2331. Evaluate Boolean Binary Tree

Solved

Easy | Topics | Companies | Hint

You are given the `root` of a **full binary tree** with the following properties:

- **Leaf nodes** have either the value `0` or `1`, where `0` represents `False` and `1` represents `True`.
- **Non-leaf nodes** have either the value `2` or `3`, where `2` represents the boolean `OR` and `3` represents the boolean `AND`.

The **evaluation** of a node is as follows:

- If the node is a leaf node, the evaluation is the **value** of the node, i.e. `True` or `False`.
- Otherwise, **evaluate** the node's two children and **apply** the boolean operation of its value with the children's evaluations.

Return *the boolean result of **evaluating** the* `root` *node.*

A **full binary tree** is a binary tree where each node has either `0` or `2` children.

A **leaf node** is a node that has zero children.

```
/**
* Definition for a binary tree node.
* public class TreeNode {
*     public var val: Int
*     public var left: TreeNode?
*     public var right: TreeNode?
*     public init() { self.val = 0; self.left = nil; self.right =
nil; }
*     public init(_ val: Int) { self.val = val; self.left = nil;
self.right = nil; }
*     public init(_ val: Int, _ left: TreeNode?, _ right:
TreeNode?) {
*         self.val = val
*         self.left = left
*         self.right = right
*     }
* }
*/
class Solution {
   func evaluateTree(_ root: TreeNode?) -> Bool {
       guard let root = root else {
           return false
```

```
        }

        if(root.val == 0) {
            return false
        } else if(root.val == 1) {
            return true
        } else if (root.val == 2) {
            return evaluateTree(root.left) ||
evaluateTree(root.right)
        } else if (root.val == 3) {
            return evaluateTree(root.left) &&
evaluateTree(root.right)
        } else {
            return false
        }
    }
}
```