1123. Lowest Common Ancestor of Deepest Leaves

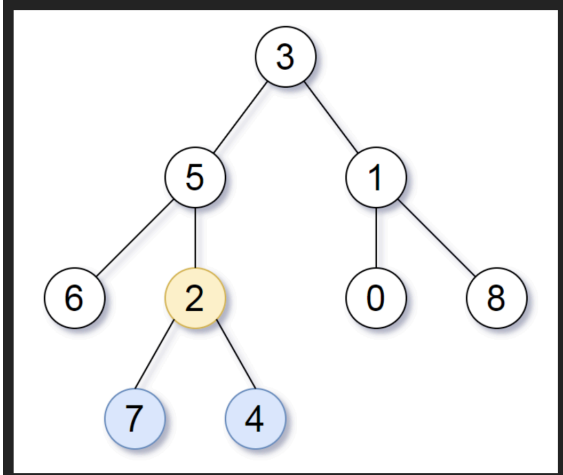
Medium · Topics · Companies · Hint

Given the `root` of a binary tree, return *the lowest common ancestor of its deepest leaves.*

Recall that:

- The node of a binary tree is a leaf if and only if it has no children
- The depth of the root of the tree is `0`. if the depth of a node is `d`, the depth of each of its children is `d + 1`.
- The lowest common ancestor of a set `S` of nodes, is the node `A` with the largest depth such that every node in `S` is in the subtree with root `A`.

Example 1:




```
Input: root = [3,5,1,6,2,0,8,null,null,7,4]
Output: [2,7,4]
Explanation: We return the node with value 2, colored in yellow in the diagram.
The nodes coloured in blue are the deepest leaf-nodes of the tree.
Note that nodes 6, 0, and 8 are also leaf nodes, but the depth of them is 2, but the depth of nodes 7 and 4 is 3.
```

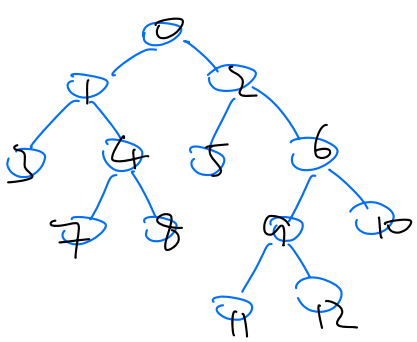
· DFS

Python3 · Auto

```python
# 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 lcaDeepestLeaves(self, root: Optional[TreeNode]) -> Optional[TreeNode]:
        def dfs(node):
            if not node:
                return (0,None)

            left_depth, left_LCA = dfs(node.left)
            right_depth, right_LCA = dfs(node.right)

            if left_depth > right_depth:
                return (left_depth+1, left_LCA)
            elif left_depth < right_depth:
                return (right_depth+1, right_LCA)
            else:
                return(left_depth+1, node)

        return dfs(root)[1]
```

- depth is used for tracking
- LCA is the goal, only used in the end.

Eg.



3.7.8.5.11.12.10 : (0, None)

9:(1, 9)

6:(2,