```
/*
// Definition for a Node.
class Node {
    public int val;
    public Node left;
    public Node right;
    public Node next;

    public Node() {}

    public Node(int _val,Node _left,Node _right,Node _next) {
        val = _val;
        left = _left;
        right = _right;
        next = _next;
    }
};
*/
class Solution {
    public Node connect(Node root) {
        if (root == null)
            return root;
        Queue<Node> s = new LinkedList<>();
        s.offer(root);
        while (!s.isEmpty()) {
            int size = s.size();
            while (size != 0) {
                size--;
                Node n = s.poll();
                if (size != 0)
                    n.next = s.peek();
                if (n.left != null) {
                    s.offer(n.left);
                }
                if (n.right != null) {
                    s.offer(n.right);
                }
            }
        }
        return root;
    }
}
```

```
public void connect(TreeLinkNode root) {
        if (root == null) return;

        TreeLinkNode node = root.next;
        while (node != null) {
            if (node.left != null) {
                node = node.left;
                break;
            }
        if (node.right != null) {
            node = node.right;
            break;
        }
        node = node.next;
    }

    if (root.right != null) root.right.next = node;
    if (root.left != null) root.left.next = (root.right == null) ? node : root.right;

    connect(root.right);
    connect(root.left);
}
```