Doubt Clearing Session - Part XIII

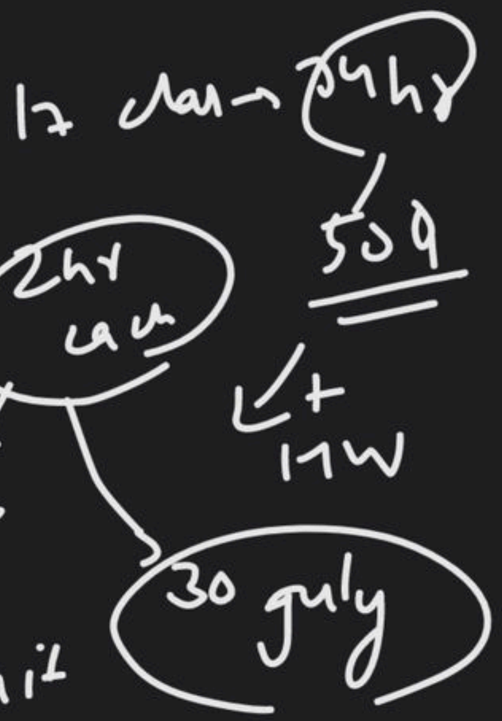
Foundation Course on Data Structures & Algorithm - III

Love Babbar • Lesson 13 • July 10, 2022

Doubt Session :-

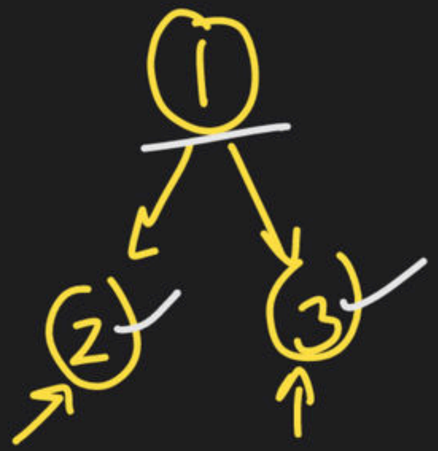
95-99%

1. Sliding Window Assignment
2. Array assignment
3. STL & BT assignment → commit
4. String → KMP algo
5. LRU Cache → code
6. 450 Cracker Sheet

