```java
/*Given a binary tree, return the zigzag level order traversal of its nodes' values. (ie, from left to right, then right to left for the next level and alternate between).

        For example:
        Given binary tree [3,9,20,null,null,15,7],
        3
        / \
        9  20
        /  \
        15   7
        return its zigzag level order traversal as:
        [
        [3],
        [20,9],
        [15,7]
        ]*/


import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import java.util.Queue;

class TreeNode {
    int val;
    TreeNode left;
    TreeNode right;

    TreeNode(int x) {
        val = x;
    }
}


class Solution {

    public List<List<Integer>> zigzagLevelOrder(TreeNode root) {

        List<List<Integer>> result = new ArrayList<>();

        if (root == null) {
            return result;
        }

        Queue<TreeNode> queue = new LinkedList<>();
        queue.offer(root);

        while (!queue.isEmpty()) {

            LinkedList<Integer> currentLayer = new LinkedList<>();
            int size = queue.size();

            if (result.size() % 2 == 0) {
                for (int i = 0; i < size; i++) {
                    TreeNode tempNode = queue.poll();
                    currentLayer.addLast(tempNode.val);
                    if (tempNode.left != null) {
                        queue.offer(tempNode.left);
                    }
                    if (tempNode.right != null) {
                        queue.offer(tempNode.right);
                    }
                }
            } else {
                for (int i = 0; i < size; i++) {
                    TreeNode tempNode = queue.poll();
                    currentLayer.addFirst(tempNode.val);
                    if (tempNode.left != null) {
                        queue.offer(tempNode.left);
                    }
                    if (tempNode.right != null) {
                        queue.offer(tempNode.right);
                    }
                }
            }

            result.add(currentLayer);
        }

        return result;
    }
}
```