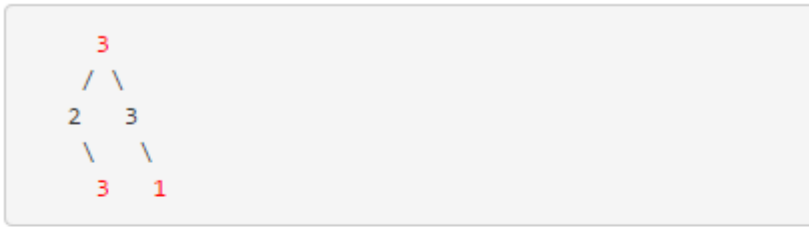
# 337. House Robber III

The thief has found himself a new place for his thievery again. There is only one entrance to this area, called the "root." Besides the root, each house has one and only one parent house. After a tour, the smart thief realized that "all houses in this place forms a binary tree". It will automatically contact the police if two directly-linked houses were broken into on the same night.

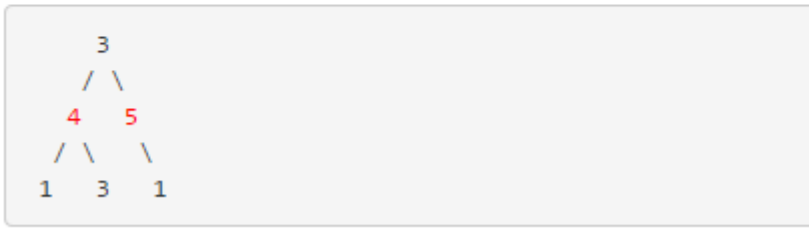
Determine the maximum amount of money the thief can rob tonight without alerting the police.

**Example 1:**




Maximum amount of money the thief can rob = 3 + 3 + 1 = **7**.

**Example 2:**




Maximum amount of money the thief can rob = 4 + 5 = **9**.

```
class Solution {
    unordered_map<TreeNode *, int> memorization;
public:
    int rob(TreeNode* root) {
        if (root == nullptr)
            return 0;
        if (memorization.count(root))
            return memorization[root];
        int l(0), r(0);
        if (root->left)
            l = rob(root->left->left) + rob(root->left->right);
        if (root->right)
            r = rob(root->right->left) + rob(root->right->right);
        return memorization[root] = max(root->val + l + r,
rob(root->left) + rob(root->right));
    }
};
```

```
// my java solution
// author : zzw
public class main_java {

     HashMap<TreeNode, Integer> memory;
     public int rob(TreeNode root) {
        if(root == null) return 0;
        if(memory.containsKey(root))
            return memory.get(root);
        int l=0; int r=0;
        if(root.left!=null)
        {
          l = rob(root.left.left)+rob(root.left.right);
        }
        if(root.right!=null)
        {
          r = rob(root.right.left)+rob(root.right.right);
        }
        int cost = Math.max(root.val+l+r,
rob(root.right)+rob(root.left));
        memory.put(root, cost);
        return cost;
    }

     public static void main(String[] args) {
          // TODO Auto-generated method stub

     }

}
```