给定一个二叉搜索树和一个目标结果，如果 BST 中存在两个元素且它们的和等于给定的目标结果，则返回 true。

**案例 1:**

**输入：**

```
    5
   / \
  3   6
 / \   \
2   4   7

Target = 9
```

**输出：** `True`

**案例 2:**

**输入：**

```
    5
   / \
  3   6
 / \   \
2   4   7

Target = 28
```

**输出：** `False`

思路，用`map`把已经出现的变量标记起来，然后进行树的深度遍历，若是找到相等的值，就返回，没有相等的值，则继续遍历，遍历到最后还没有，则返回`false`

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(map<int,bool> &bj,int k,TreeNode* root){
        if(root==NULL){
            return false;
```

```
        }
        if(bj[k-root->val]){
            return true;
        }
        bj[root->val]=true;
        if(root->left!=NULL){
            if(dfs(bj,k,root->left)){
                return true;
            }
        }
        if(root->right!=NULL){
            if(dfs(bj,k,root->right)){
                return true;
            }
        }
        return false;
    }
    bool findTarget(TreeNode* root, int k) {
        map<int,bool> bj;
        return dfs(bj,k,root);
    }
};
```