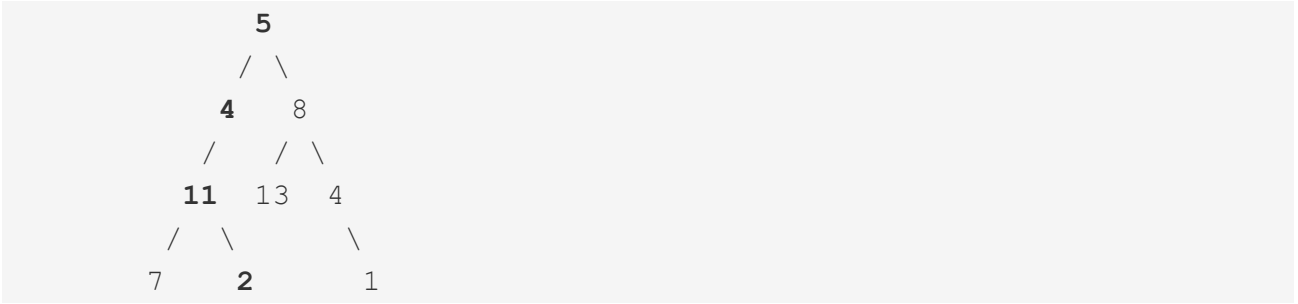
给定一个二叉树和一个目标和，判断该树中是否存在根节点到叶子节点的路径，这条路径上所有节点值相加等于目标和。

说明: 叶子节点是指没有子节点的节点。

示例:

给定如下二叉树，以及目标和 `sum = 22`，



返回 `true`, 因为存在目标和为 22 的根节点到叶子节点的路径 `5->4->11->2`。

思路，个人还是太不细心，是root到叶子节点间的距离，而个人算为了根节点。该题为深度遍历，遍历到符合sum的路径，返回，不符合则继续遍历。需注意，当根为NULL时，需手动剔除。

代码

```
/**
 * 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:
    bool dfs(TreeNode* root, int sum){
        if(root->left!=NULL){
            if(dfs(root->left,sum-root->left->val)){
                return true;
            }
        }
```

```
        if(root->right!=NULL){
            if(dfs(root->right,sum-root->right->val)){
                return true;
            }
        }
        if(root->left==NULL&&root->right==NULL&&sum==0){
            return true;
        }
        return false;
    }
    bool hasPathSum(TreeNode* root, int sum) {
        if(root==NULL){
            return false;
        }
        return dfs(root,sum-root->val);
    }
};
```