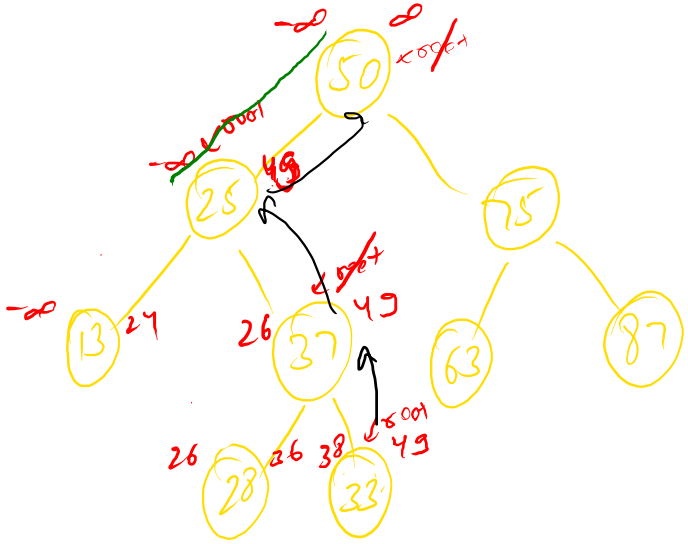
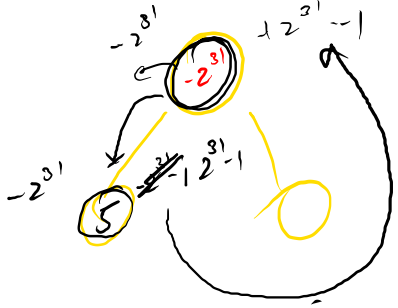
validate BST



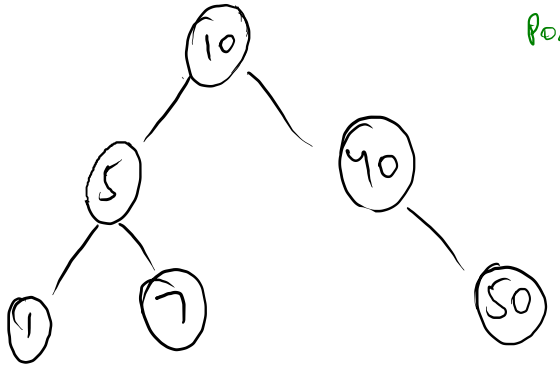


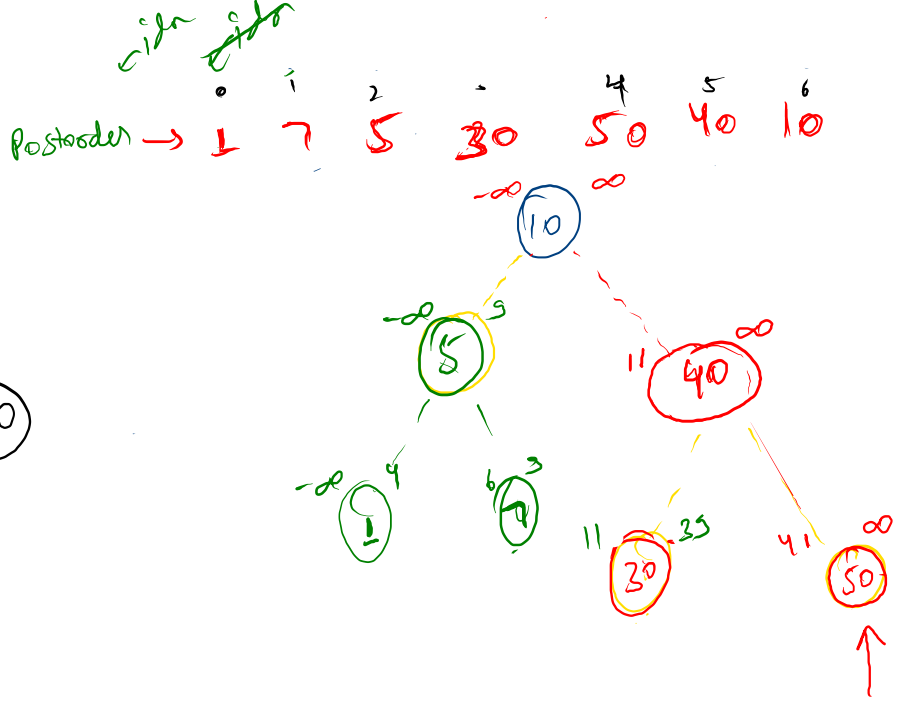
O(n)

