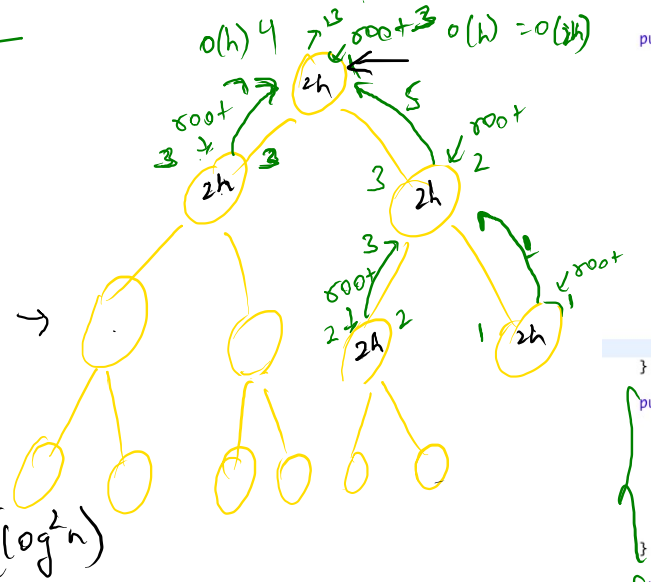
Count Complete tree nodes

① $h = \log n$

② 1 node → 2^h

$h + 2^h$ ≈ 2^h

$O(h^2) \Rightarrow O(\log^2 n)$

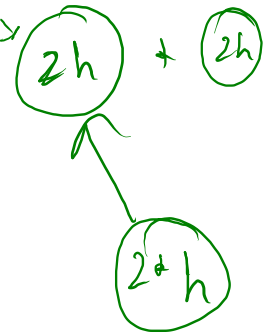


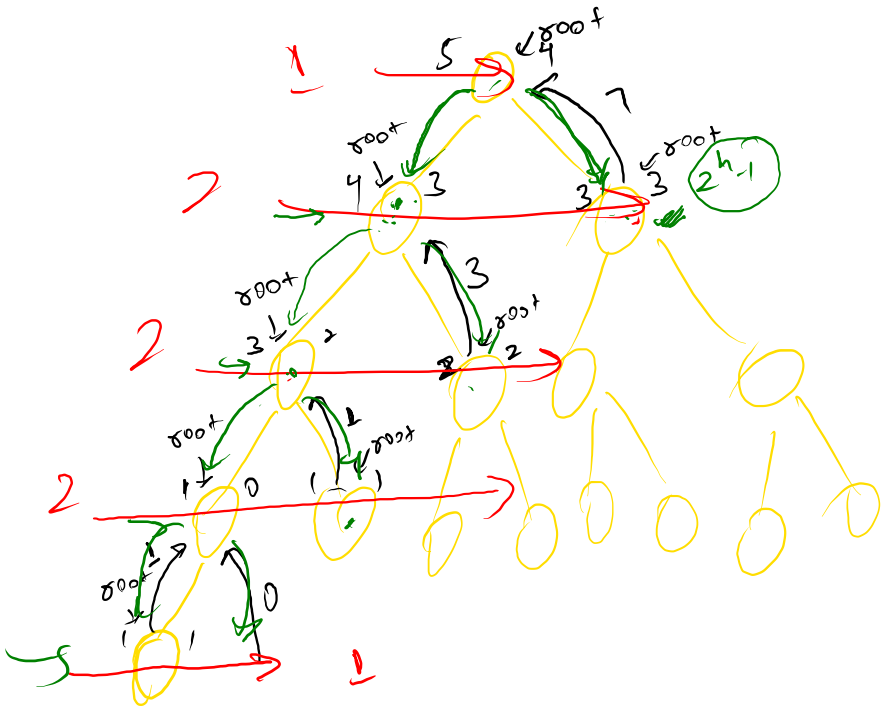

```java
public int countNodes(TreeNode root) {
    if(root == null){
        return 0;
    }

    int leftheight = getHeightleft(root);
    int rightheight = getHeightright(root);

    if(leftheight == rightheight ){
        return ((1<<leftheight)-1);
    }

    int left =  countNodes(root.left);
    int right = countNodes(root.right);

    return left+right+1;
}

public int getHeightleft(TreeNode root){
    int count =1;
    while(root.left != null){
        root = root.left;
        count++;
    }

    return count;
}

public int getHeightright(TreeNode root){
    int count =1;
    while(root.right != null){
        root = root.right;
        count++;
    }
```

