75 Days of Code Day 35   Problem1448. Count Good Nodes in Binary Tree

Type :BST / dfs

Given a binary tree root, a node X in the tree is named good if in the path from root to X there are no nodes with a value greater than X.

Return the number of good nodes in the binary tree.

Example 1:

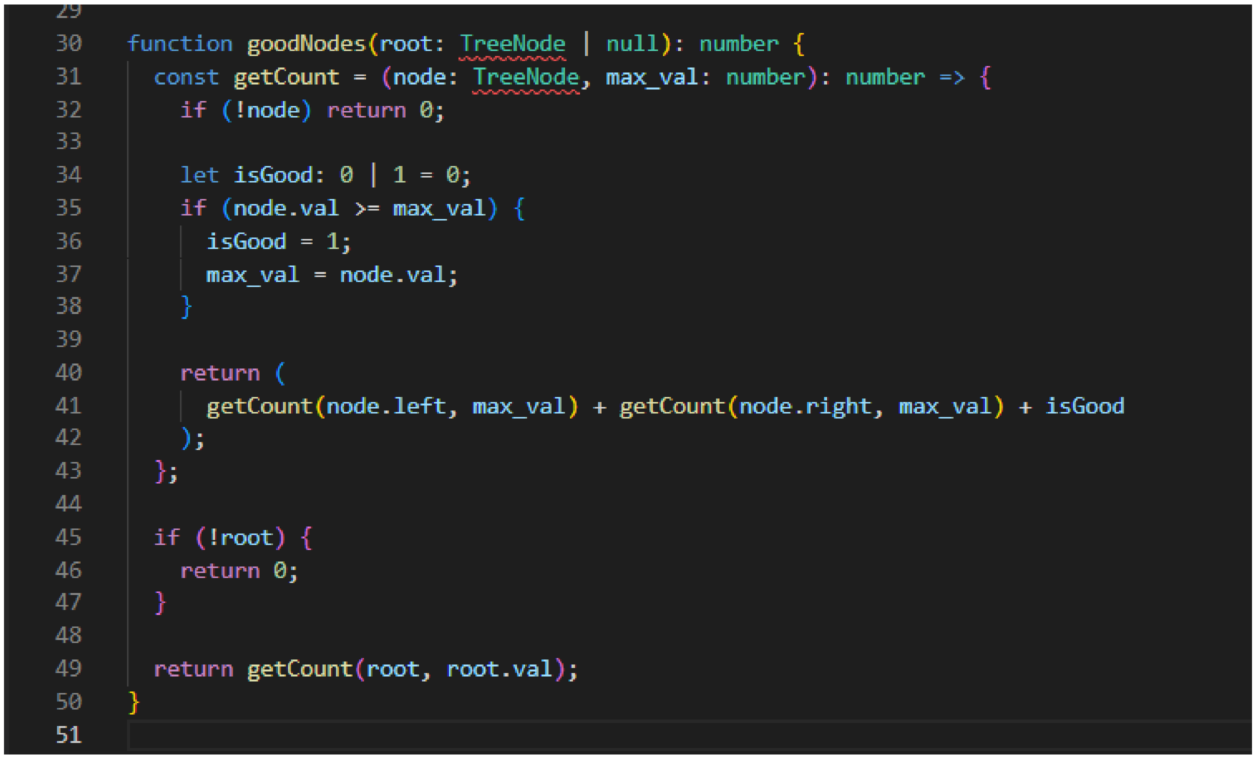
Input: root = [3,1,4,3,null,1,5]
Output: 4
Explanation: Nodes in blue are good.
Root Node (3) is always a good node.
Node 4 -> (3,4) is the maximum value in the path starting from the root.
Node 5 -> (3,4,5) is the maximum value in the path
Node 3 -> (3,1,3) is the maximum value in the path.
Example 2:

Input: root = [3,3,null,4,2]
Output: 3
Explanation: Node 2 -> (3, 3, 2) is not good, because "3" is higher than it.
Example 3:

Input: root = [1]
Output: 1
Explanation: Root is considered as good.

Solution using DFS

1. Traverse the list with recursion



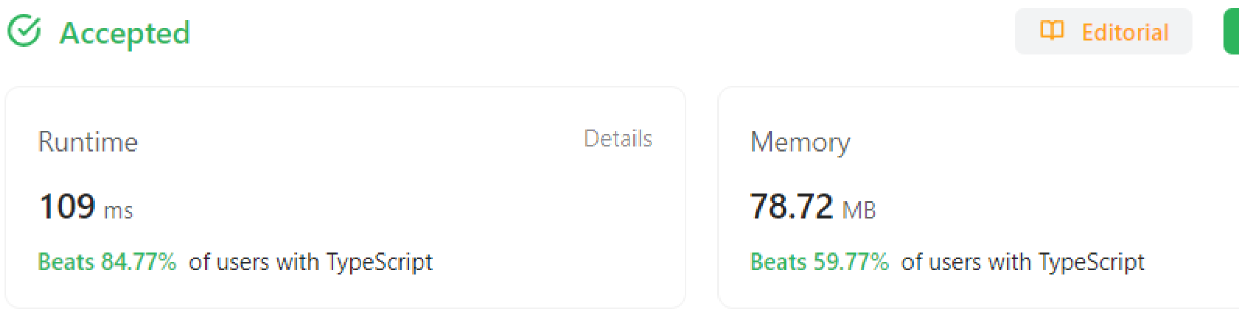
```typescript
function goodNodes(root: TreeNode | null): number {
  const getCount = (node: TreeNode, max_val: number): number => {
    if (!node) return 0;

    let isGood: 0 | 1 = 0;
    if (node.val >= max_val) {
      isGood = 1;
      max_val = node.val;
    }

    return (
      getCount(node.left, max_val) + getCount(node.right, max_val) + isGood
    );
  };

  if (!root) {
    return 0;
  }

  return getCount(root, root.val);
}
```



Accepted

Runtime

109 ms

Beats 84.77% of users with TypeScript

Memory

78.72 MB

Beats 59.77% of users with TypeScript

Shubham Agrahari