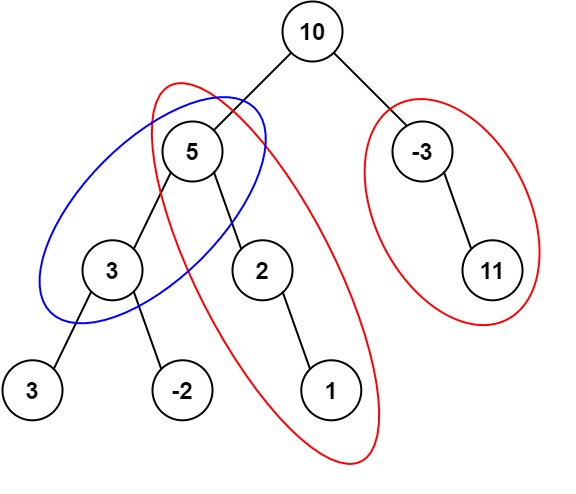
# 题目

给定一个二叉树的根节点root ，和一个整数targetSum，求该二叉树里节点值之和等于targetSum的路径的数目。

路径不需要从根节点开始，也不需要在叶子节点结束，但是路径方向必须是向下的（只能从父节点到子节点）。

示例 1：



输入：root = [10,5,-3,3,2,null,11,3,-2,null,1], targetSum = 8

输出：3

解释：和等于 8 的路径有 3 条，如图所示。

示例 2：

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

输出：3

提示:

二叉树的节点个数的范围是 [0,1000]

-109 <= Node.val <= 109

-1000 <= targetSum <= 1000

注意：本题与主站 437 题相同：

https://leetcode-cn.com/problems/path-sum-iii/

# 分析

/\*\*

 \* Definition for a binary tree node.

 \* struct TreeNode {

 \*     int val;

 \*     TreeNode \*left;

 \*     TreeNode \*right;

 \*     TreeNode() : val(0), left(nullptr), right(nullptr) {}

 \*     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

 \*     TreeNode(int x, TreeNode \*left, TreeNode \*right) : val(x), left(left), right(right) {}

 \* };

 \*/

class Solution {

private:

    vector<TreeNode\*> path;

    vector<TreeNode\*> result;

    int target;

public:

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

     //这个函数只找包含root的路径

        if(!root) return 0;

        int res = 0;

        if(root->val == sum) res++;

        res += pathSum\_nroot(root->left,sum - root->val); //自递归

        res += pathSum\_nroot(root->right,sum - root->val);

        return res;

    }

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

    int res = 0;

        if(!root) return res;

        if(root->val == sum) res++;

        //和root节点无关的

        res += pathSum(root->left,sum);

        res += pathSum(root->right,sum);

        //和root有关的，这里不能直接调用pathSum,否则递归以后造成“断开的路径”

        res += pathSum\_nroot(root->left,  sum - root->val);

        res += pathSum\_nroot(root->right, sum - root->val);

        return res;

    }

};