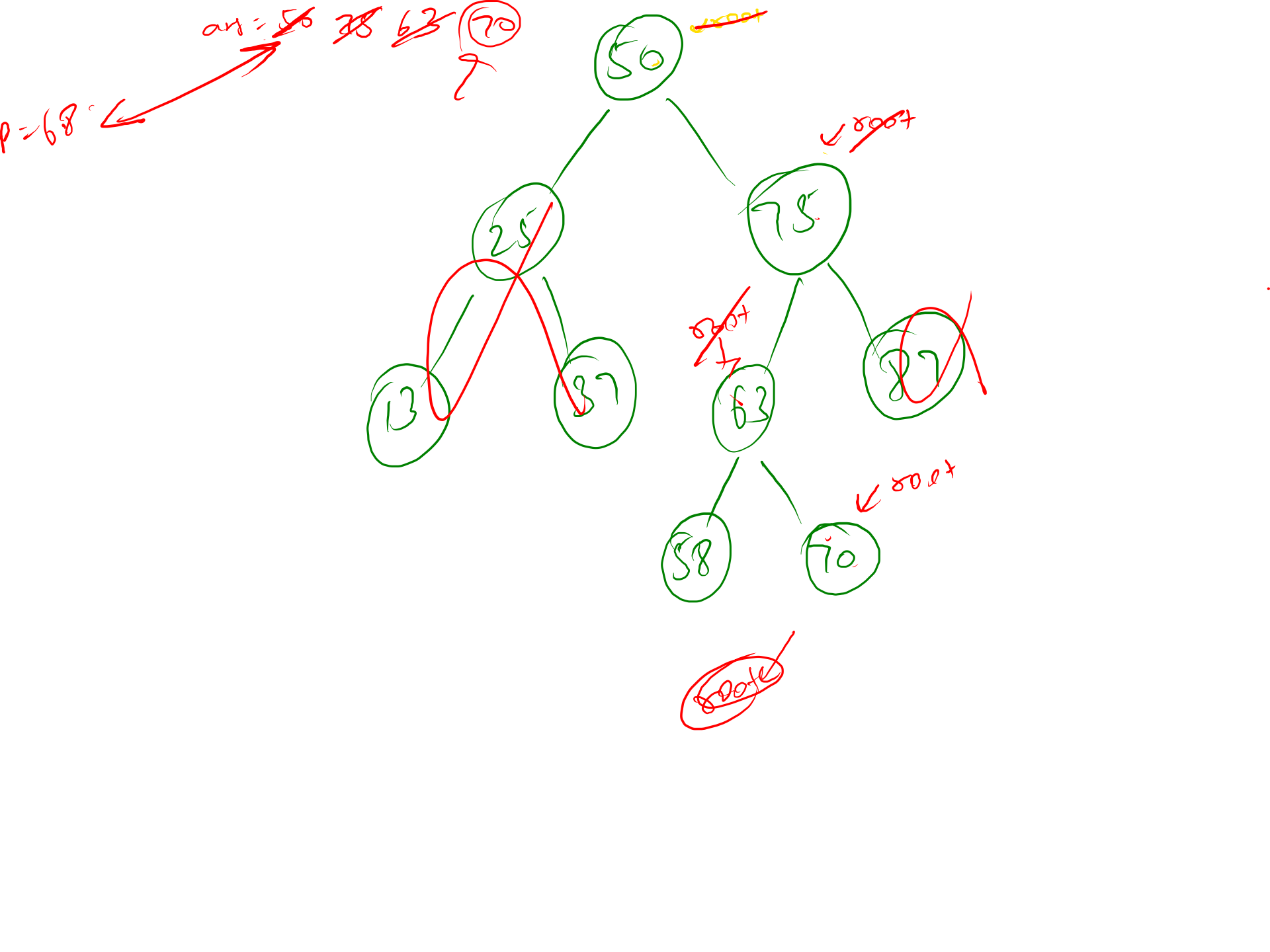
$\Rightarrow 2^h - 1$

