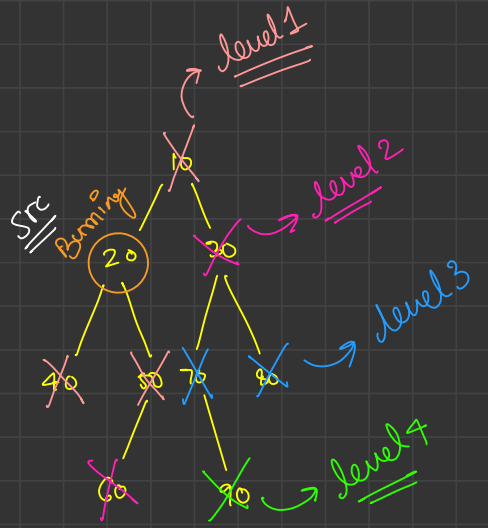
Burning Trees

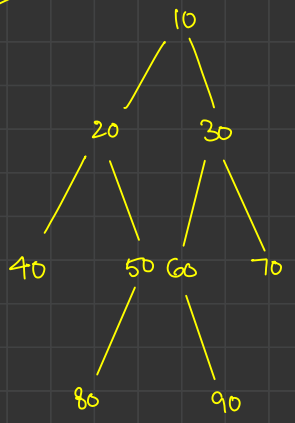
(BFS)

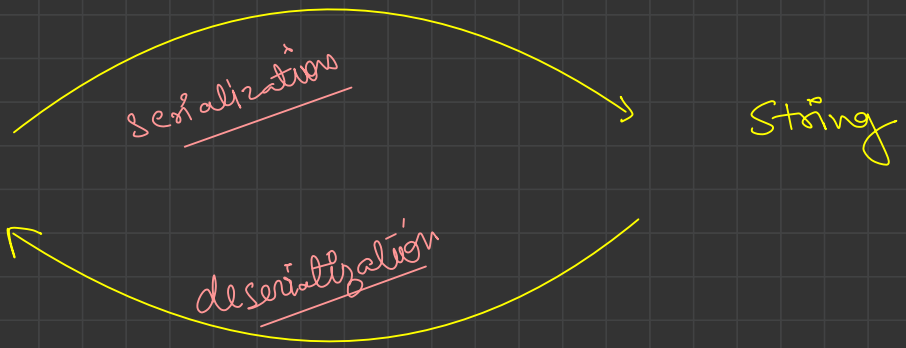


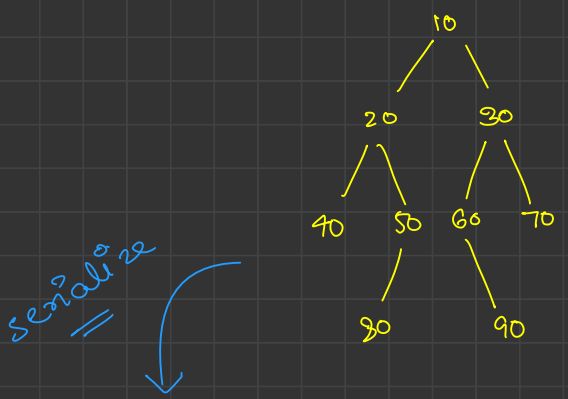
time = 4 units

Serialize and Deserialize a Binary tree

BT

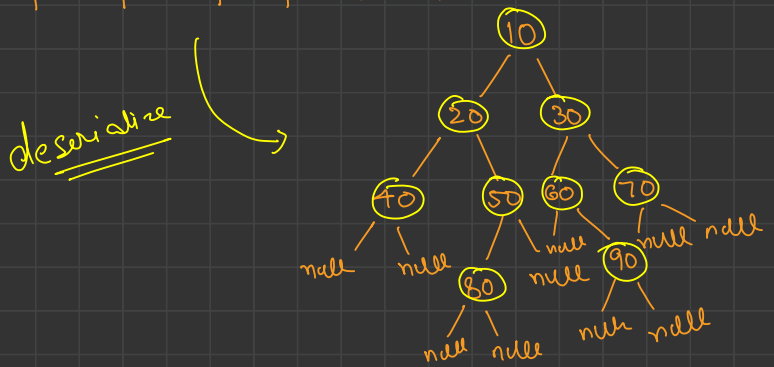






pre - order

"[10, 20, 40, null, null, 50, 80, null, null, null, 30, 60, null, 90, null, null, 70, null, null]

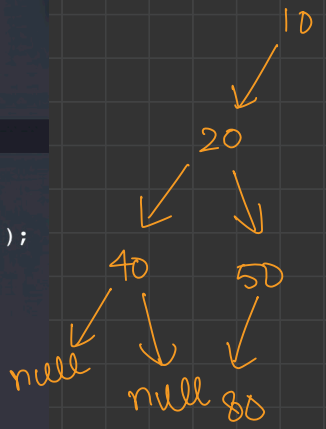


"[10, 20, 40, null, null, 50, 80, null, null, null, 30, 60, null, 90, null, null, 70, null, null]

```java
int si = 0;
static TreeNode buildTree(String[] pre) {
    if (si >=pre.length) {
        return null;
    }

    if (pre[si].equals("null") == true) {
        return null;
    }

    TreeNode root = new TreeNode(Integer.parseInt(pre[si]));
    si++;

    root.left = buildTree(pre);
    root.right = buildTree(pre);
}
```