-2^{31} : Integer.MIN_VALUE

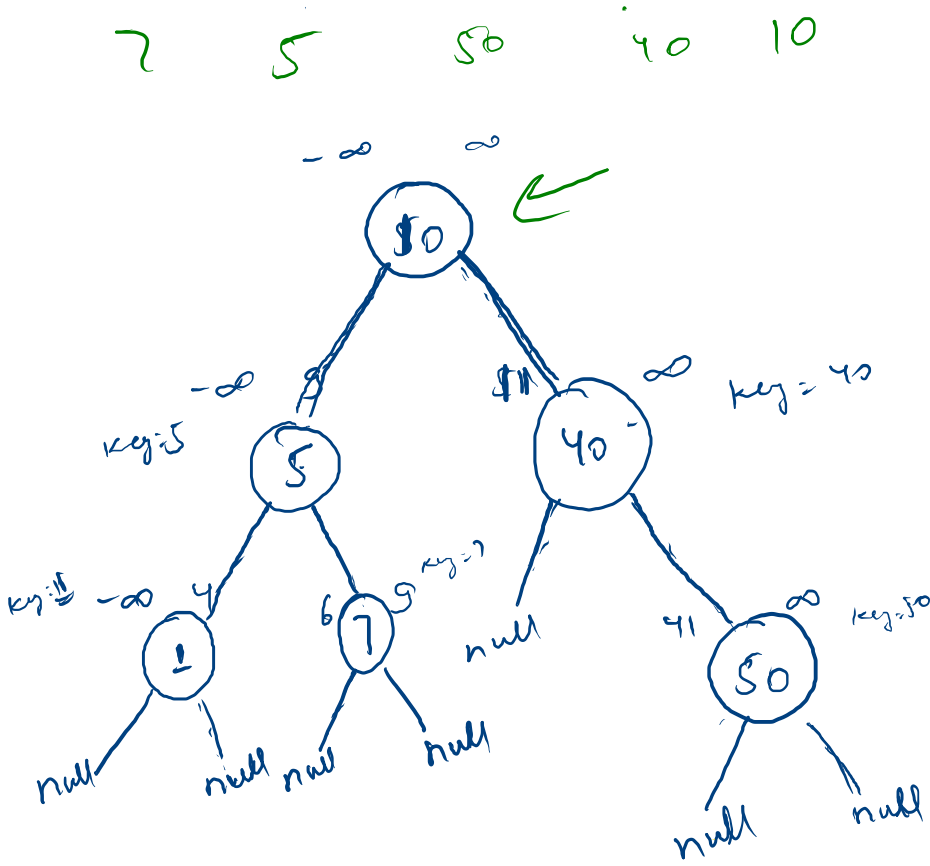
$2^{31}-1$: Integer.MAXVALUE

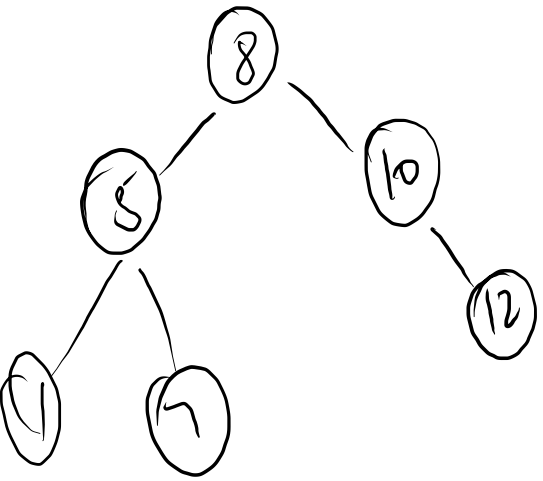
```
boolean traversal(TreeNode node, int min, int max)
{
    if (node == null) return true;
    if (node.val > max || node.val < min) return false;
    boolean temp1 = traversal(node.left, min, node.val-1);
    boolean temp2 = traversal(node.right, node.val+1, max);
    if (temp1 == false || temp2 == false) return false;
    return true;
}
```