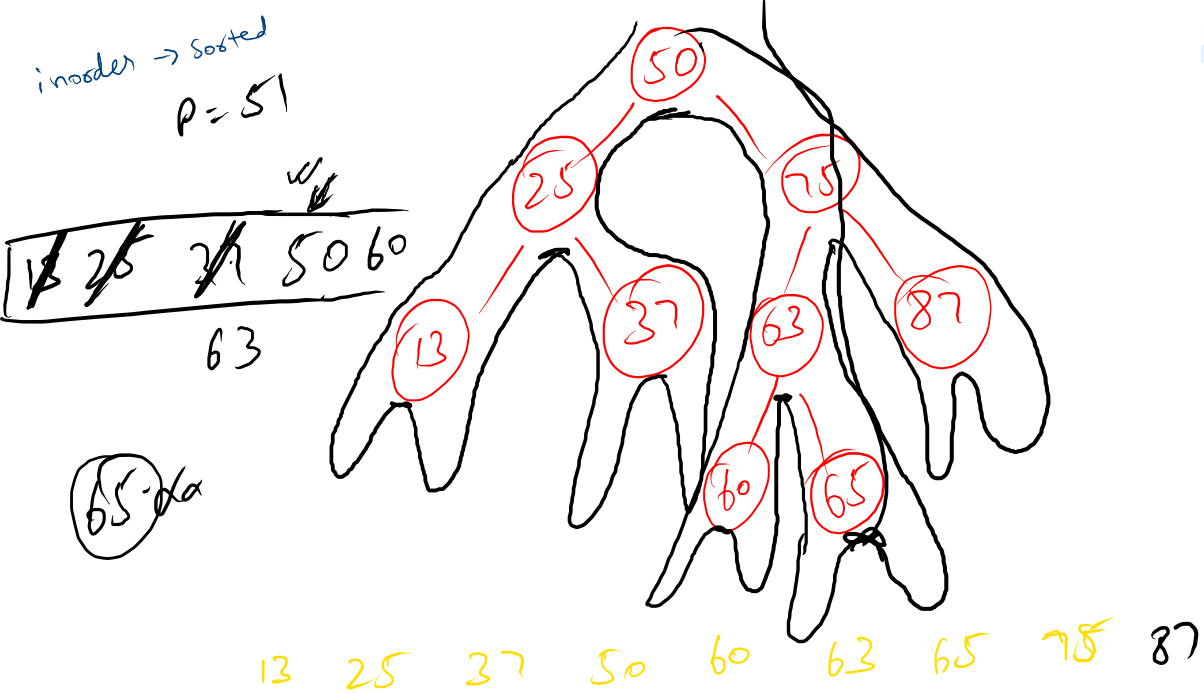
O(h)

O(h)



