437. Path Sum III

Add to List ▾

Total Accepted: **15056**   Total Submissions: **38971**   Difficulty: **Easy**   Contributors: **Stomach_ache** (/stomach_ache/)

You are given a binary tree in which each node contains an integer value.
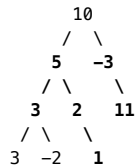
Find the number of paths that sum to a given value.

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

The tree has no more than 1,000 nodes and the values are in the range -1,000,000 to 1,000,000.

Example:

```
root = [10,5,-3,3,2,null,11,3,-2,null,1], sum = 8

      10
     /  \
    5   -3
   / \    \
  3   2   11
 / \   \
3  -2   1

Return 3. The paths that sum to 8 are:

1.  5 -> 3
2.  5 -> 2 -> 1
3.  -3 -> 11
```

Hide Tags | Tree (/tag/tree/)

Hide Similar Problems | (E) Path Sum (/problems/path-sum/) | (M) Path Sum II (/problems/path-sum-ii/)

Have you met this question in a real interview? Yes No

Discuss (https://discuss.leetcode.com/category/562) | Top Solutions | Pick One (/problems/random-one-question/)

C++ ▾   ⟳   </>

```cpp
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    int pathSum(TreeNode* root, int sum) {
        if(!root) return 0;
        return helper(root,0,sum)+pathSum(root->left,sum)+pathSum(root->right,sum);
    }
private:
    int helper(TreeNode* root, int sum, int target){
        if(!root) return 0;
        sum += root->val;
        return (sum==target) + helper(root->left,sum,target)+helper(root->right,sum,target);
    }
};
```

Custom Testcase ☐
Contribute Testcase ❓

Run Code   Submit Solution

Notes