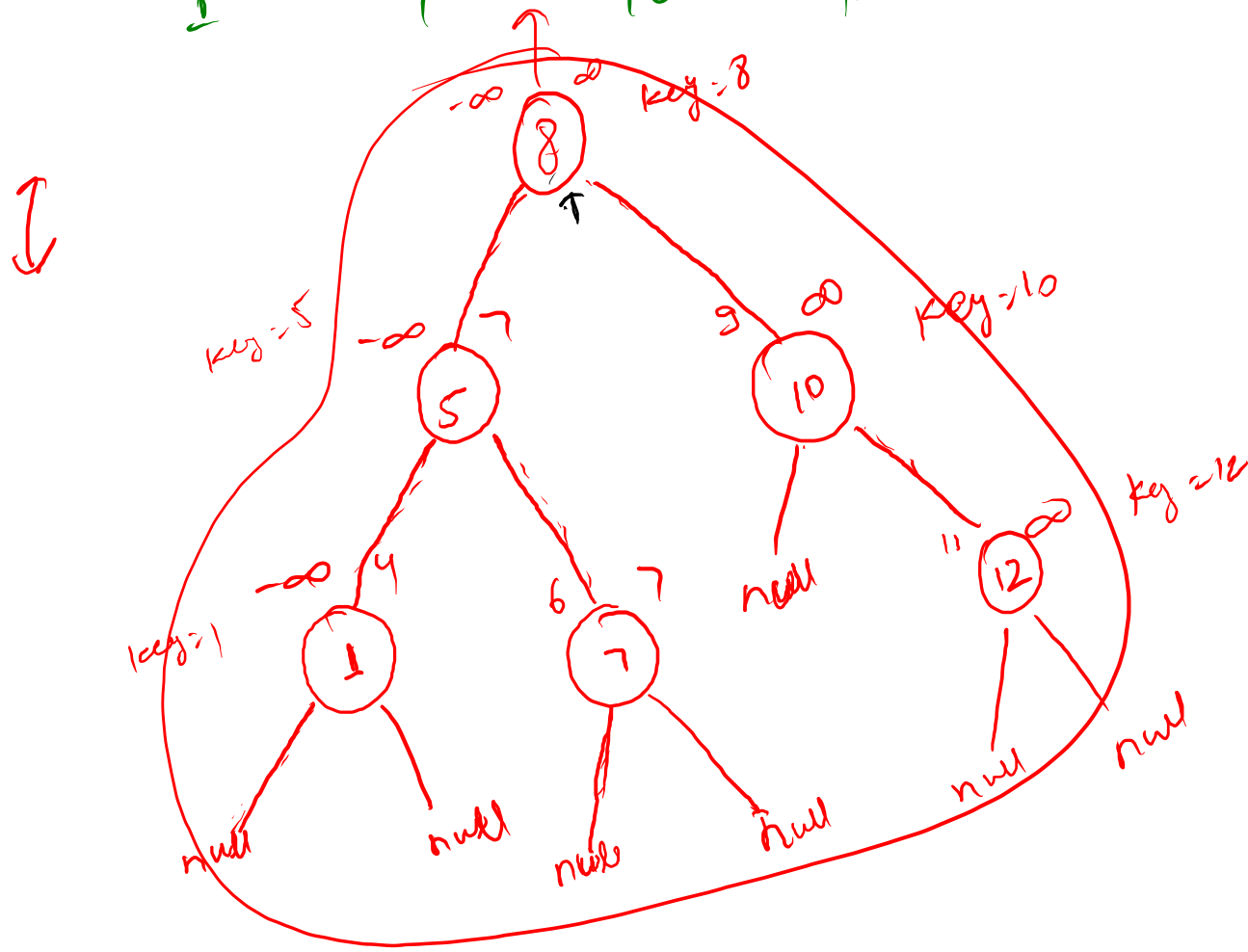

10
5
40
1
7
50

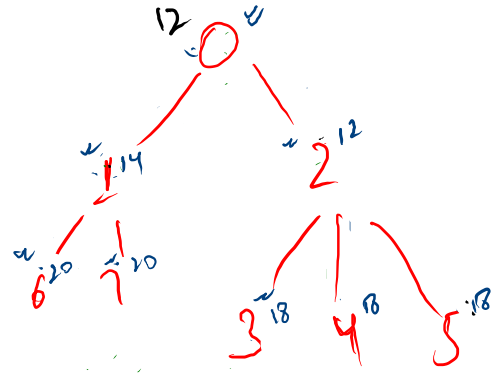

Postorder → 1 7 5 30 50 40 10
10
5
40
1
7
30
50

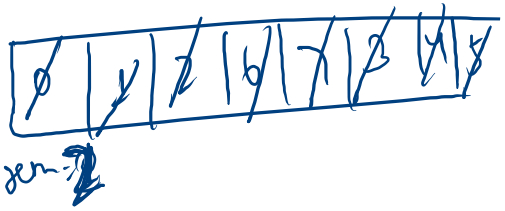
```java
public static Node constructTree(int post[], int n) {
    idx = post.length - 1;
    return construct(post, Integer.MIN_VALUE, Integer.MAX_VALUE);

}

static int idx;

public static Node construct(int[] post, int min, int max) {
    if (idx == -1) {
        return null;
    }

    if (post[idx] >= min && post[idx] <= max) {
        Node root = new Node(post[idx]);
        int key = post[idx];
        idx--;

        root.right = construct(post, key + 1, max);

        root.left = construct(post, min, key - 1);

        return root;
    }

    return null;

}
```