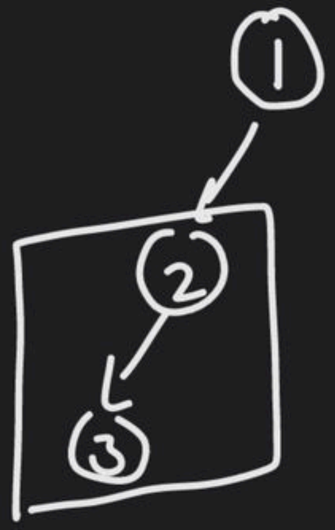


→ Doubt:-

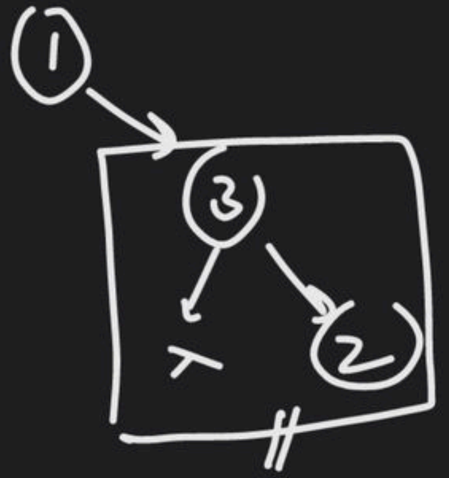
LCA → Lowest Common Ancestor



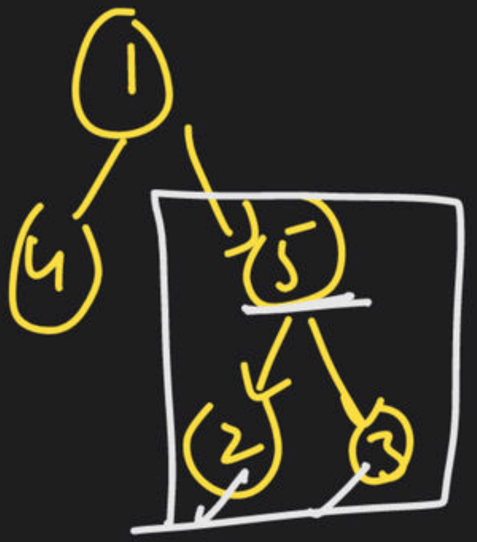
LCA(2, 3) → 1



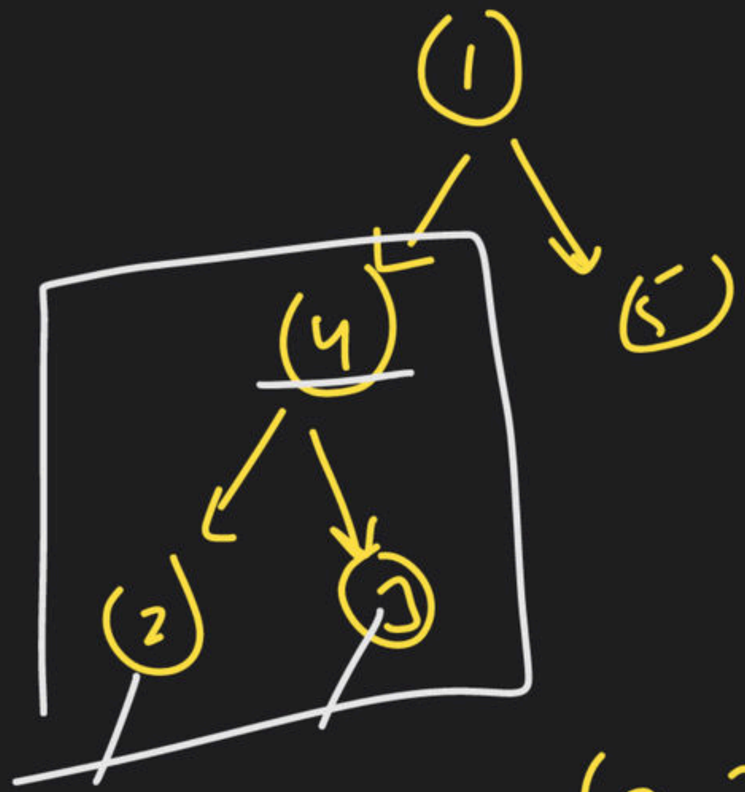
LCA(2, 3) = 2



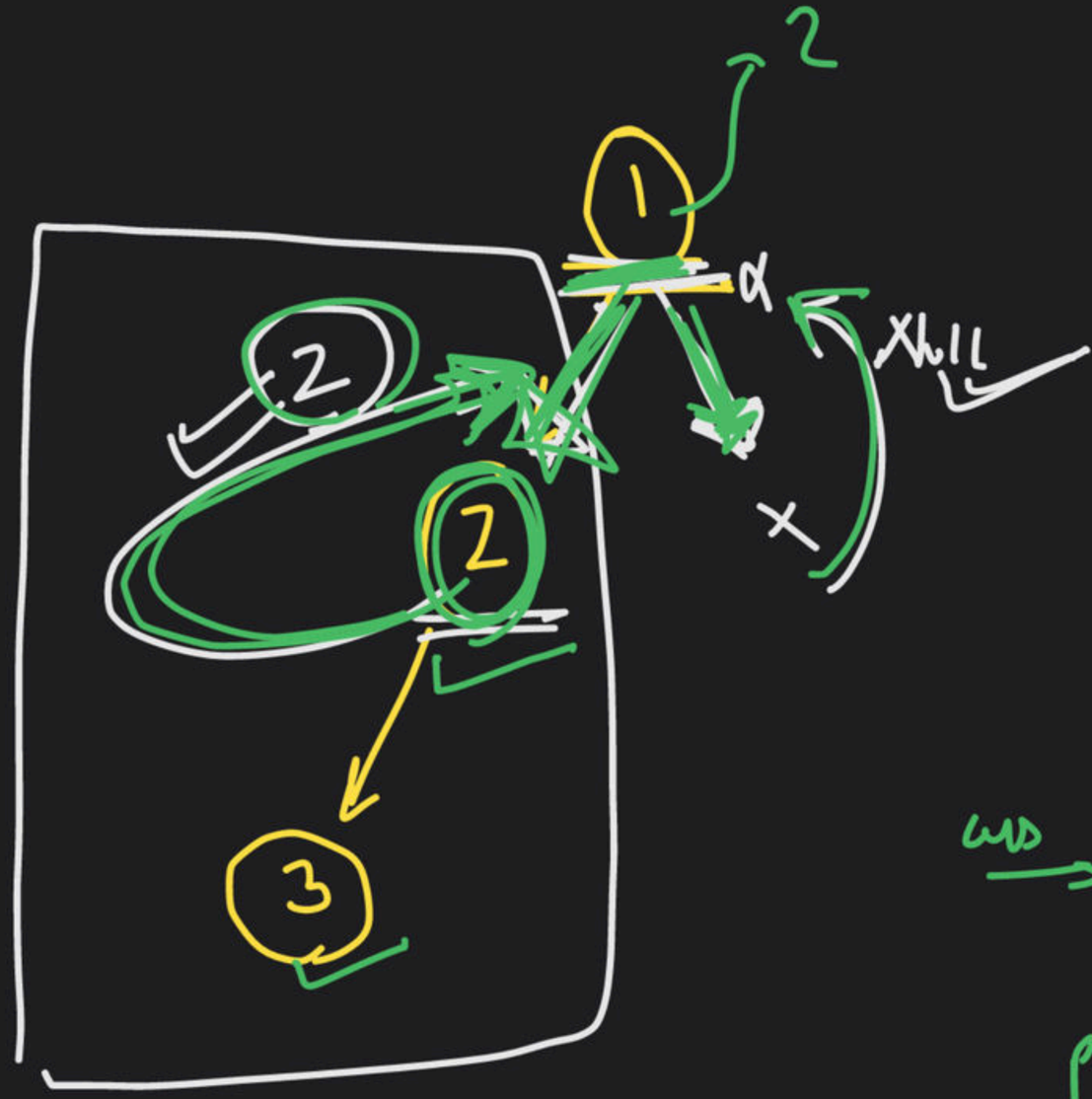
