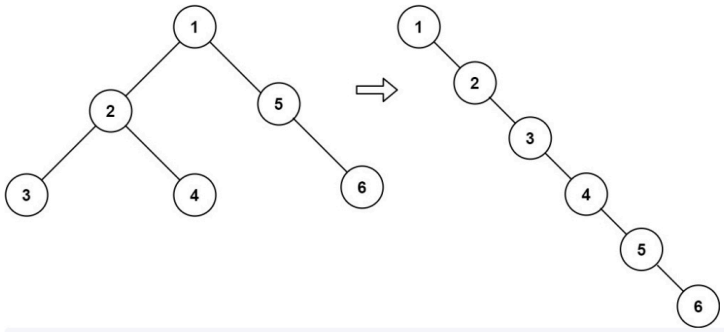
114. Flatten Binary Tree to Linked List

Medium 3886 391 Add to List Share

Given the `root` of a binary tree, flatten the tree into a "linked list":

- The "linked list" should use the same `TreeNode` class where the `right` child pointer points to the next node in the list and the `left` child pointer is always `null`.
- The "linked list" should be in the same order as a **pre-order traversal** of the binary tree.

Example 1:




```
Input: root = [1,2,5,3,4,null,6]
Output: [1,null,2,null,3,null,4,null,5,null,6]
```

Example 2:

```
Input: root = []
Output: []
```

Example 3:

```
Input: root = [0]
Output: [0]
```

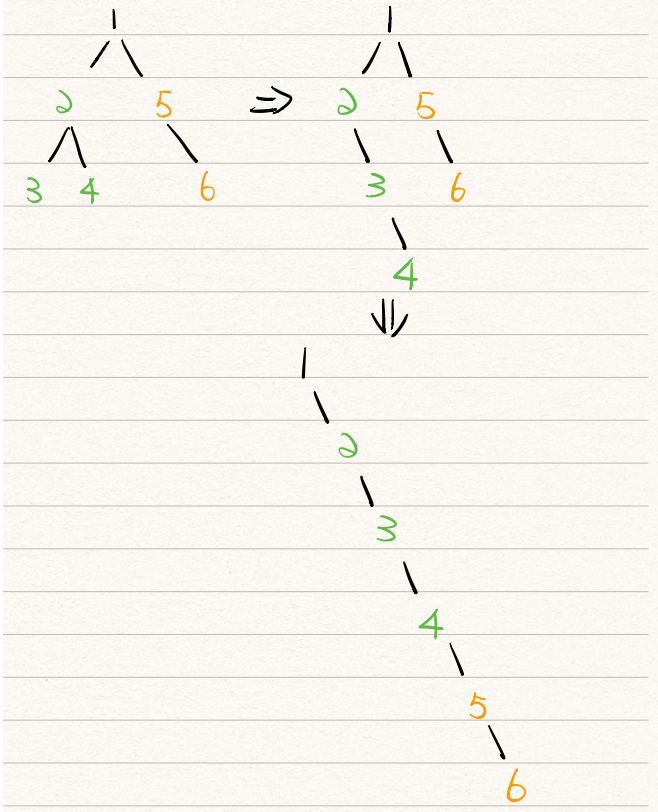
Constraints:

- The number of nodes in the tree is in the range `[0, 2000]`.
- `-100 <= Node.val <= 100`

Follow up: Can you flatten the tree in-place (with `O(1)` extra space)?

1. Flatten the left child tree and the right child tree.

2. Connect the right child tree blew the left child tree, and take the left child tree as the right child tree.



```cpp
/**
 * 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:
    void flatten(TreeNode* root) {
        if (root == NULL) {
            return;
        }

        // flatten the left tree and the right tree
        flatten(root->left);
        flatten(root->right);

        // connect the right tree below the left tree
        TreeNode *left = root->left;
        TreeNode *right = root->right;
        if (left != NULL) {
            TreeNode *p = left;
            while (p->right != NULL) {
                p = p->right;
            }
            p->right = right;
        } else {
            left = right;
        }

        // take the left tree as the right tree
        root->left = NULL;
        root->right = left;
    }
};
```