623. Add One Row to Tree

Solved

Medium | Topics | Companies

Given the `root` of a binary tree and two integers `val` and `depth`, add a row of nodes with value `val` at the given depth `depth`.

Note that the `root` node is at depth `1`.

The adding rule is:

- Given the integer `depth`, for each not null tree node `cur` at the depth `depth - 1`, create two tree nodes with value `val` as `cur`'s left subtree root and right subtree root.
- `cur`'s original left subtree should be the left subtree of the new left subtree root.
- `cur`'s original right subtree should be the right subtree of the new right subtree root.
- If `depth == 1` that means there is no depth `depth - 1` at all, then create a tree node with value `val` as the new root of the whole original tree, and the original tree is the new root's left subtree.

Example 1:


```
/**
* 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 addOneRow(_ root: TreeNode?, _ val: Int, _ depth: Int) ->
TreeNode? {

        if (depth == 1){
           var newRoot = TreeNode(val)
           newRoot.left = root
           return newRoot
        }

       var tempDepth = 1
```

```
        var queue = [root]

        while(queue.isEmpty == false) {
            if(tempDepth == depth - 1) {
                for i in (0..<queue.count) {
                    var a = queue.removeFirst()!

                    // if(a.left != nil) {
                        var newRoot = TreeNode(val)
                        newRoot.left = a.left
                        a.left = newRoot
                    // }

                    // if(a.right != nil) {
                        var newRoot2 = TreeNode(val)
                        newRoot2.right = a.right
                        a.right = newRoot2
                                }
            } else {

                for i in (0..<queue.count) {
                    var a = queue.removeFirst()!
                    if(a.left != nil) {
                        queue.append(a.left)
                    }

                    if(a.right != nil) {
                        queue.append(a.right)
                    }
                }

                tempDepth = tempDepth + 1

            }
        }

        return root
    }
}
```