LCA(2, 3) = 3

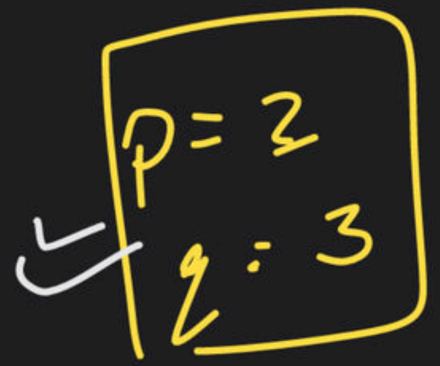


LCA(2, 3) = 5

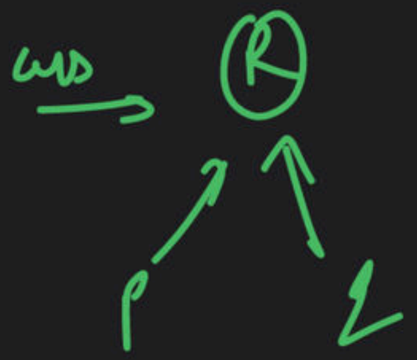


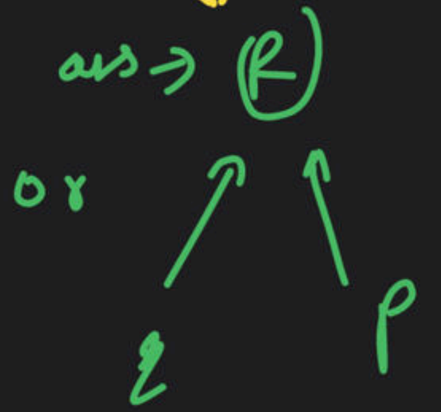
LCA (2, 3) = 4

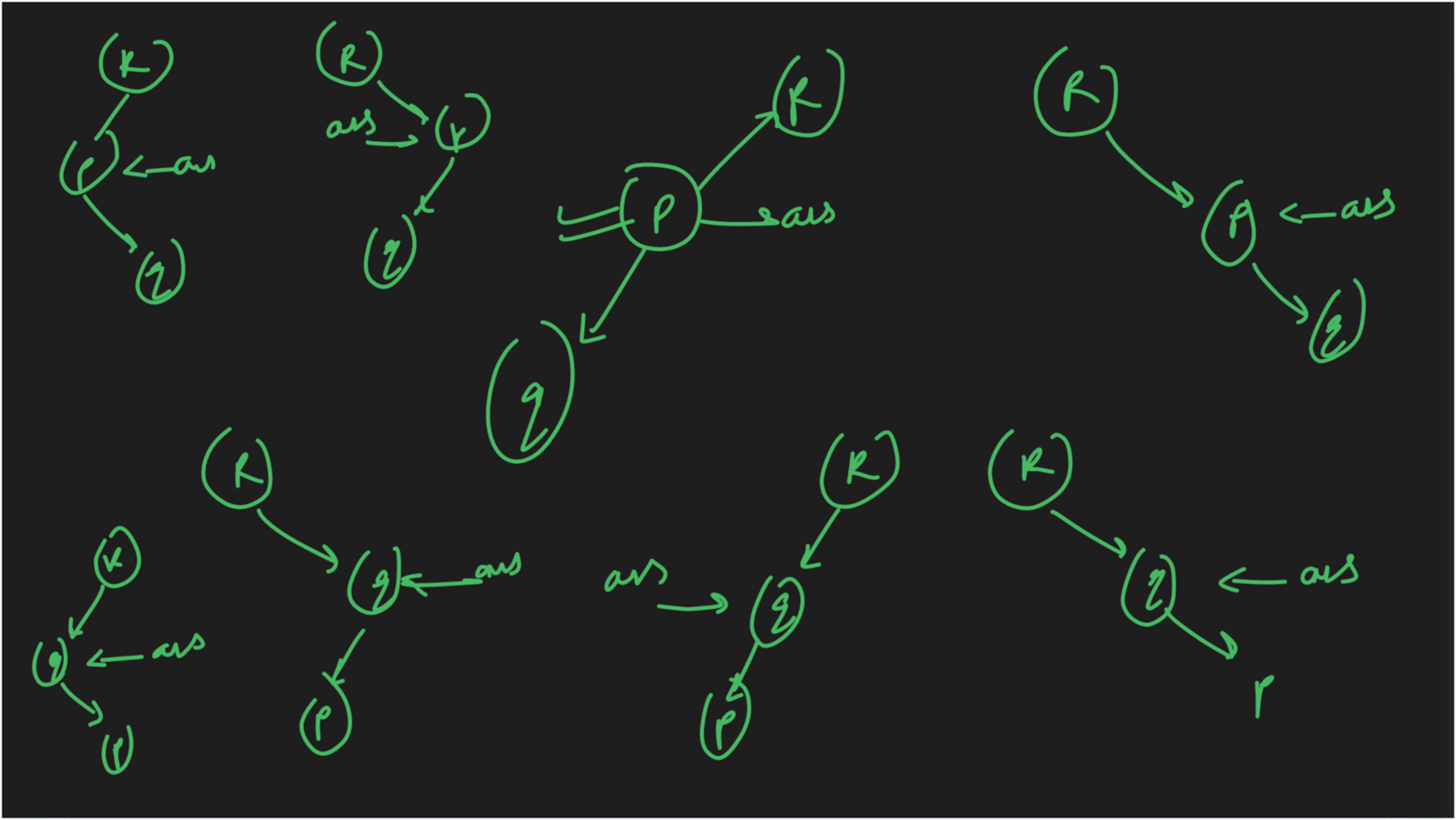

2
1
2
2
3
x
x

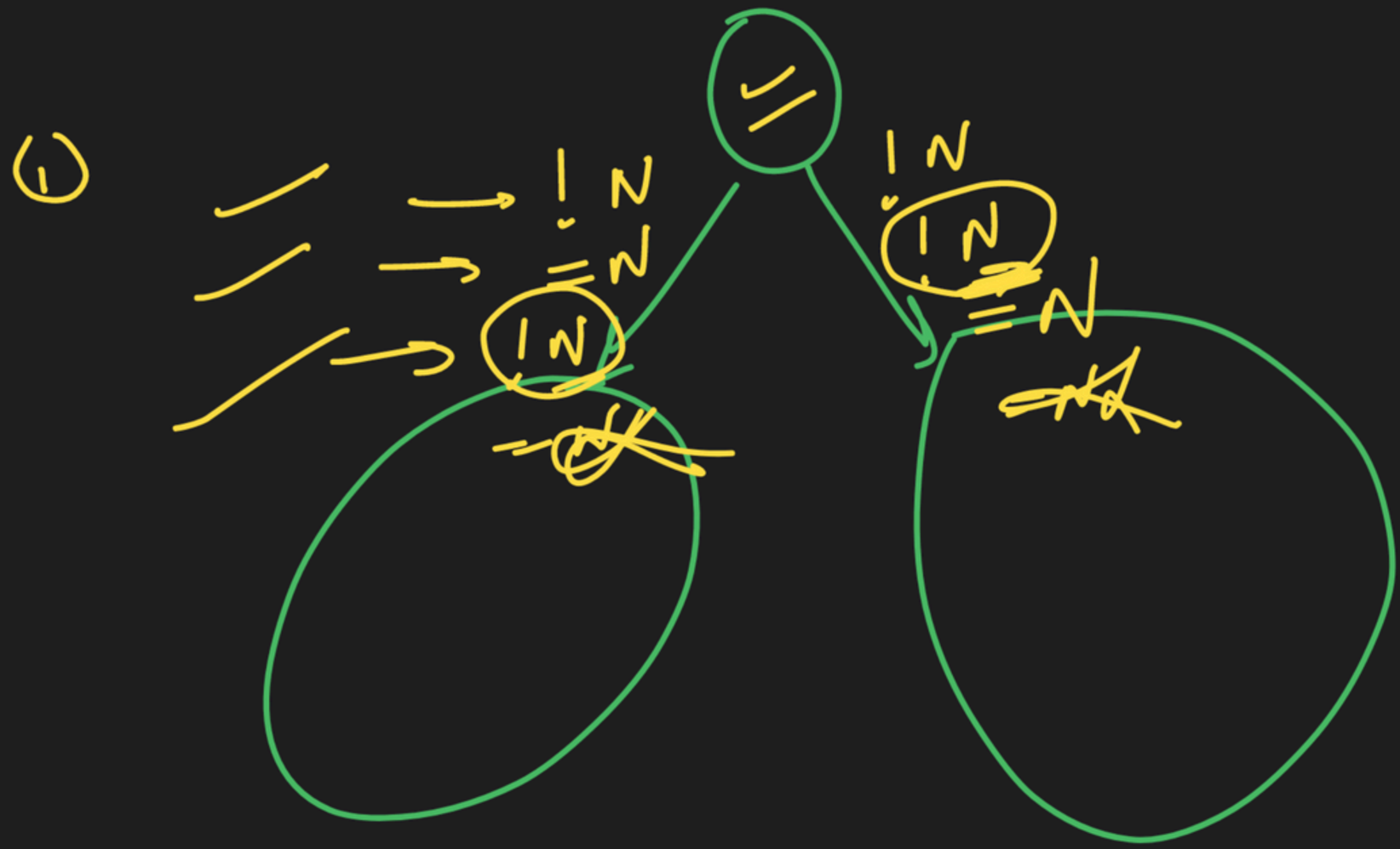

p = 2
q : 3


L(A(p, q)


