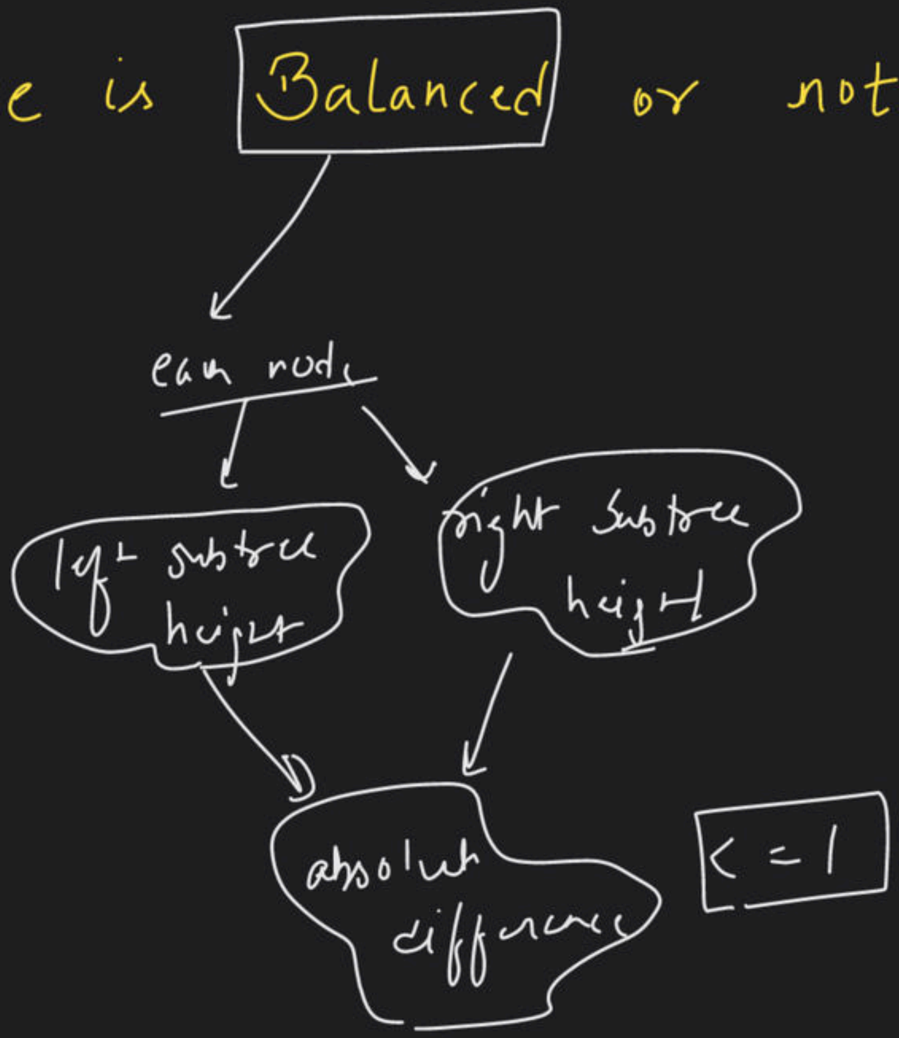
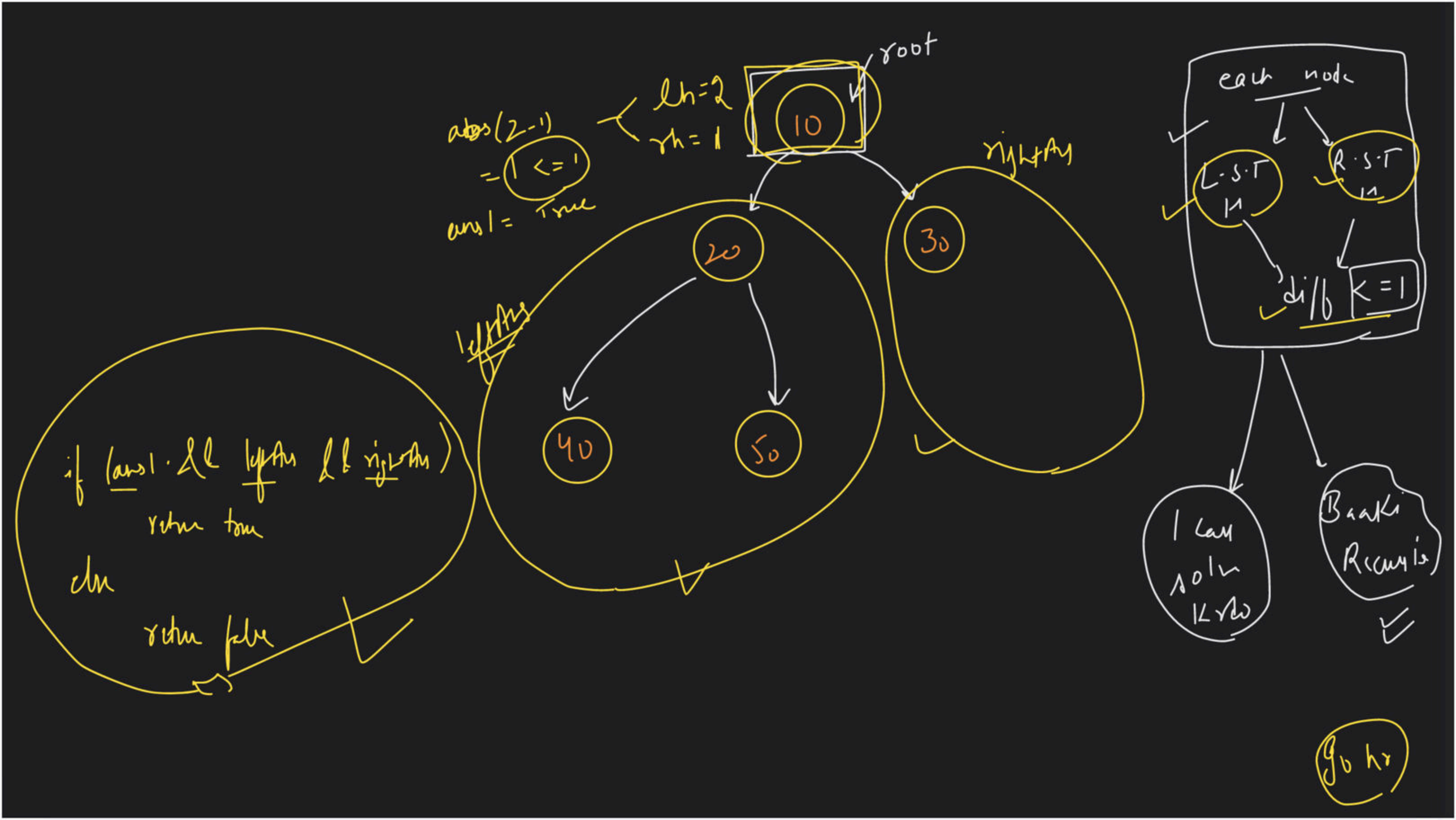
Trees Class - 2