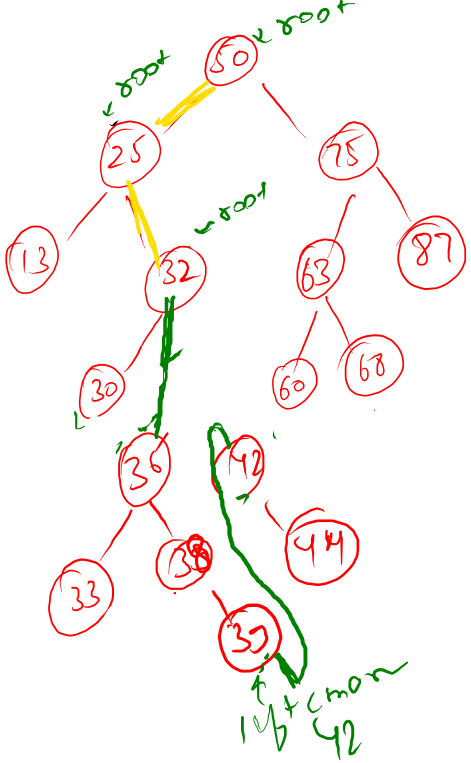



ans = 50 38 63 70
p = 68
50
25
75
13
37
63
87
58
70




```java
class Solution {
    public List<Integer> closestKValues(TreeNode root, double target, int k) {
        LinkedList<Integer> ans = new LinkedList<>();

        traversal(root, target,k,ans);

        return ans;
    }

    public void traversal(TreeNode root,double target,int k,LinkedList<Integer>ans){
        if(root == null){
            return;
        }

        traversal(root.left,target,k,ans);

        if(ans.size()< k){
            ans.addLast(root.val);
        }else{
            boolean [redacted];
            if(Math.abs(target - root.val) < Math.abs(target - ans.peekFirst())){
                [redacted];
                ans.removeFirst();
                ans.addLast(root.val);
            }

            if(flag == false){
                [redacted]
            }


        }

        traversal(root.right,target,k,ans);
    }
```

delete node in a BST

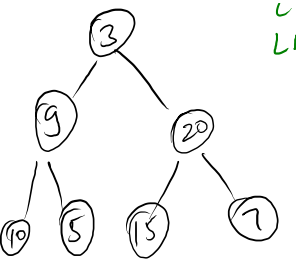


TreeNode traversal

```
if(root.val < key){
root.right = traversal(root.right);
}else if(root.val > key){
root.left = traversal(root.left);
}else{
root = deleteNode(root);
}
return root
```

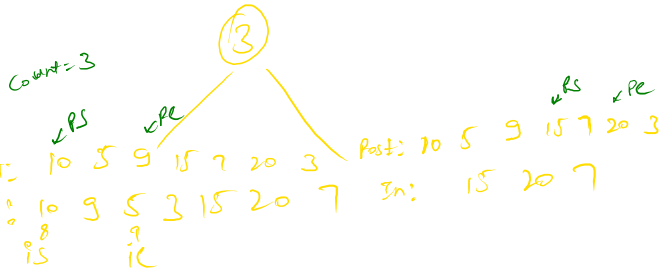
```
public TreeNode delete(TreeNode root){
    if(root.left == null && root.right == null){
        return null;
    }else if(root.left == null){
        return root.right;
    }else if(root.right == null){
        return root.left;
    }else{
        TreeNode rightc = root.right;
        TreeNode leftc = root.left;


        TreeNode leftcmax = leftc;
        while(leftcmax.right !=null ){
            leftcmax = leftcmax.right;
        }

        leftcmax.right = rightc;


        return leftc;

    }
}
```