ans
R
p
q


ans
R
or
q
p


K
r
aus
q
R
aus
K
z
P
K
aus
q
R
p
aus
q
K
q
aus
r
K
q
aus
p
R
aus
q
p
R
z
aus
r



IN & IN

↳ Root

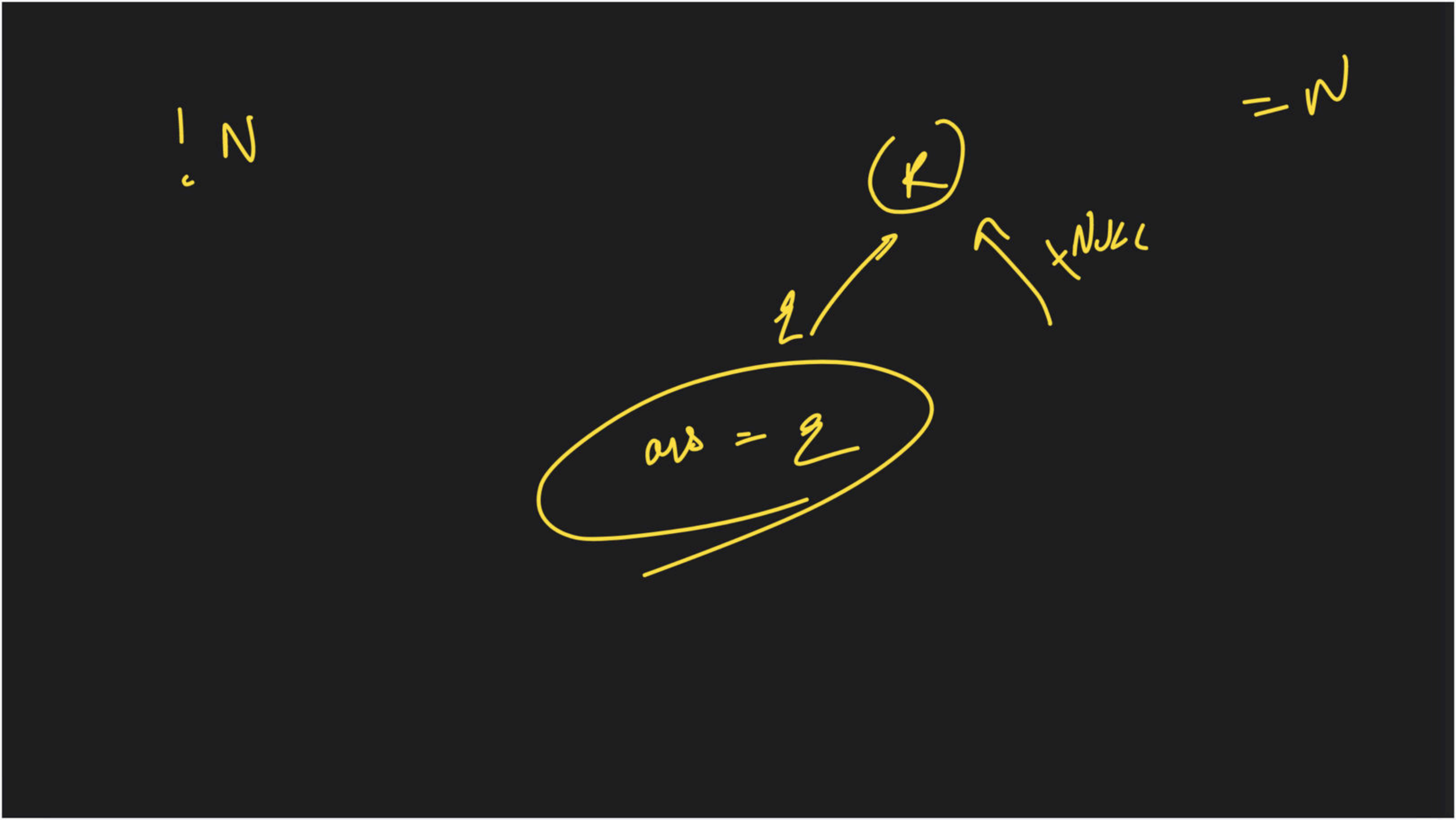
!N && !N

ans = R

$= N$

$! N$

$ans \to (r)$

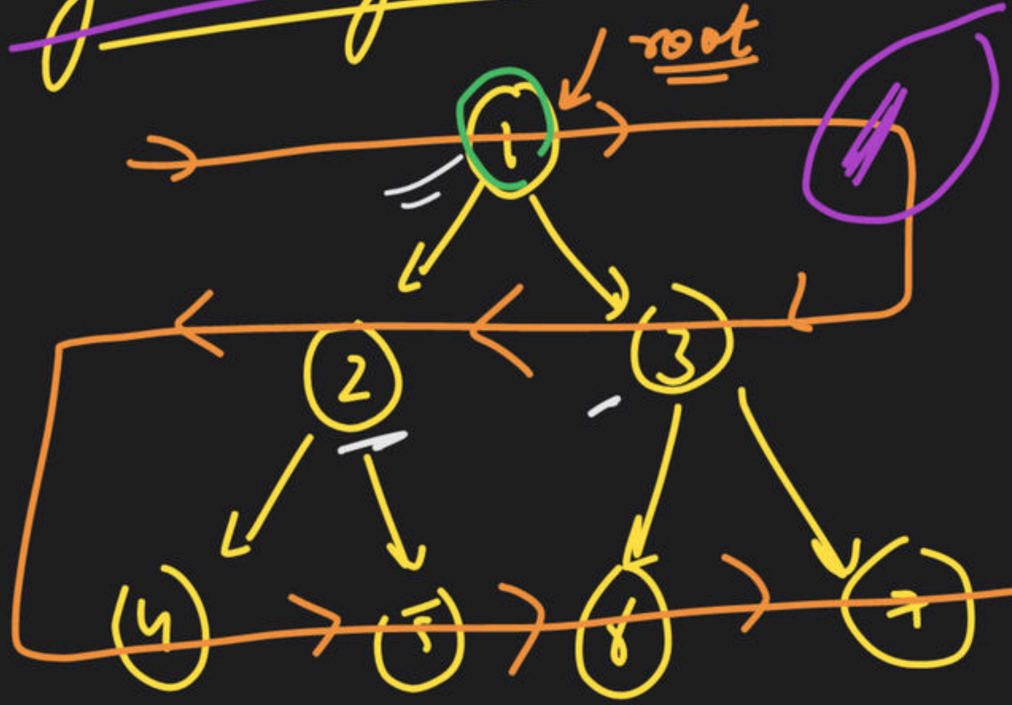

!N
=N
K
+NULL
ans = 2

```
Node * LCA ( Node* root)
{
    if (root == NULL)
        return NULL;

    if ( root == p || root == q )
        return root;

    else
    {
        Node* left = LCA (root -> left)
        Node* right = LCA (root -> right);
```

```
if (left != NULL && right != NULL)
        return root;
else if (right != NULL)
        return right;
else
        return left;
}
```

Zig - Zag traversal :-

Test case → Dry run



1 3 2

4 5 7

Lvl 0

Lvl 1

Lvl 2

```
flag = true;
```

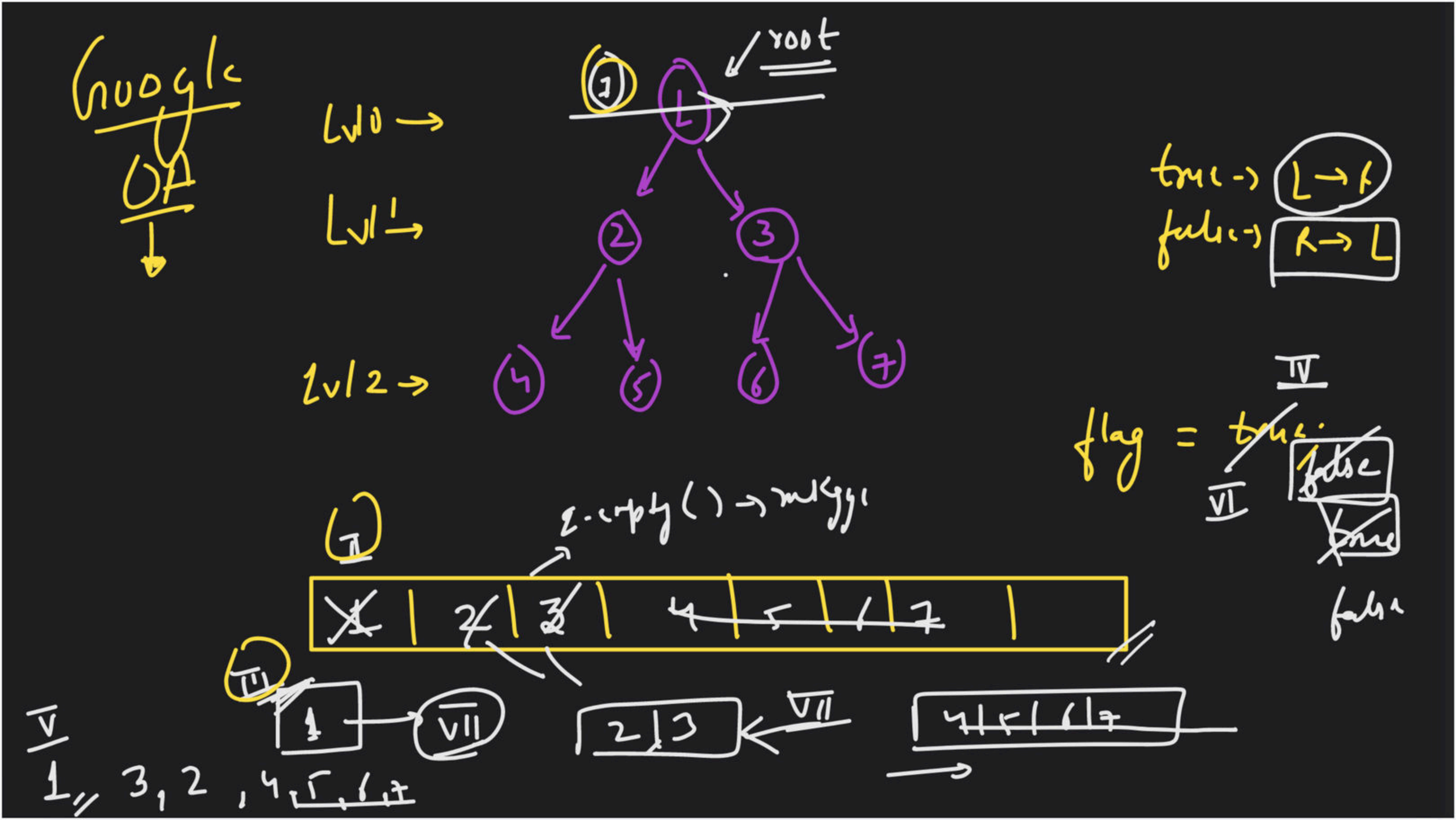
solution :-

interview :-

ratta

Google

OA


Lvl 0 →
Lvl 1 →
Lvl 2 →
root
1
2
3
4
5
6
7
true → L → R
false → R → L
flag = true
false
II
III
IV
V
VI
VII
1 // 3, 2, 4, 5, 6, 7

Google OA

↳ (1)

Vector → "K" part me divide Karo

each part → "n" → attach element

part → first index - 1 (even)

Last index → (odd)

How to Solve?