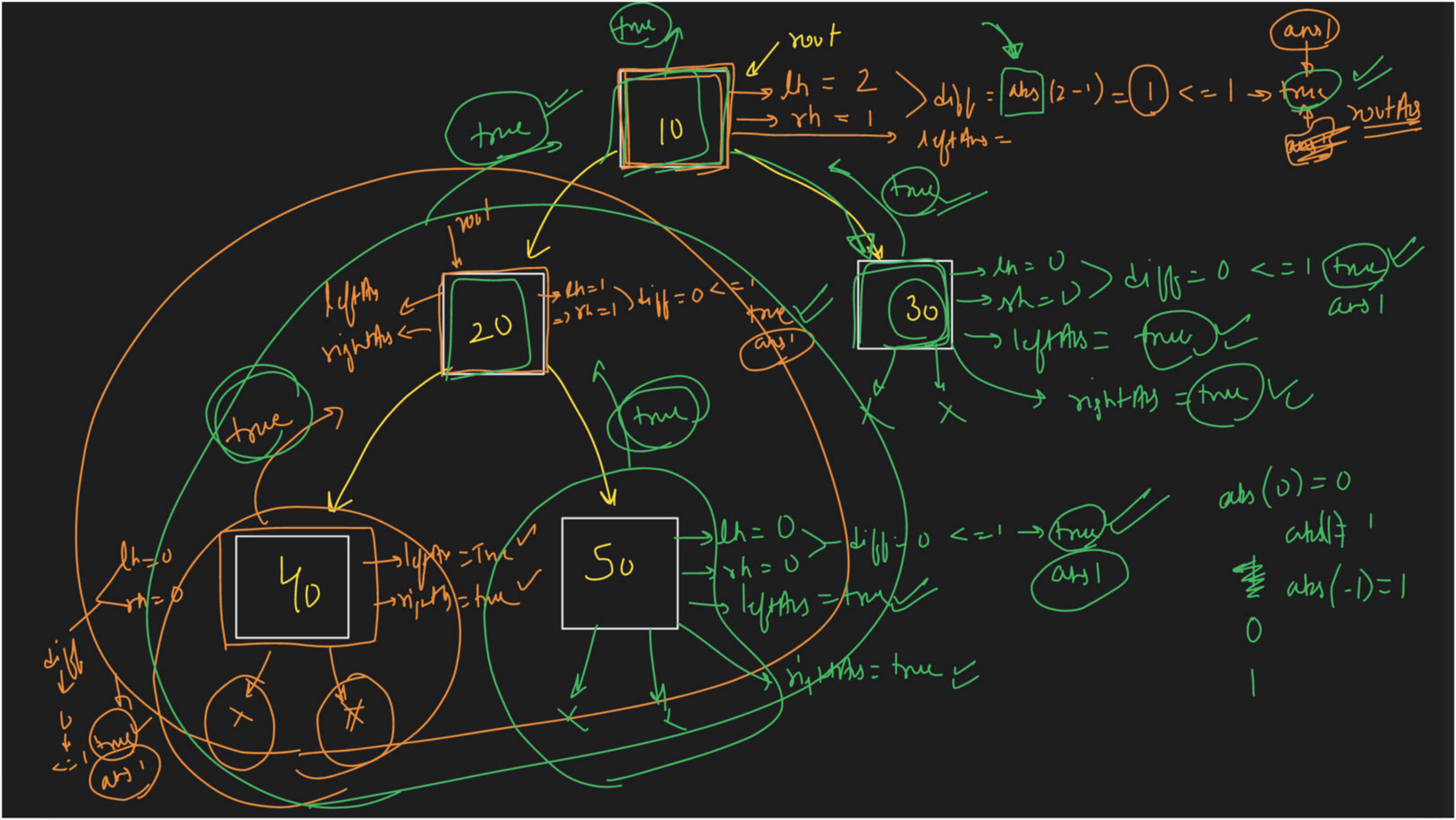
Special class

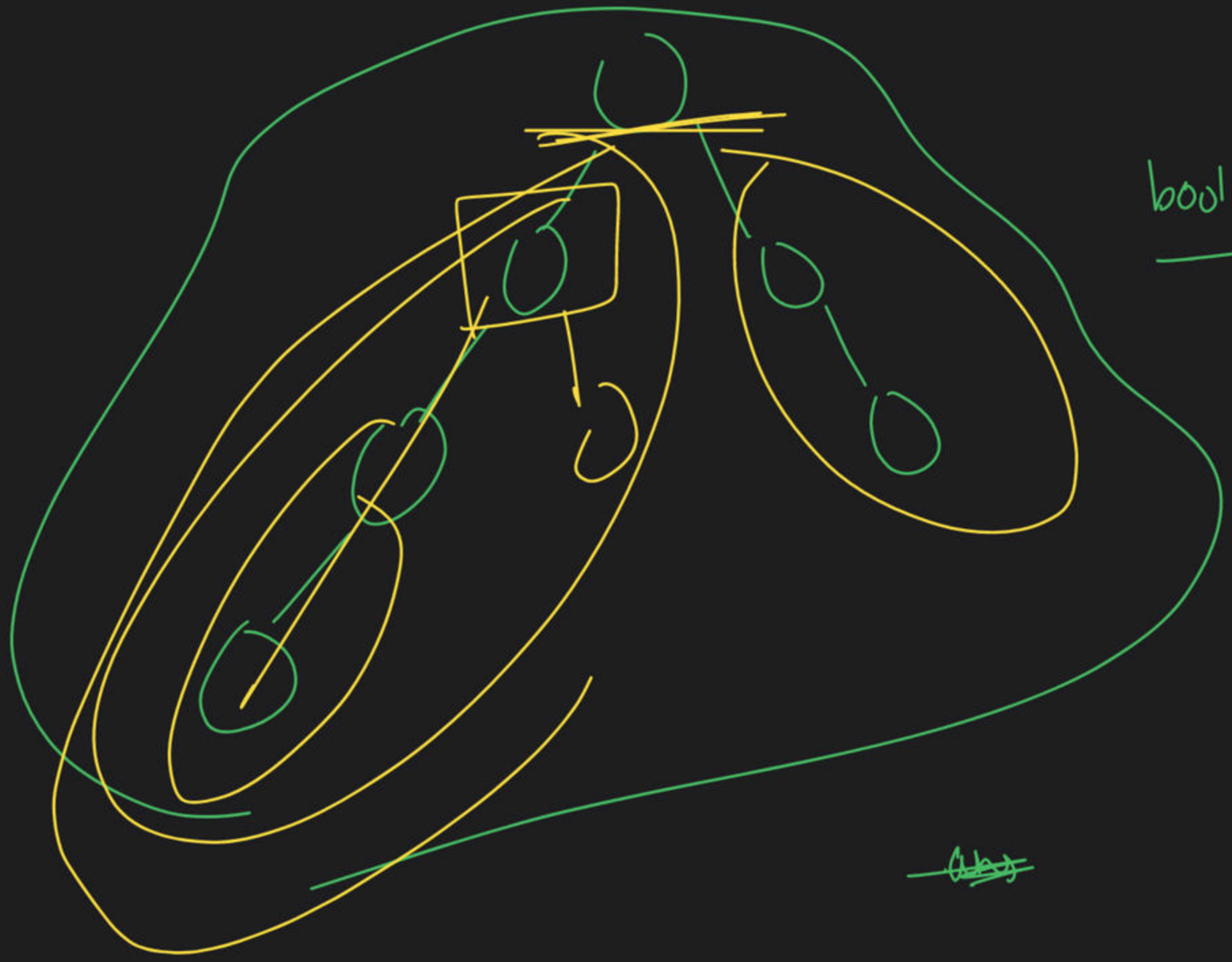
Love Babbar • May 7, 2023

→ Check if a Binary Tree is **Balanced** or not




root
10
lh=2
rh=1
abs(2-1)
= 1 <= 1
ans1 = True
20
30
40
50
leftAns
rightAns
if (ans1 && leftAns && rightAns)
return true
else
return false
each node
L.S.T
R.S.T
diff
K=1
I can solve it now
Baaki Recursion


true
root
10
lh = 2
rh = 1
diff = abs(2-1) = 1 <= 1 → true
ans1
rootAns
leftAns =
20
30
40
50
lh=1
rh=1
diff = 0 <= 1
lh = 0
rh = 0
leftAns = true
rightAns = true
X
abs(0) = 0
abs(-1) = 1
0
1