PS + count - 1

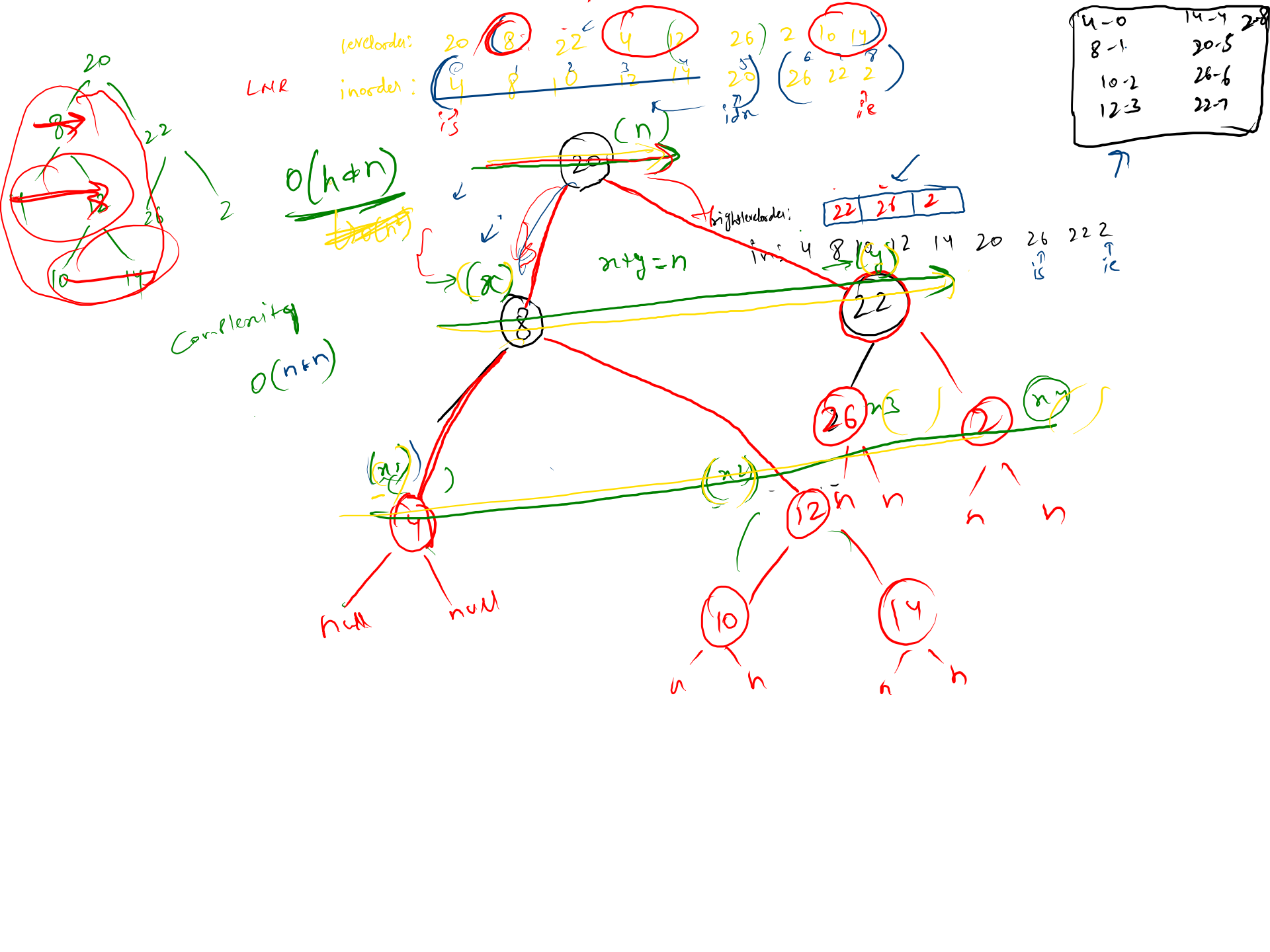


LRN Post: 10 5 9 15 7 20 3

LNR In: 10 9 5 3 15 20 7



Compl


levelorder:
inorder:
LNR
O(h+n)
Complexity
O(n+n)
x+y=n
rightlevelorder:
null
null

$a^b \% (10^9+7)$

$(a + a + ab + a + a) \% 10^9+7$

$(a \% 10^9+7) * (a \% 10^9+7) \% 10^9+7$

1 2 3 4 5 6 7 8 9 3 4 → int range → fast exp.

1 * 10 + 2 = 1234567

980 % 11 =

89
11 | 980
88
10
98
1 9

98 100 19

$a^{p-1} \% p = 1$ — p is a prime no.

$a^{k(p-1)} \% p = 1$

b >>>

fermat's little theorem

$a^b \% m \Rightarrow a^{b \% (m-1)} \% m$

$a \% b \Rightarrow a \% b^{m-2} \% m$

$a^b \% m$ $\quad b = (m-1)q1 + r1$

$a^b \% m = a^{q1(m-1)+r1} \% m$

$= (a^{q1(m-1)} * a^{r1}) \% m \Rightarrow (a^{q1(m-1)} \% m * a^{r1} \% m) \% m$

$a^b \% m = a^{r1} \% m \Rightarrow a^{b \% (m-1)}$

Nim game

42 49 13

101100 110001 001110

1 → 0

XOR = 0 → non-zero xor value

XOR = 0 ← min 1 move