BT

Split a strings into max no. of unique substring

H/W → High Priority

(2) trie, try

i/p → vector of strings

{aab, aac}

o/p →

{5, 5}

Trie

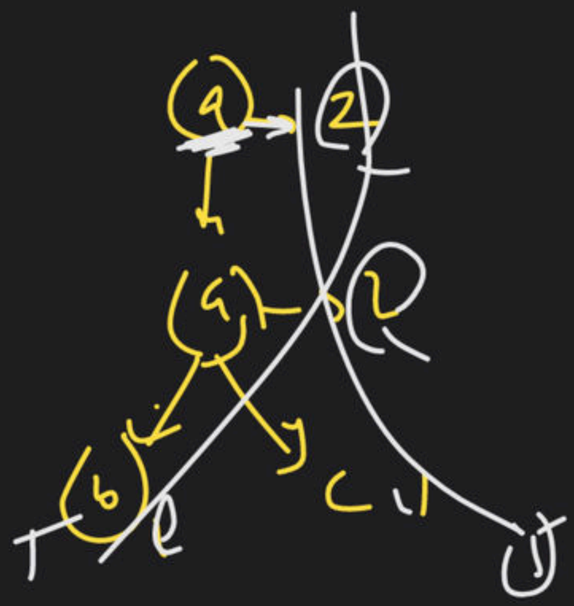
trie

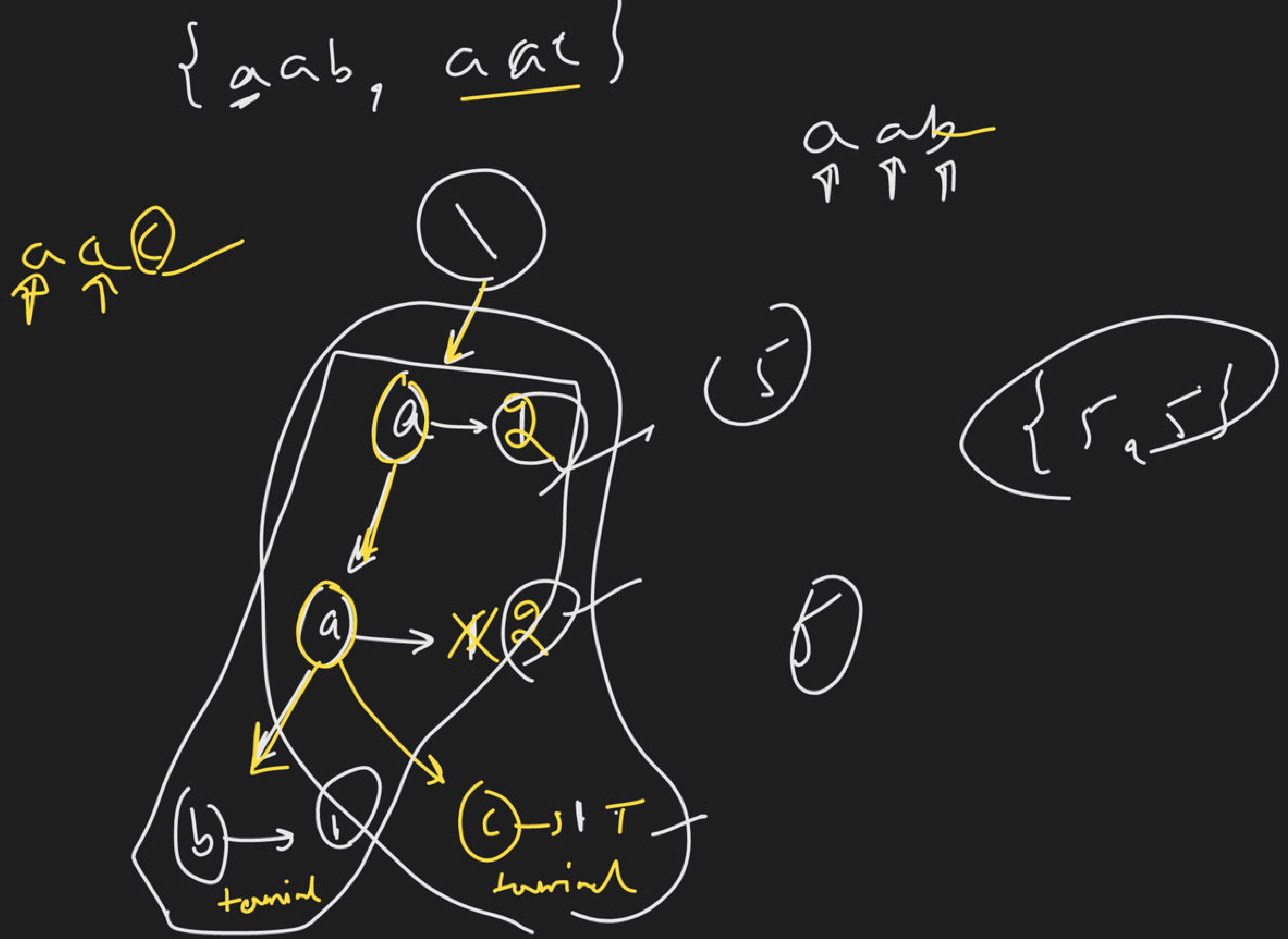
college

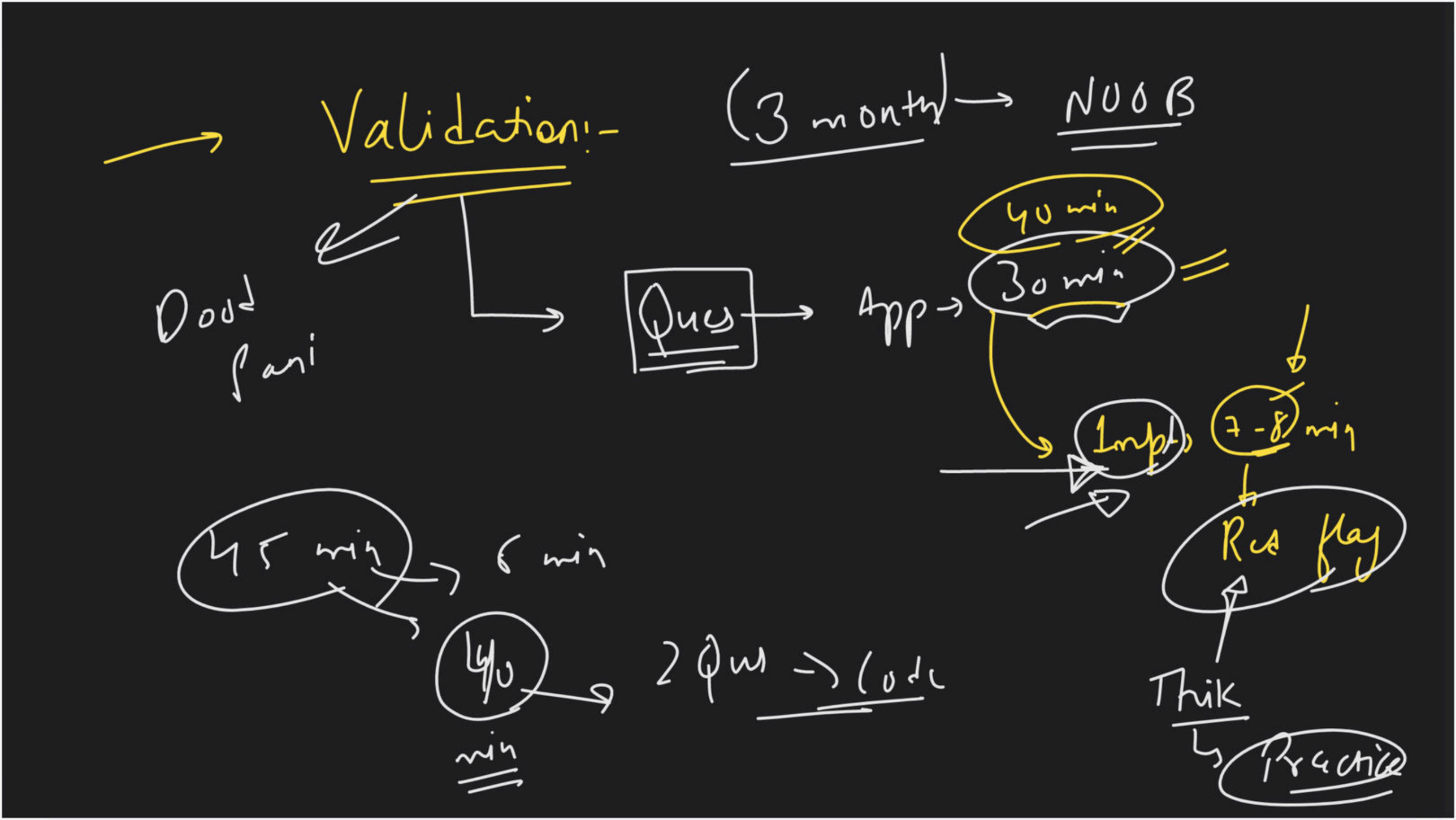
MNIT