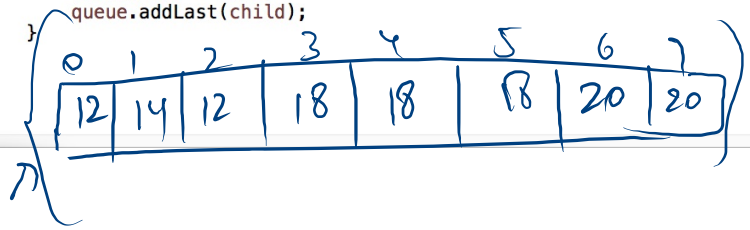

8
5
10
1
7
12

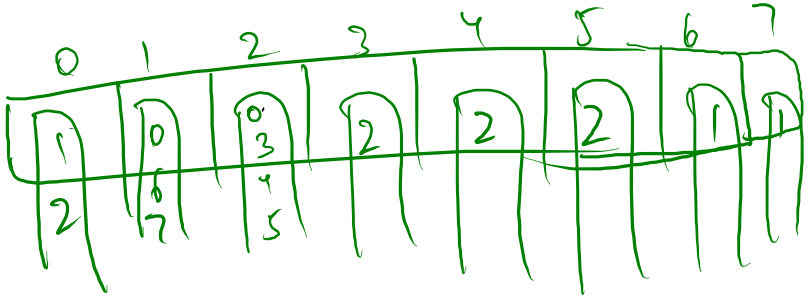

key = 8
key = 5
key = 10
key = 1
key = 12
8
5
10
1
7
12
null

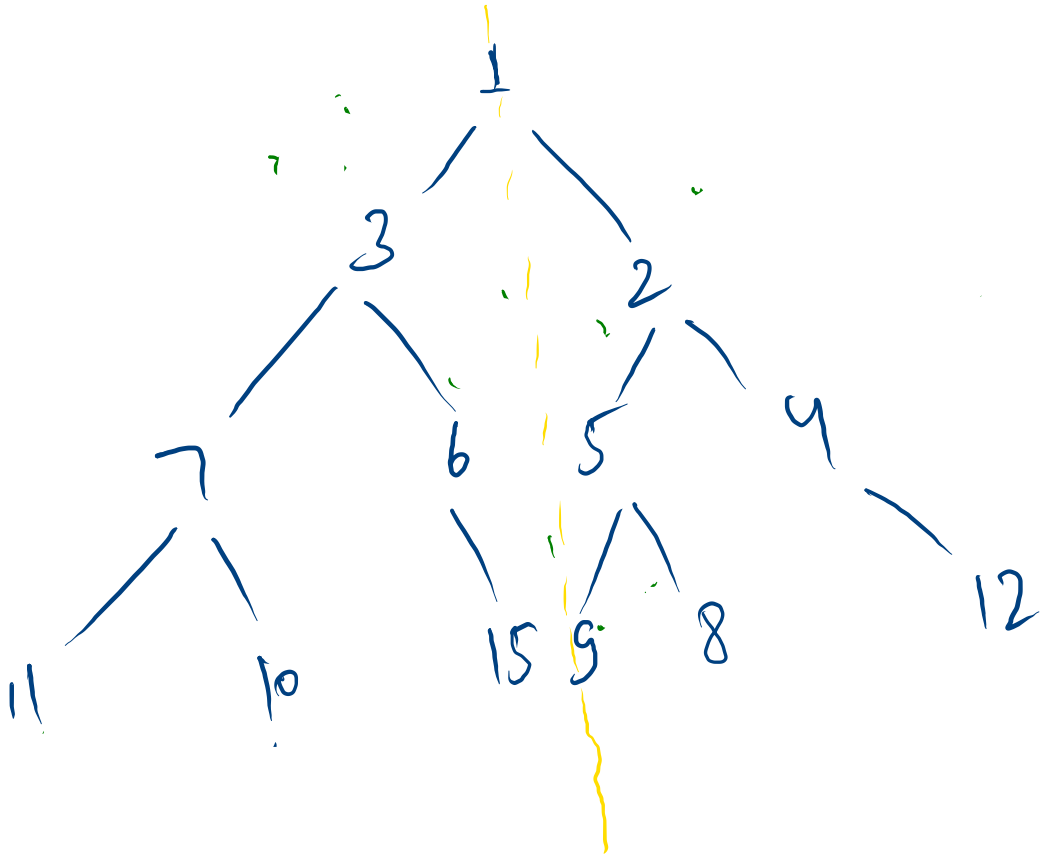




```
public void answercalc(int src, boolean[] vis, int N) {
    LinkedList<Integer> queue = new LinkedList<>();
    queue.addLast(src);
    vis[src] = true;
    while (queue.size() > 0) {
        int size = queue.size();
        while (size-- > 0) {
            int rem = queue.removeFirst();

            ArrayList<Integer> children = tree.get(rem);
            for (int child : children) {
                if (vis[child] == false) {
