Problem 968: Binary Tree Cameras

Problem Information

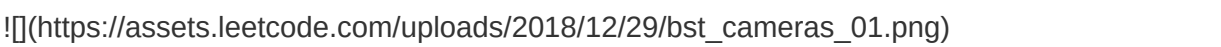
Difficulty: Hard
Acceptance Rate: 47.53%
Paid Only: No
Tags: Dynamic Programming, Tree, Depth-First Search, Binary Tree

Problem Description

You are given the `root` of a binary tree. We install cameras on the tree nodes where each camera at a node can monitor its parent, itself, and its immediate children.

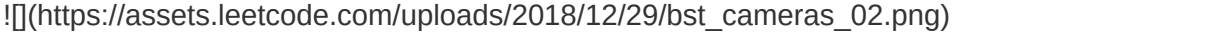
Return _the minimum number of cameras needed to monitor all nodes of the tree_.

Example 1:



Input: root = [0,0,null,0,0] **Output:** 1 **Explanation:** One camera is enough to monitor all nodes if placed as shown.

Example 2:



Input: root = [0,0,null,0,null,0,null,null,0] **Output:** 2 **Explanation:** At least two cameras are needed to monitor all nodes of the tree. The above image shows one of the valid configurations of camera placement.

Constraints:

* The number of nodes in the tree is in the range `[1, 1000]`. * `Node.val == 0`

Code Snippets

C++:

```
/**
* 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 {
public:
int minCameraCover(TreeNode* root) {

}
};
```

Java:

```
/**
* Definition for a binary tree node.
* public class TreeNode {
* int val;
* TreeNode left;
* TreeNode right;
* TreeNode() {}
* TreeNode(int val) { this.val = val; }
* TreeNode(int val, TreeNode left, TreeNode right) {
* this.val = val;
* this.left = left;
* this.right = right;
* }
* }
*/
class Solution {
public int minCameraCover(TreeNode root) {
```

```
}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def minCameraCover(self, root: Optional[TreeNode]) -> int:
```