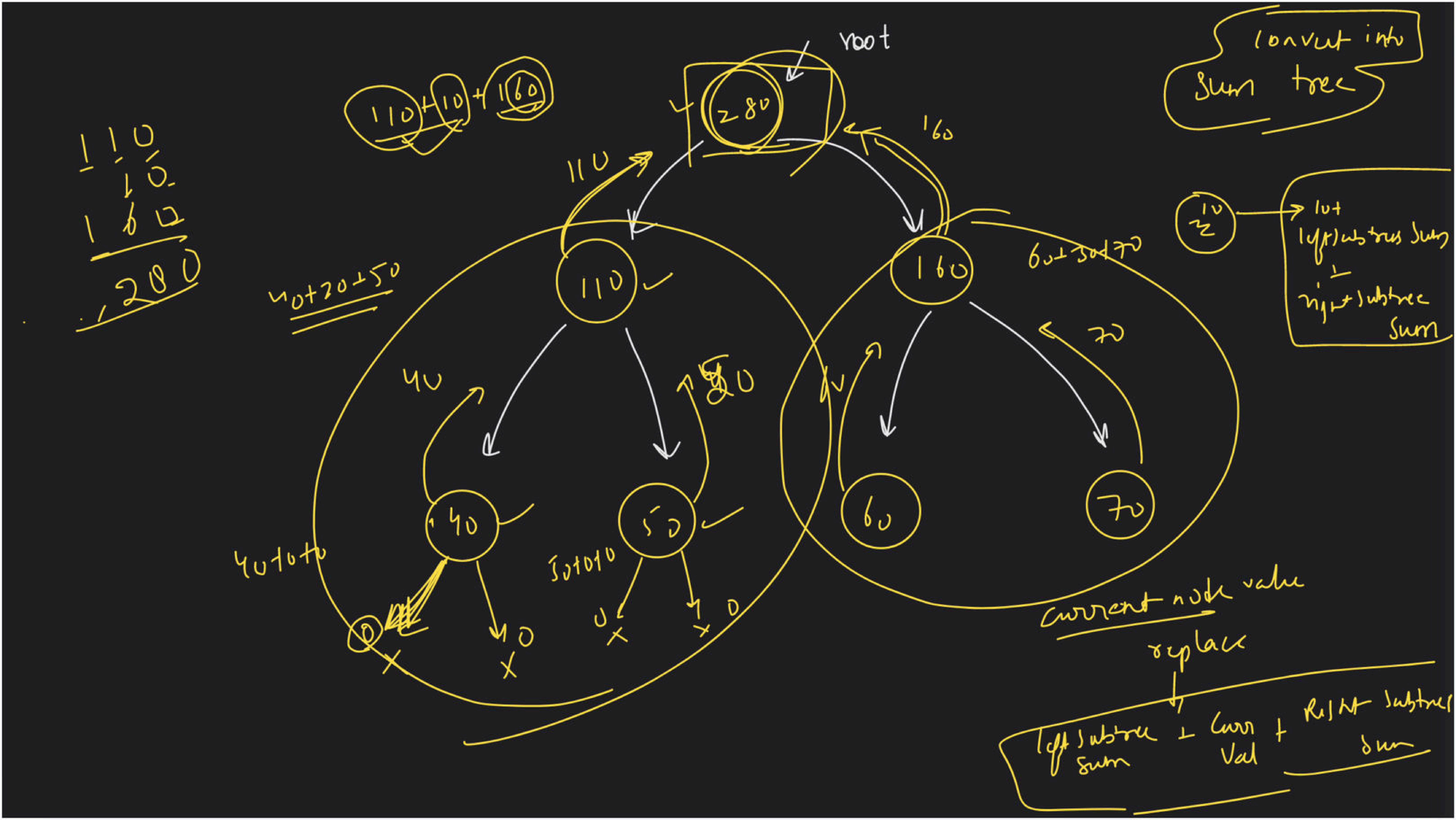
bool ans1 = (diff <= 1)