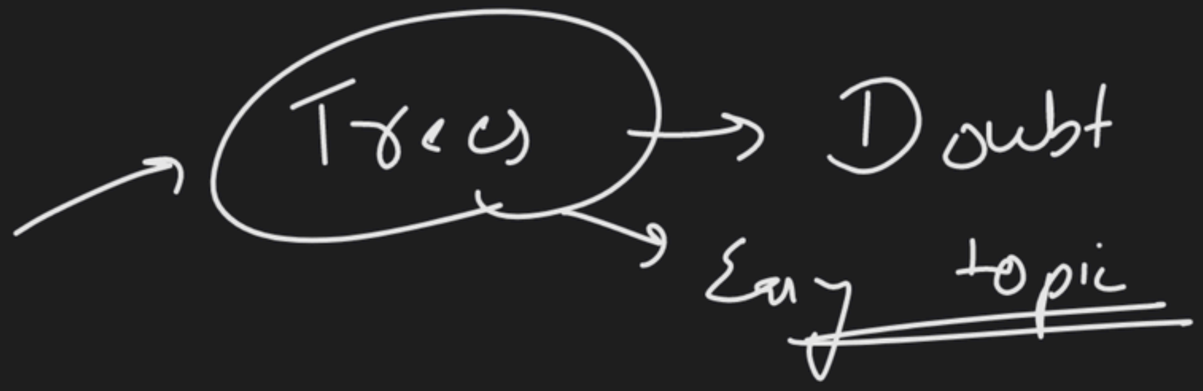
aab → {a, aa, aab}

aac → {a, aa, aac}

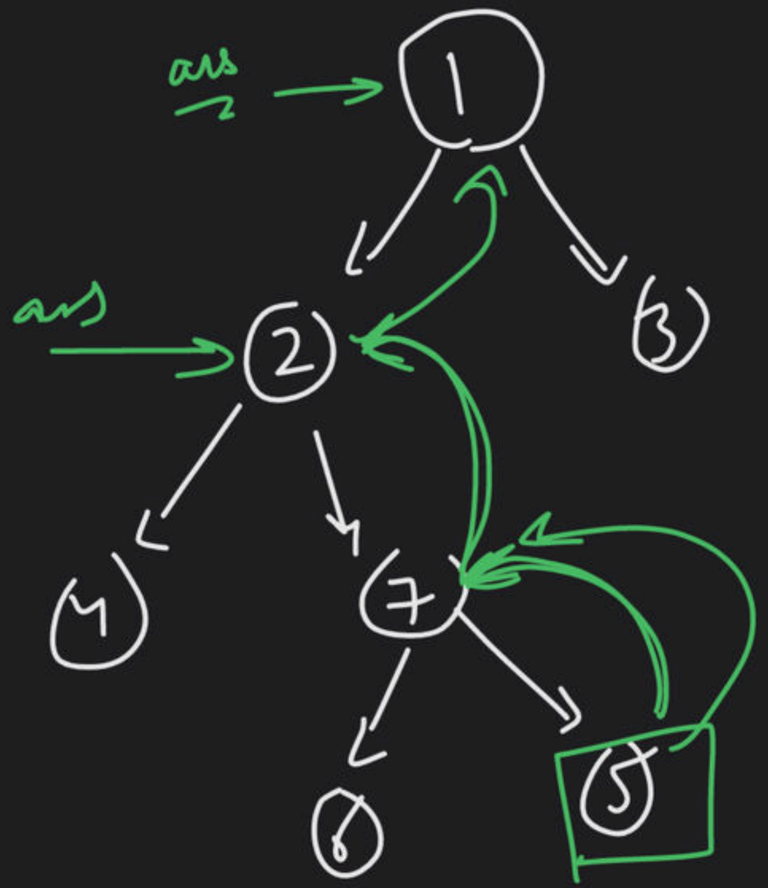



{aab, aac}
aac
aab
a
a
b
c
terminal
terminal


Validation:-
(3 months) → NOOB
Doubt
Pani
Ques
App →
40 min
30 min
Impl.
7-8 min
Real Play
45 min
6 min
40 min
2 Ques → Code
Thik
Practice


