ZigZag Tree Traversal

Given a Binary Tree. Find the Zig-Zag Level Order Traversal of the Binary Tree.

Example 1:

Input: 3
   /
2   1
Output: 3 1 2

Example 2:

Input: 7
      /
     9   7
   / \ /
  8   8   6
 /
 10   9
Output: 7 7 9 8 8 6 9 10

```
from collections import deque
def zigZagTraversal(root):
    level = 0
    stack = deque()
    stack.appendleft(root)
    ans = []

    while True:
        if len(stack) == 0:
            break
        size = len(stack)
        while size > 0:
            if level % 2 == 0:
                temp = stack.popleft()
                ans.append(temp.data)
                if temp.left:
                    stack.append(temp.left)
                if temp.right:
                    stack.append(temp.right)
            else:
                temp = stack.pop()
                ans.append(temp.data)
                if temp.right:
                    stack.appendleft(temp.right)
                if temp.left:
                    stack.appendleft(temp.left)
            size = size-1
        level = level + 1
    return ans
```

Approach2 : Two Stacks.

```
def zigzagLevelOrder(self, root: TreeNode) -> List[List[int]]:
        if root is None:
            return
        currentLevel = []
        nextLevel = []
        currentLevel = [root]
        ans = []
        level = True

        while True:
            size = len(currentLevel)
            if size==0:
                break
            temp = []
            while size>0:
                node = currentLevel.pop()
                temp.append(node.val)
                if level:
                    if node.left:nextLevel.append(node.left)
                    if node.right:nextLevel.append(node.right)
                else:
                    if node.right:nextLevel.append(node.right)
                    if node.left:nextLevel.append(node.left)
                size = size-1
            if len(currentLevel)==0:
                level = False if level else True
                currentLevel = nextLevel
                nextLevel = []
                ans.append(temp)
        return ans
```