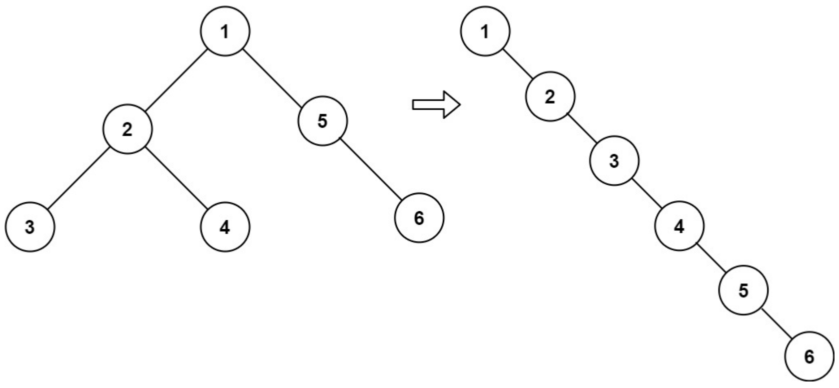
114. Flatten Binary Tree to Linked List

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

```
# Definition for a binary tree node.
# class TreeNode(object):
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution(object):
    head = None
    def flatten(self, root):
        def revPreorder(node):
            if node.right:
                revPreorder(node.right)
            if node.left:
                revPreorder(node.left)
            node.left, node.right, self.head = None,
self.head, node

        if root:
            revPreorder(root)
```