Trees
Doubt
Easy topic

K^{th} Ancestor :->



node = 5

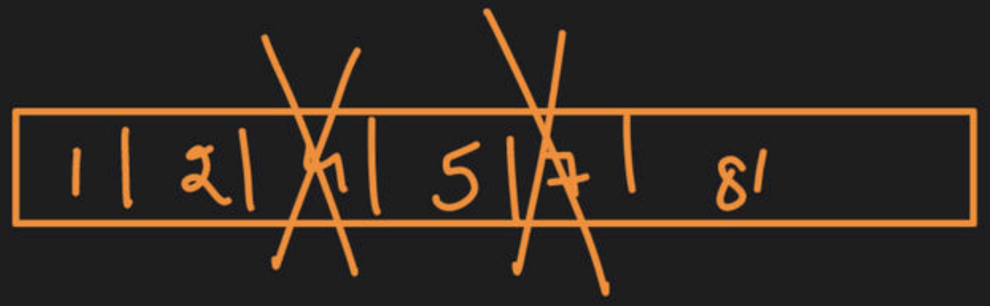
K = 2

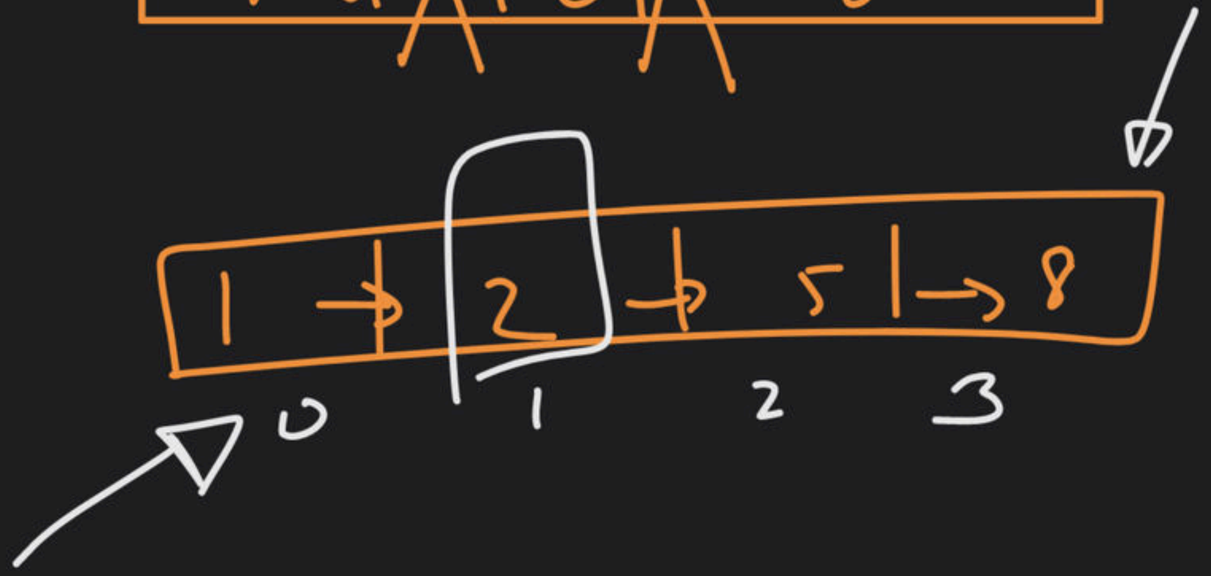
k^{th} Ancestor

K = 3

K = 1

ans = 7



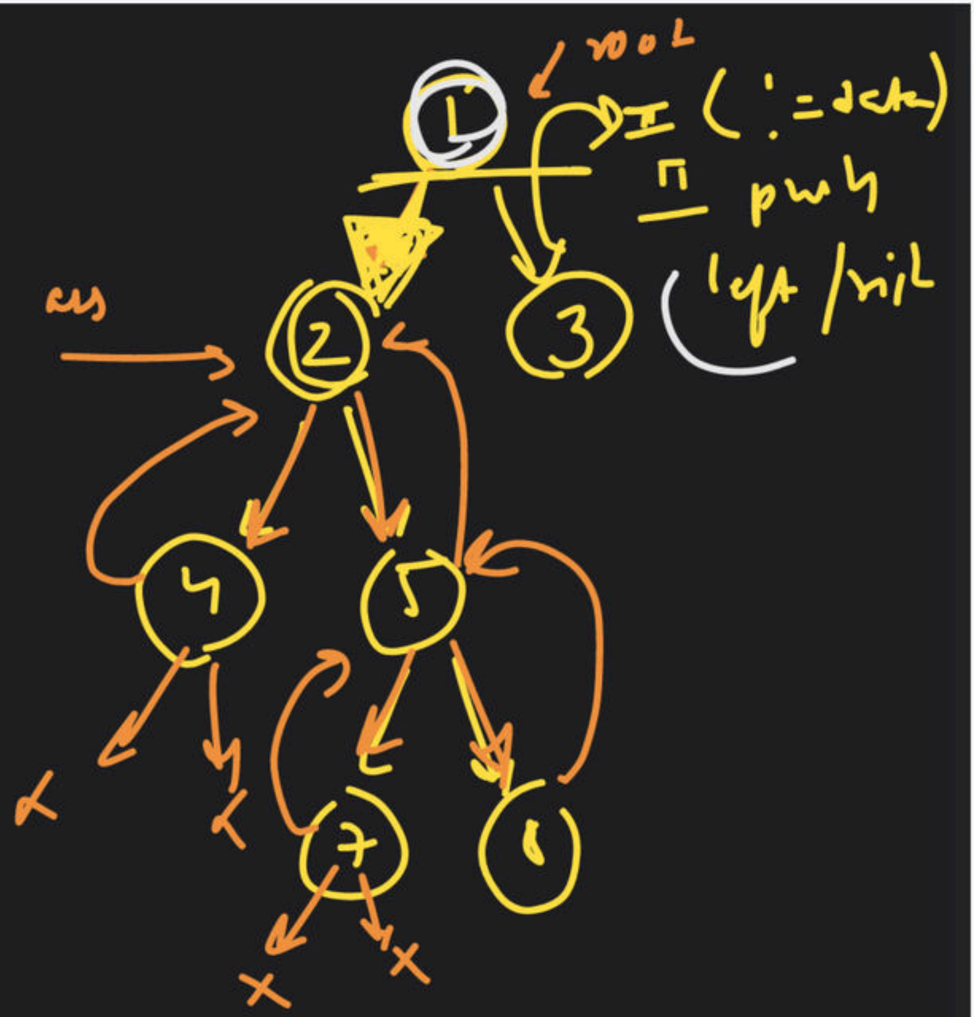