                    ans[child] = ans[rem] - count[child] + (N - count[child]);

                    vis[child] = true;
                    queue.addLast(child);
                }
            }
        }
    }
```



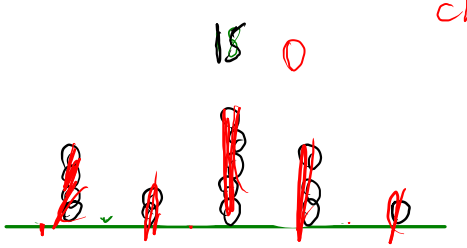




Ans: 0+1+6+28+132+30+135

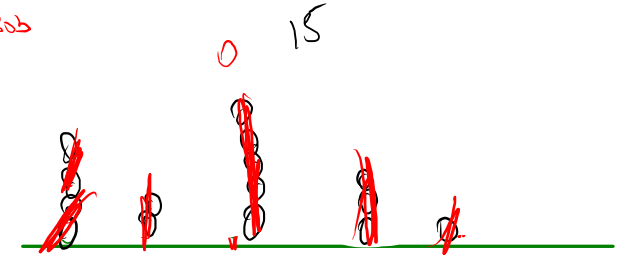
Buddy nim

Charlie



A

Charlie -> Alice -Bob



B

Alice

Bob

(i) $(\text{Count of coins})_A \; != \; (\text{Count of coins})_B$

↳ Alice

(ii) $(\text{Count of coins})_A \; == \; (\text{Count of coins})_B$

Structure diff

↳ Alice

Structure same

↳ Bob

→ 2 3 0 0 5

→ 5 3 2 0