if (diff <= 1)
ans = true
else
ans = false

Check whether BT is sum tree or not

or

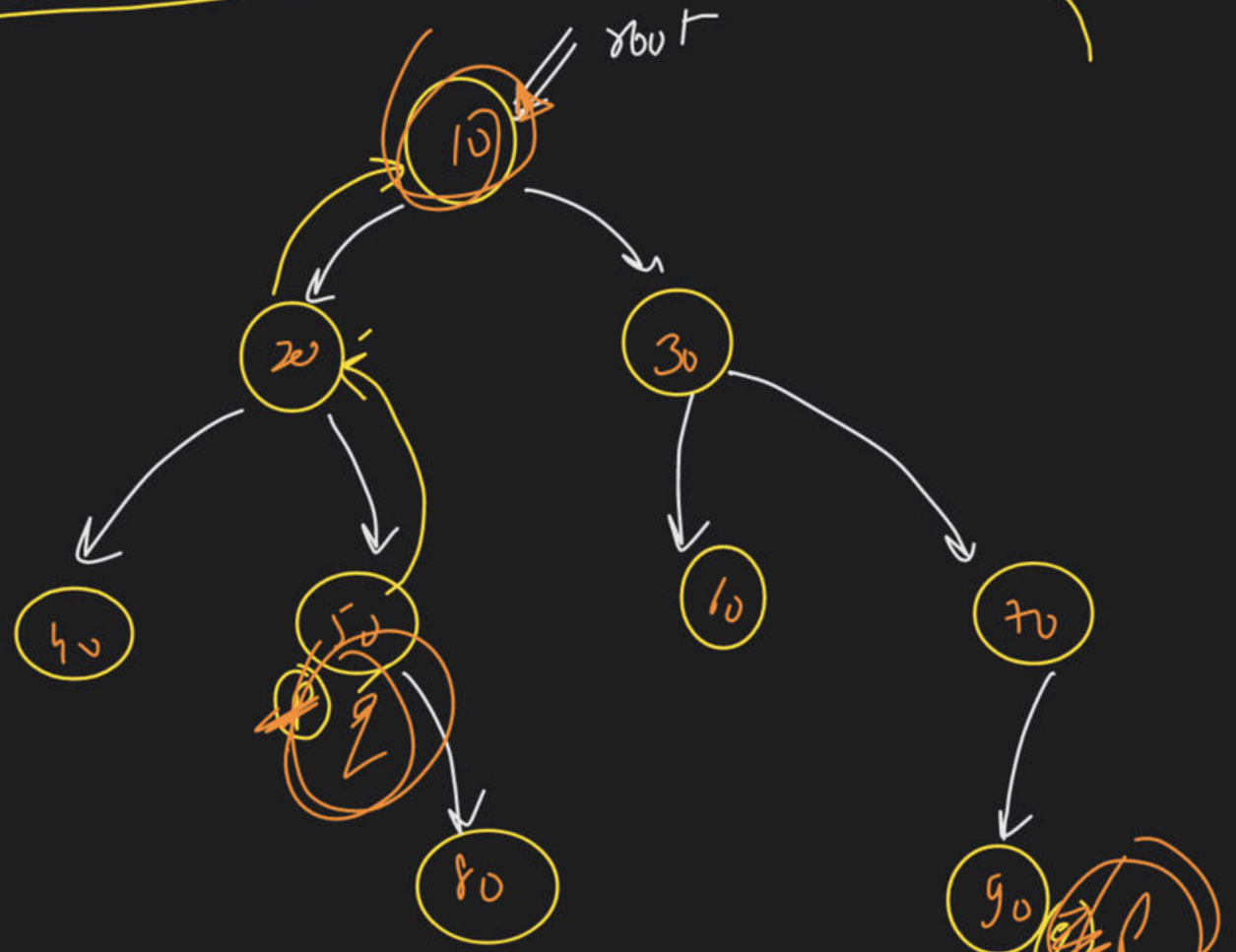
Convert BT into Sum tree


Convert into Sum tree
root
280
110
160
40
50
60
70
110+10+160
40+20+50
60+30+70
40+0+0
50+0+0
left subtree sum + right subtree sum
current node value replace
left subtree sum + curr val + Right subtree sum

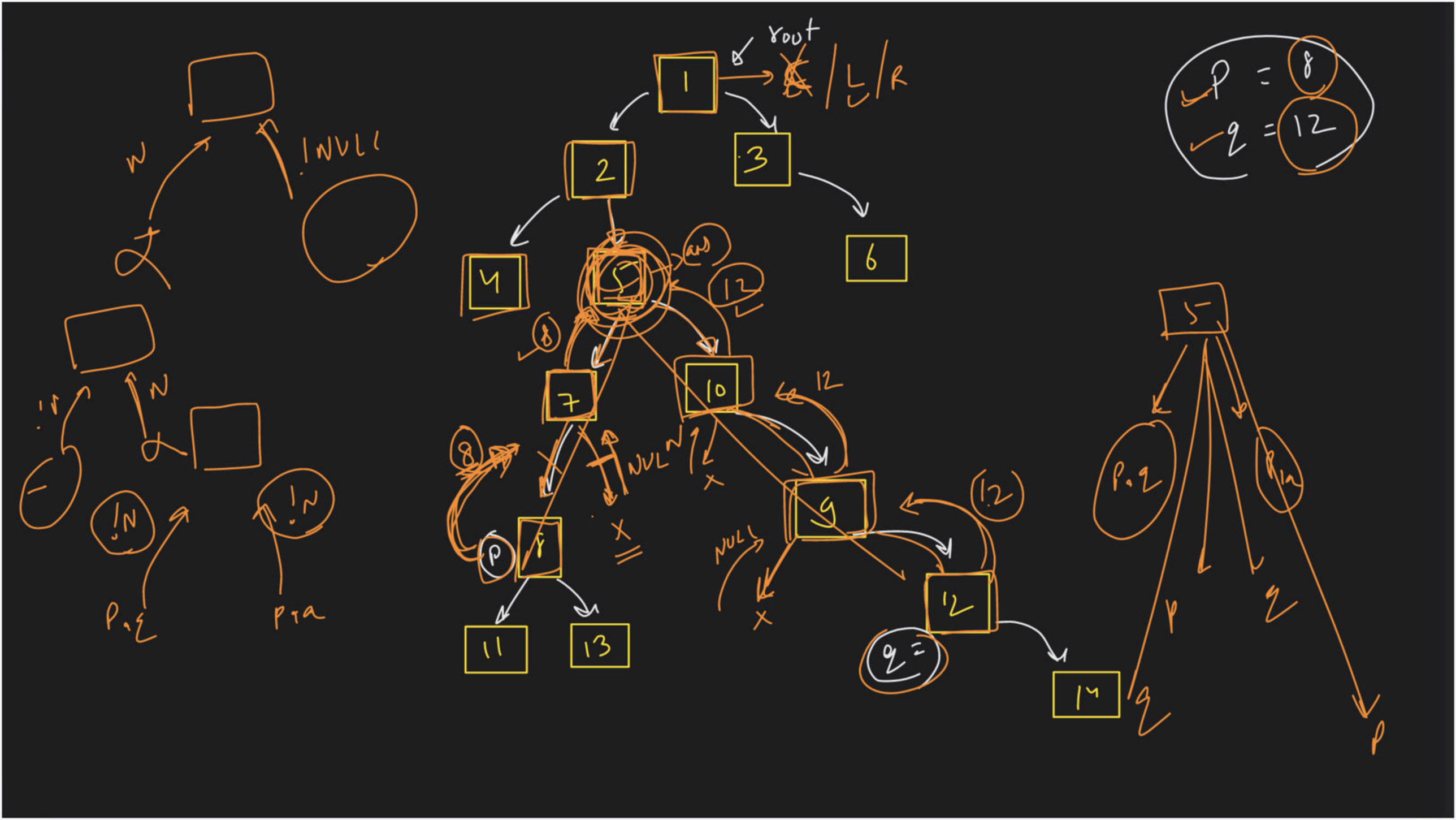
2 min

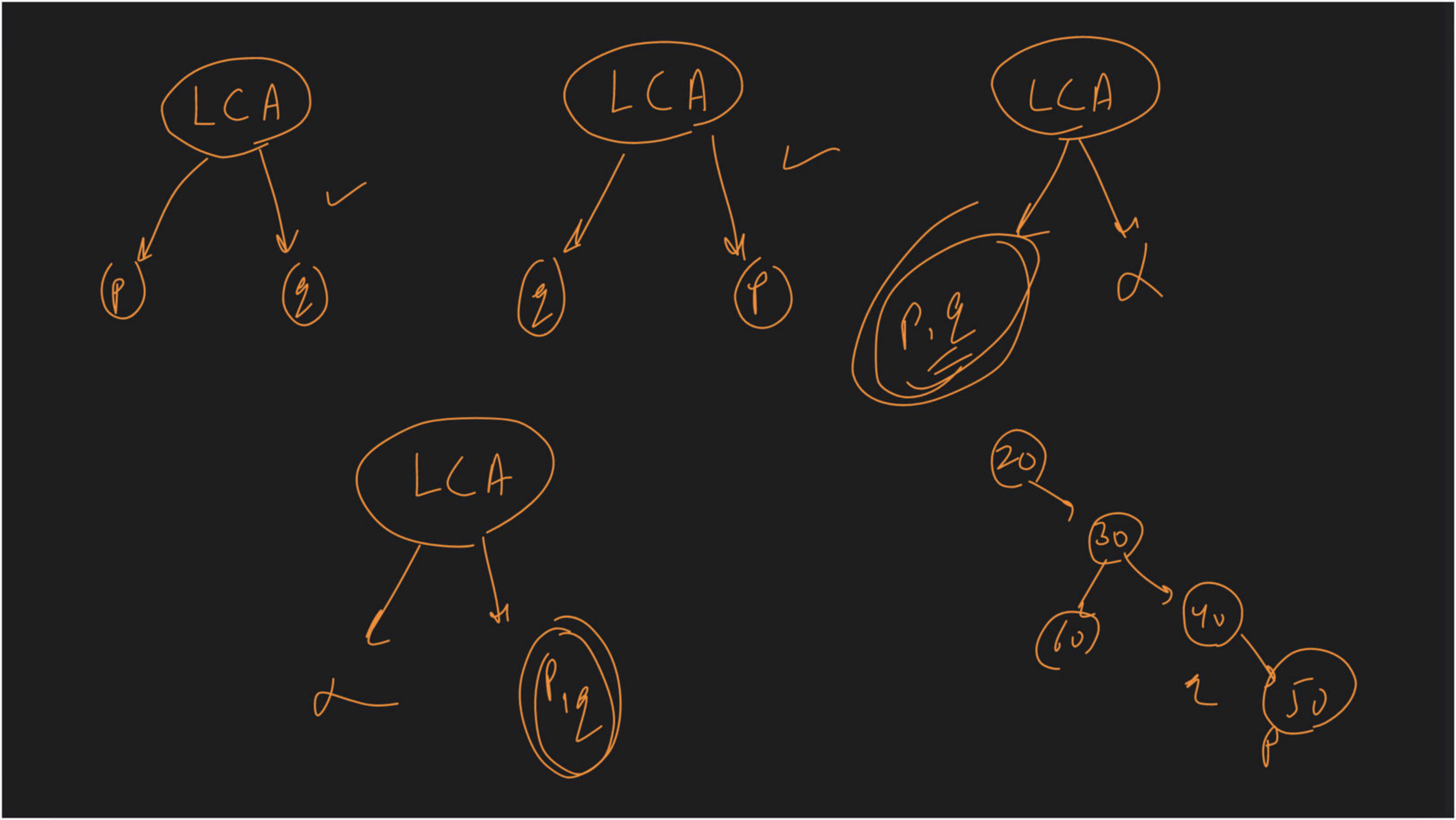
```
convertIntoSumTree(root)
{
    // BC
    if(root == NULL)
        return 0;

    int leftSum = convertIntoSumTree(root->left);
    int rightSum = convertIntoSumTree(root->right);

    root->data = leftSum + root->data + rightSum;

    return root->data;
}
```

