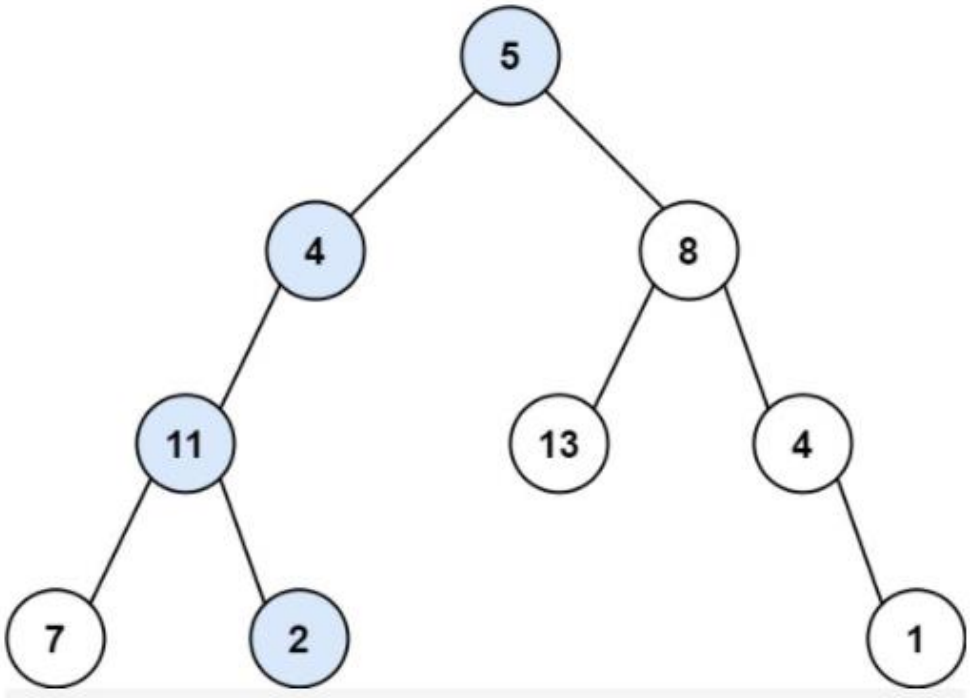
路径总和

给你二叉树的根节点 root 和一个表示目标和的整数 targetSum ，判断该树中是否存在 **根节点到叶子节点** 的路径，这条路径上所有节点值相加等于目标和 targetSum 。

叶子节点 是指没有子节点的节点。

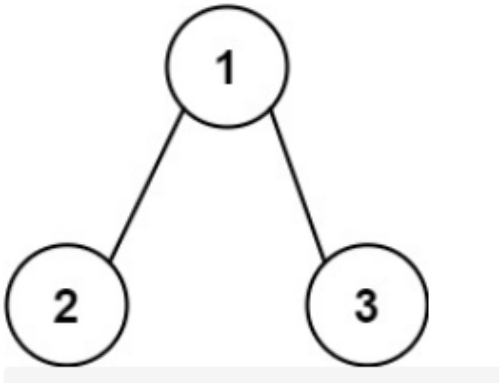
示例 1：



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

输出：true

示例 2：



输入：root = [1,2,3], targetSum = 5

输出：false

示例 3：

输入：root = [1,2], targetSum = 0

输出：false

```
/**
 * 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) {}
 * };

假定从根节点到当前节点的值之和为 val，我们可以将这个大问题转化为一个小问题：是否存在从当前节点的子节点到叶子的路径，满足其路径和为 sum - val。
 */
class Solution {
public:
    bool hasPathSum(TreeNode* root, int targetSum) {
        if(root==nullptr)
        {
            return false;
        }
        if(root->left==nullptr&&root->right==nullptr)
            return targetSum==root->val;
        return hasPathSum(root->left,targetSum-
root->val)||hasPathSum(root->right,targetSum-root->val);
    }
};
```