gain) + max(0, right_gain)
   (We choose max(0, ...) because we can discard negative contributions).
2. Max single-path gain to propagate up:
   node.val + max(0, max(left_gain, right_gain))

Maintain a global variable to track the maximum path sum encountered during traversal.

Algorithm:

1. Perform DFS.
2. Compute left_gain and right_gain recursively.
3. Update global max with node.val + left_gain + right_gain.
4. Return node.val + max(left_gain, right_gain) to parent.

Complexity:

- Time: O(N) - each node visited once.
- Space: O(H) - recursion stack (H = tree height).