Lowest Common Ancestor:

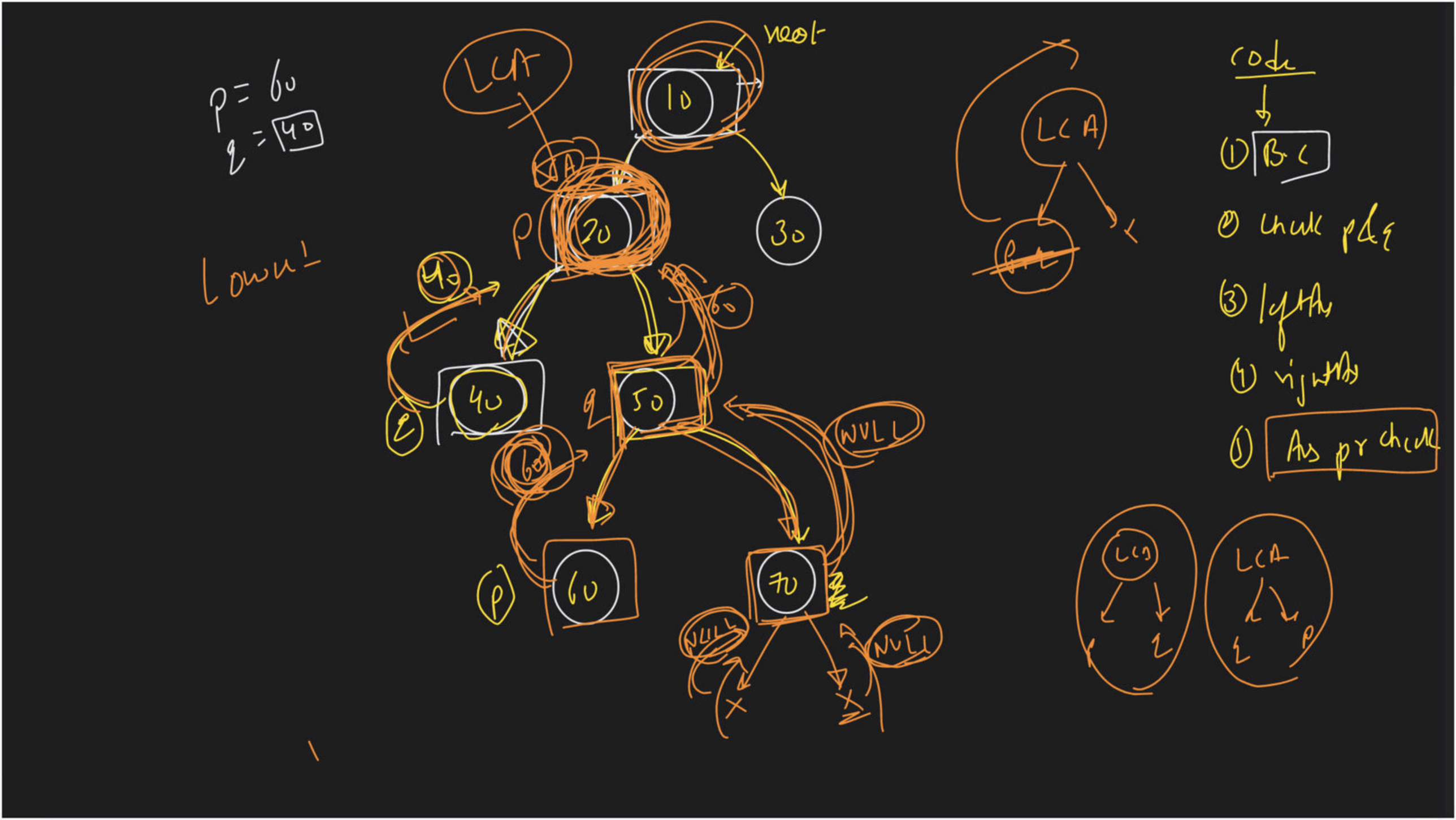


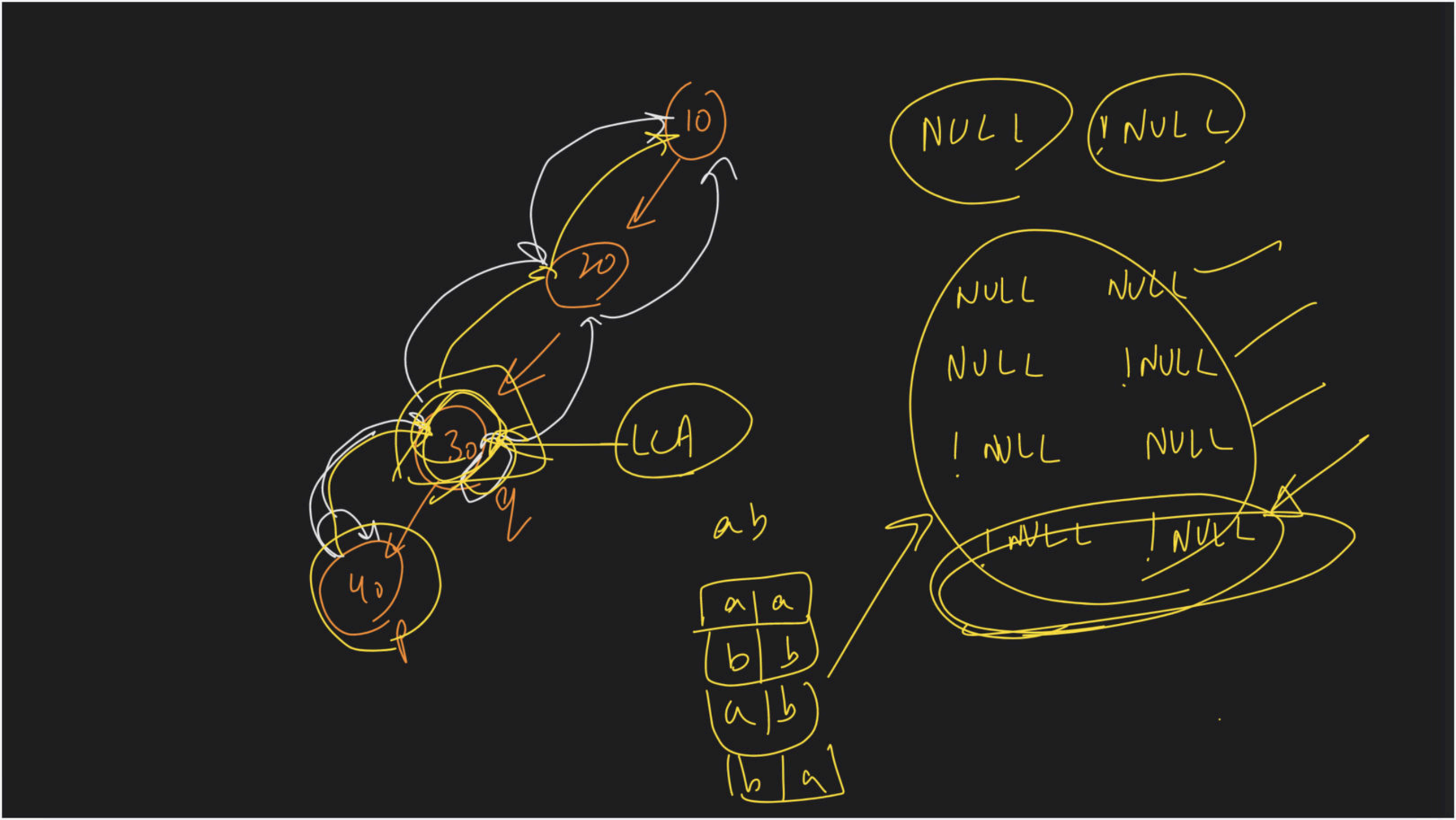
p = 50
q = 90

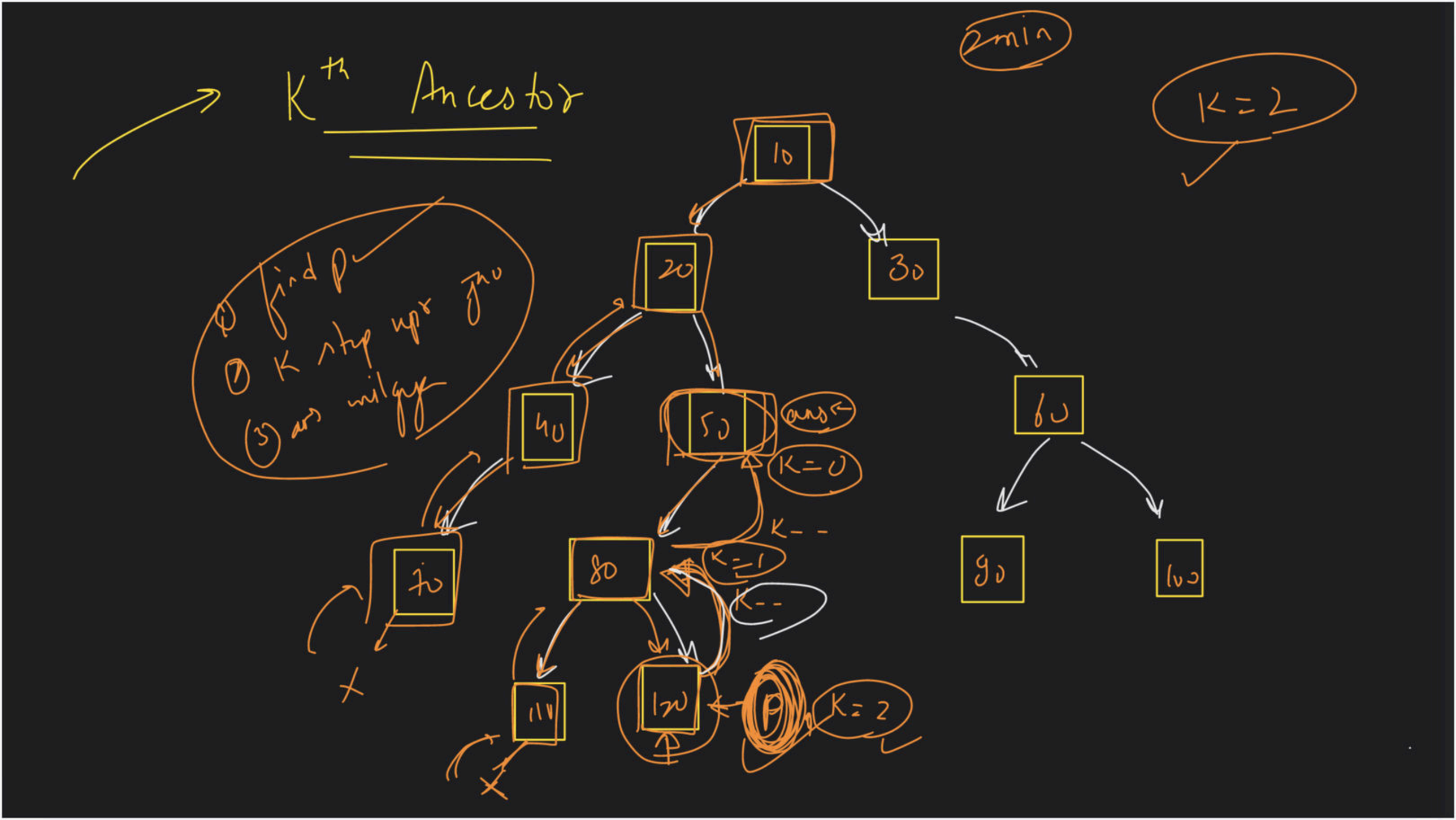

root
1
2
3
4
5
6
7
8
9
10
11
12
13
14
NULL
p = 8
q = 12
p, q
p
q


LCA
p
q
LCA
q
p
LCA
p, q
LCA
p, q
20
30
40
60
50


10
20
40
40
50
30
p
L


p = 60
q = 40
Lowest
LCA
root
10
A
p
20
30
40
60
40
50
q
60
NULL
p
60
70
NULL
NULL
LCA
code
1) B.C
2) check p & q
3) if
4) right
5) Ans pr check
LCA
LCA

