Path Sum 3 (LeetCode 437)

*Description:*

You are given a binary tree in which each node contains an integer value.

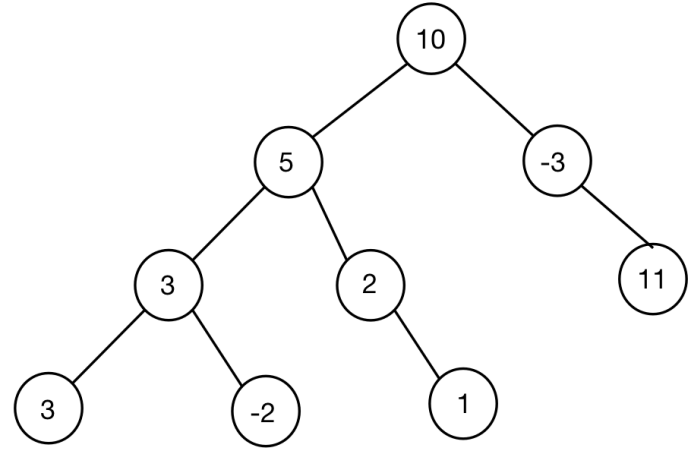
Find the number of paths that sum to a given value.

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:*

Given the below binary tree and sum = 8:



Return 3.

The paths that sum to 8 are:

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

*Code:*

*/\*\**

*\* 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:*

*void pathSum(TreeNode\* root, int sum, int & path) {*

*if (root == NULL)*

*return;*

*int sum\_left = sum - root->val;*

*if (sum\_left == 0) {*

*path ++;*

*}*

*pathSum(root->left, sum\_left, path);*

*pathSum(root->right, sum\_left, path);*

*}*

*void visitTreePath(TreeNode\* root, int sum, int &path) {*

*if(root == NULL)*

*return;*

*// Deal with the current node of Binary Tree.*

*pathSum(root, sum, path);*

*// Visit left child of the current node of Binary Tree.*

*visitTreePath(root->left, sum, path);*

*// Visit right child of the current node of Binary Tree.*

*visitTreePath(root->right, sum, path);*

*}*

*int pathSum(TreeNode\* root, int sum) {*

*int path = 0;*

*visitTreePath(root, sum, path);*

*return path;*

*}*

*};*