si ++

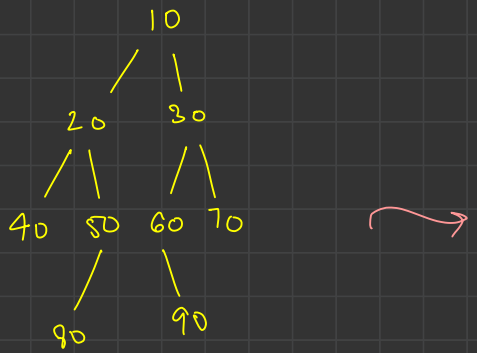
return root

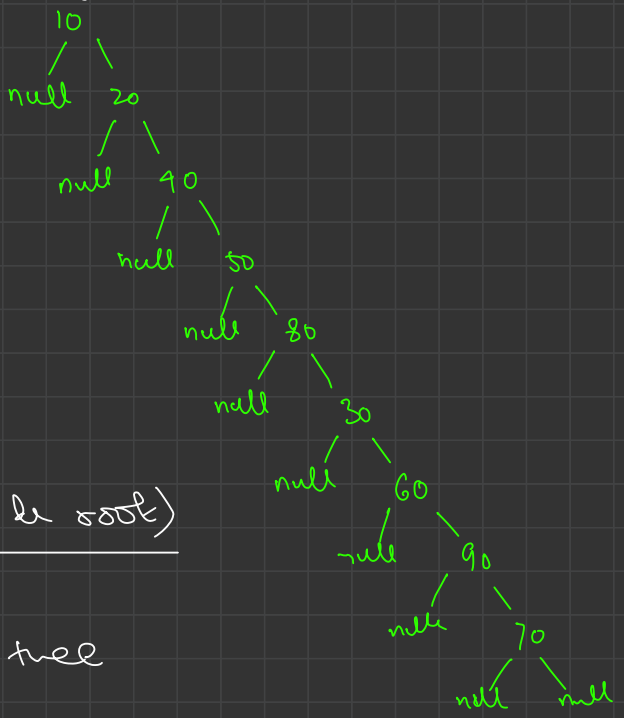


Flatten a Binary Tree {In-Place}

↓

Modify the i/p

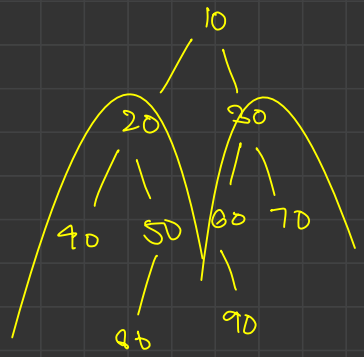




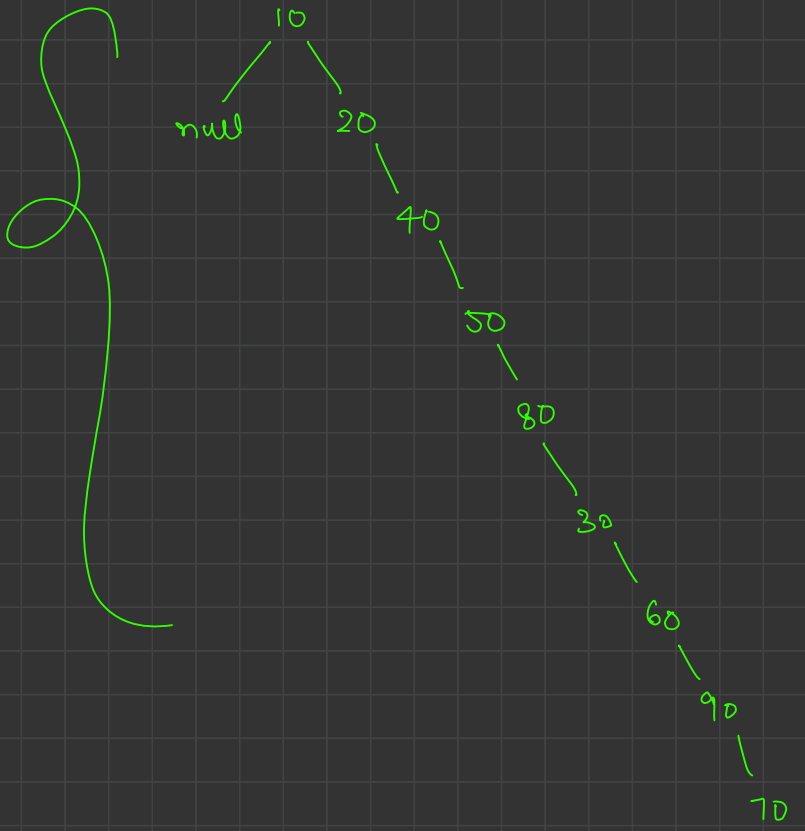
TreeNode flatten(TreeNode root)

↓

flattens a binary tree

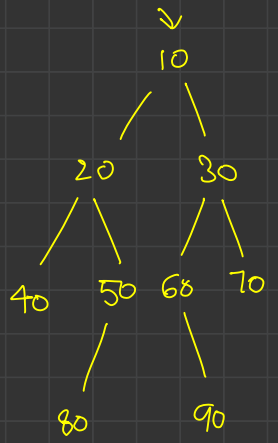



```java
public static void flatten(Node root){
    //Write code here
    if (root == null) {
        return;
    }

    flatten(root.left);

    flatten(root.right);

    Node temp =  root.right;
    root.right = root.left;
    root.left = null;

    Node curr = root;
    while (curr.right != null) {
        curr = curr.right;
    }
    curr.right = temp;
}
```



preorder traversal iterative

0 → print
1 → left
2 → right



Stack

10, 20, 40, 50, 80, 30, 60, 90, 70