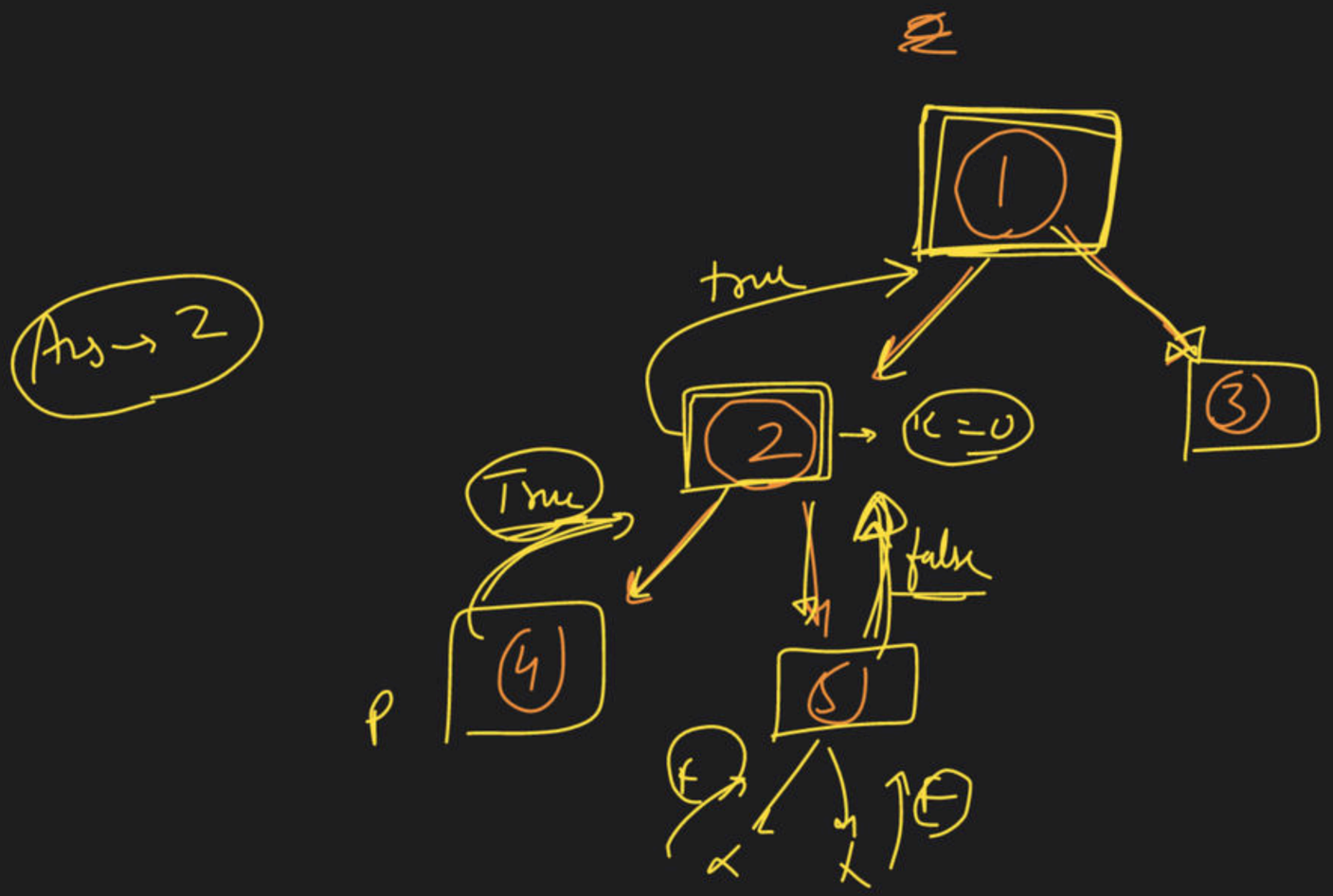

10
20
30
40
LCA
p
q
NULL
!NULL
NULL NULL
NULL !NULL
!NULL NULL
!NULL !NULL
ab
a a
b b
a b
b a

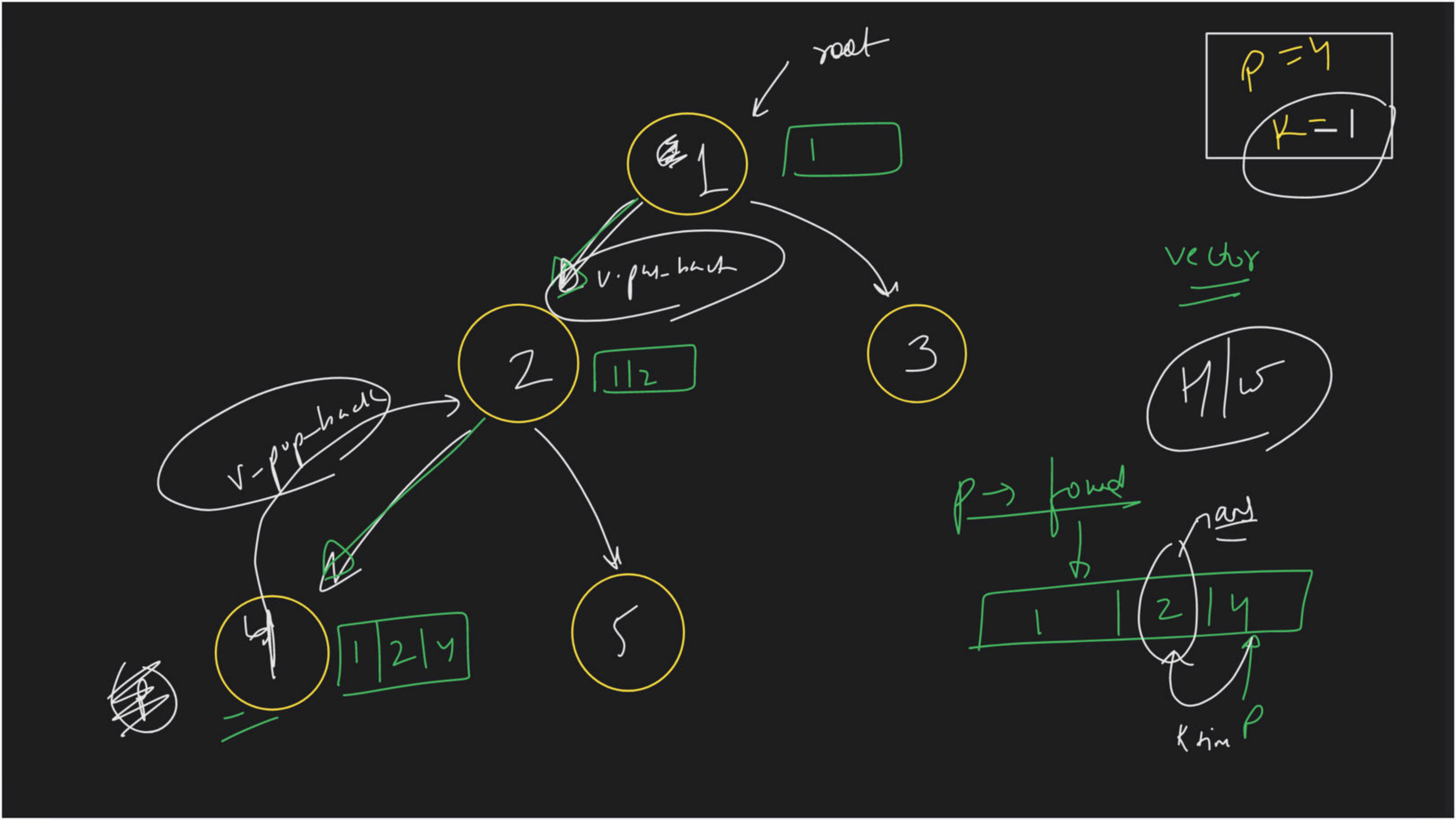

K th Ancestor
2min
K=2
1) find p
2) K step upr jao
3) ans milgya
10
20
30
40
50
60
70
80
90
100
110
120
ans=
K=0
K--
K=1
K--
p
K=2
x
x

