75 Days of Code Day 36 Problem 437. Path Sum III

Type : BST / dfs

Given the root of a binary tree and an integer targetSum, return the number of paths where the sum of the values along the path equals targetSum.

The path does not need to start or end at the root or a leaf, but it must go downwards (i.e., traveling only from parent nodes to child nodes).

Example 1:

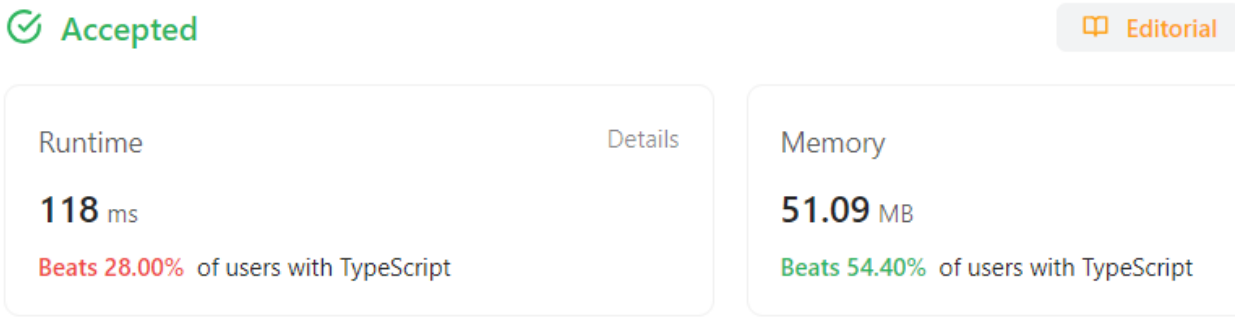
Input: root = [10,5,-3,3,2,null,11,3,-2,null,1], targetSum = 8
Output: 3
Explanation: The paths that sum to 8 are shown.
Example 2:

Input: root = [5,4,8,11,null,13,4,7,2,null,null,5,1], targetSum = 22
Output: 3

Solution using DFS

1. Traverse the list with recursion

```typescript
function pathSum(root: TreeNode | null, targetSum: number): number {
    if (!root) {
        return 0;
    }

    function dfs(node: TreeNode | null, currentSum: number): number {
        if (!node) {
            return 0;
        }

        currentSum += node.val;
        let count = 0;

        if (currentSum === targetSum) {
            count++;
        }

        count += dfs(node.left, currentSum);
        count += dfs(node.right, currentSum);

        return count;
    }

    return dfs(root, 0) + pathSum(root.left, targetSum) + pathSum(root.right, targetSum);
};
```



Shubham Agrahari