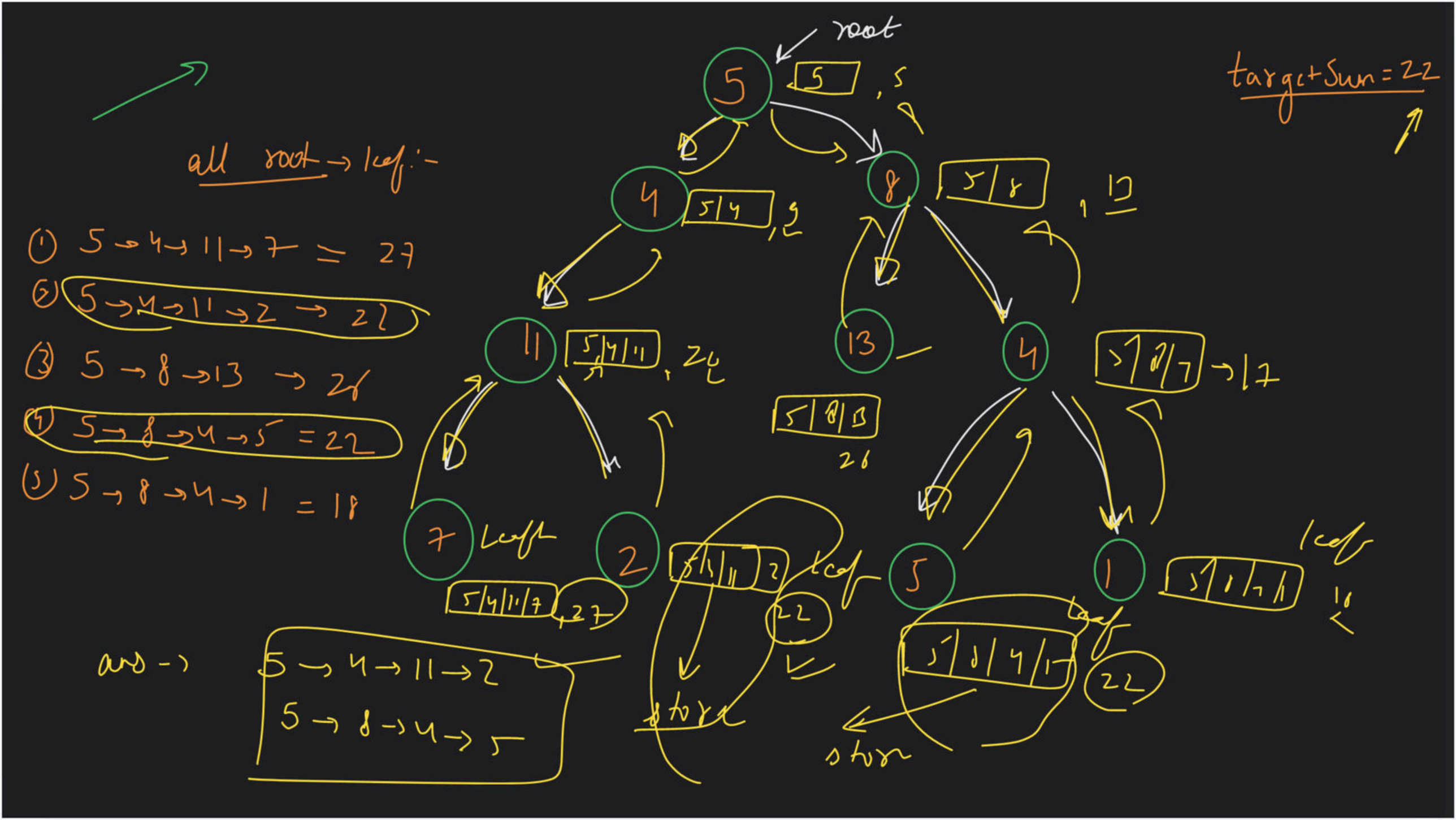
```
bool kthAncestor(root, int &k, node *p)
{
    //B.C
    if (root == NULL)
        return false;

    if (root->data == p->data)
        return true;

    bool leftAns = kthAncestor(root->left, k, p);
    bool rightAns = kthAncestor(root->right, k, p);

    if (leftAns == true || rightAns == true)
    {
        k--;
        if (k == 0)
        {
            cout << root->data;
        }
        return leftAns || rightAns;
    }
}
```

K = 1

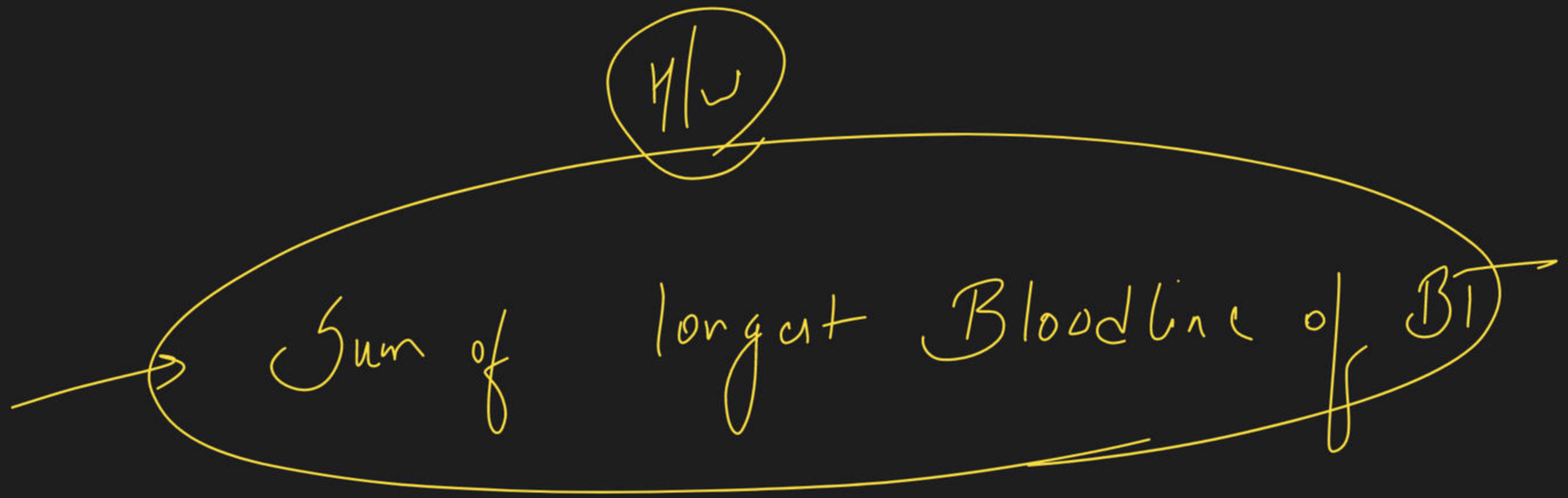
P = 4




root
1
v.push_back
2
3
v.push_back
4
5
1 | 2 | 4
p = 4
k = 1
vector
1 | 2 | 4
p → found
arr
Krim
p


all root → leaf :-
① 5 → 4 → 11 → 7 = 27
② 5 → 4 → 11 → 2 → 22
③ 5 → 8 → 13 → 26
④ 5 → 8 → 4 → 5 = 22
⑤ 5 → 8 → 4 → 1 = 18
root
target Sum = 22
leaf
leaf
store
store
ans →
5 → 4 → 11 → 2
5 → 8 → 4 → 5


targetSum = 20
path -> any node
any node


H/W
Sum of longest Bloodline of BT