K^{th} ancestor

answer index

size = 4

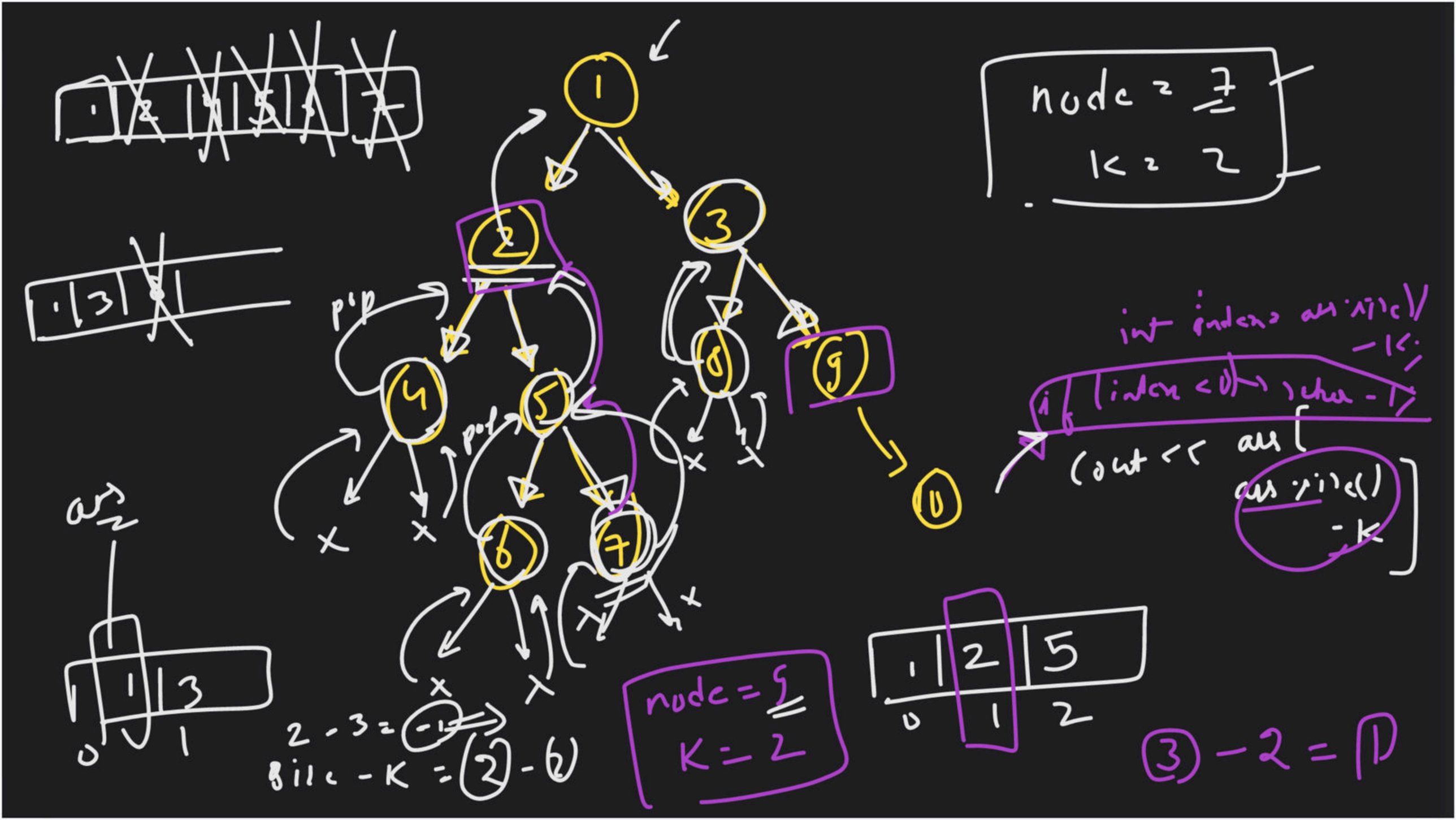
```
vec.size() - K - 1
```

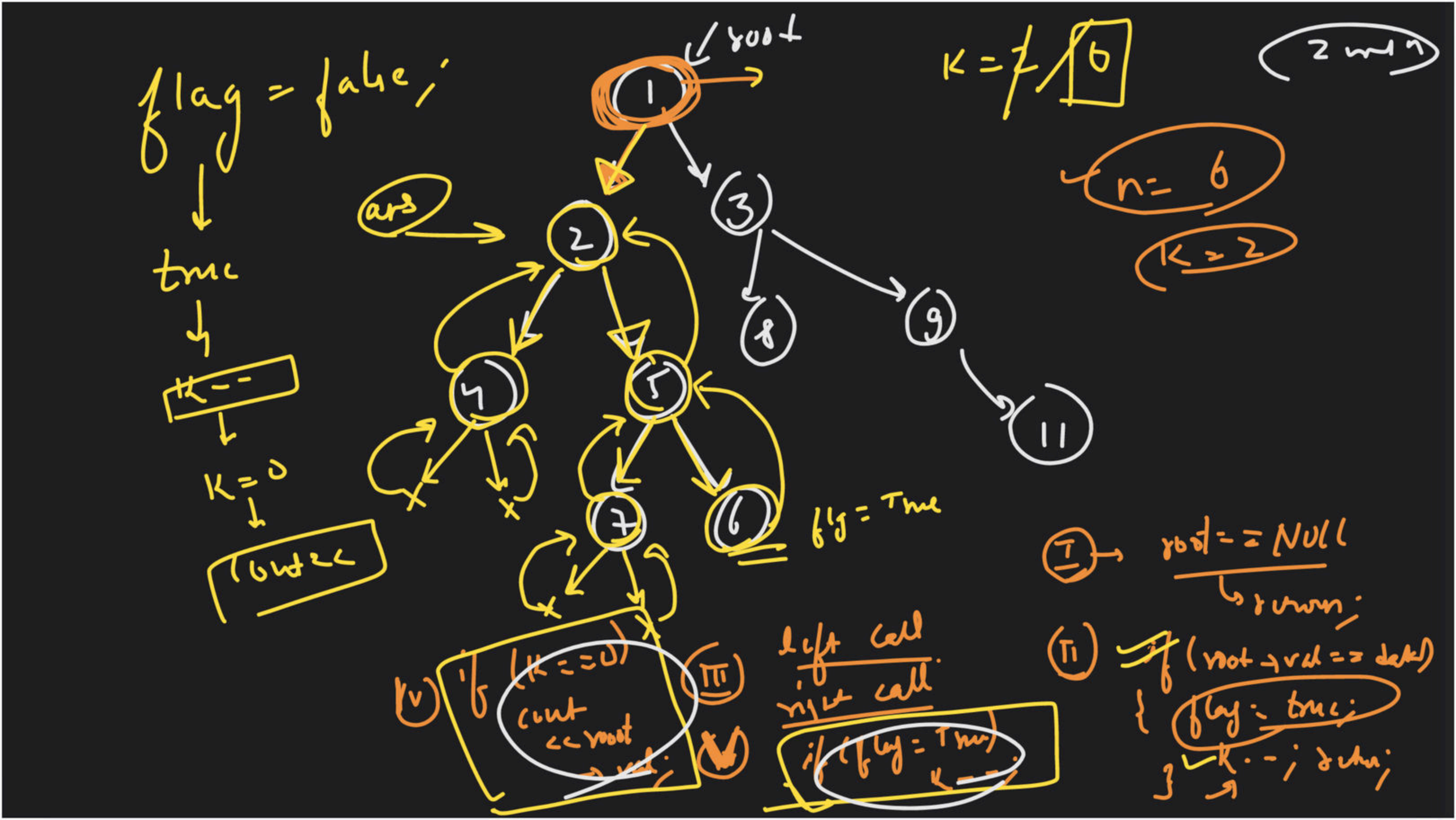


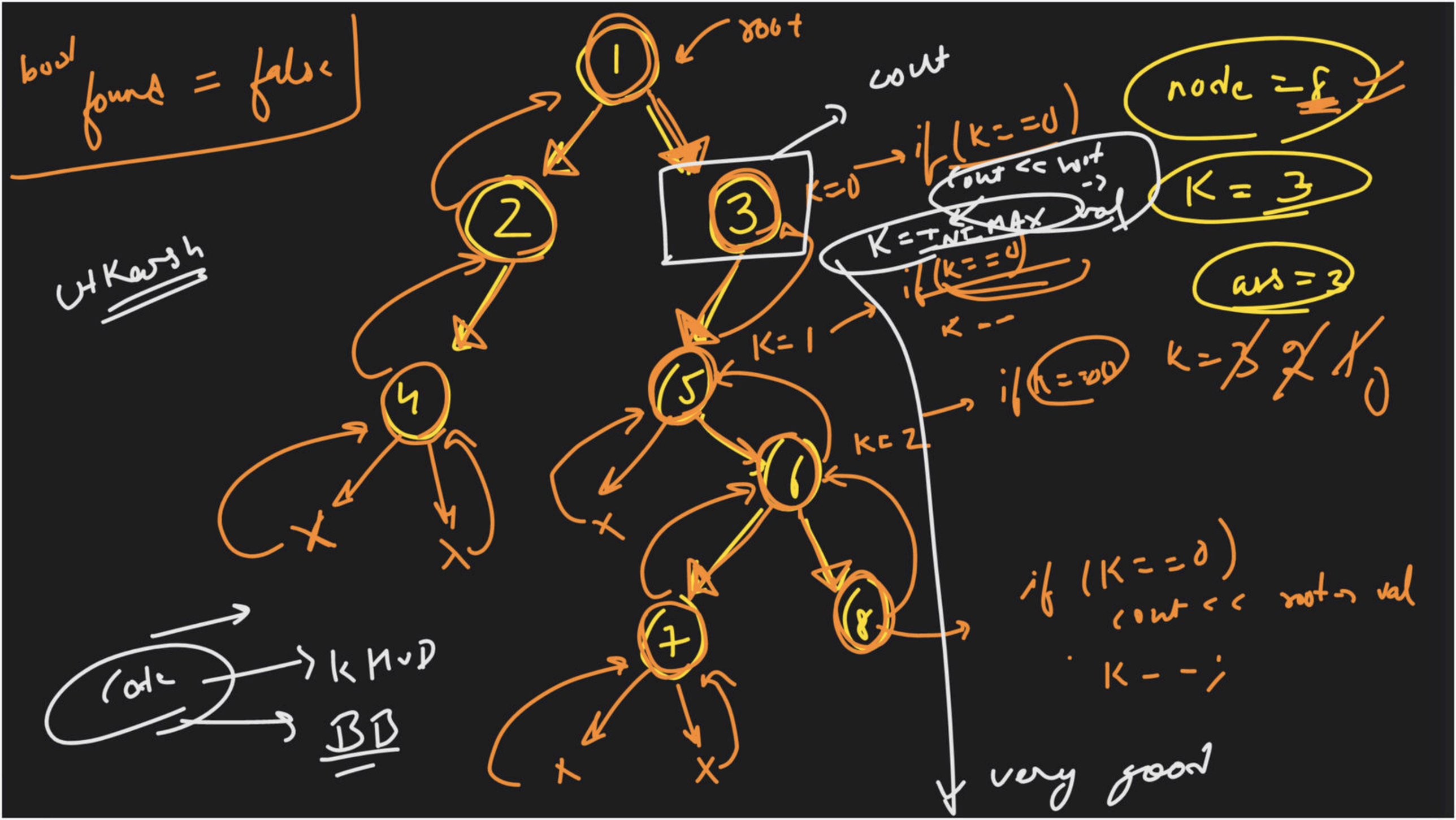
node = 8

K = 2

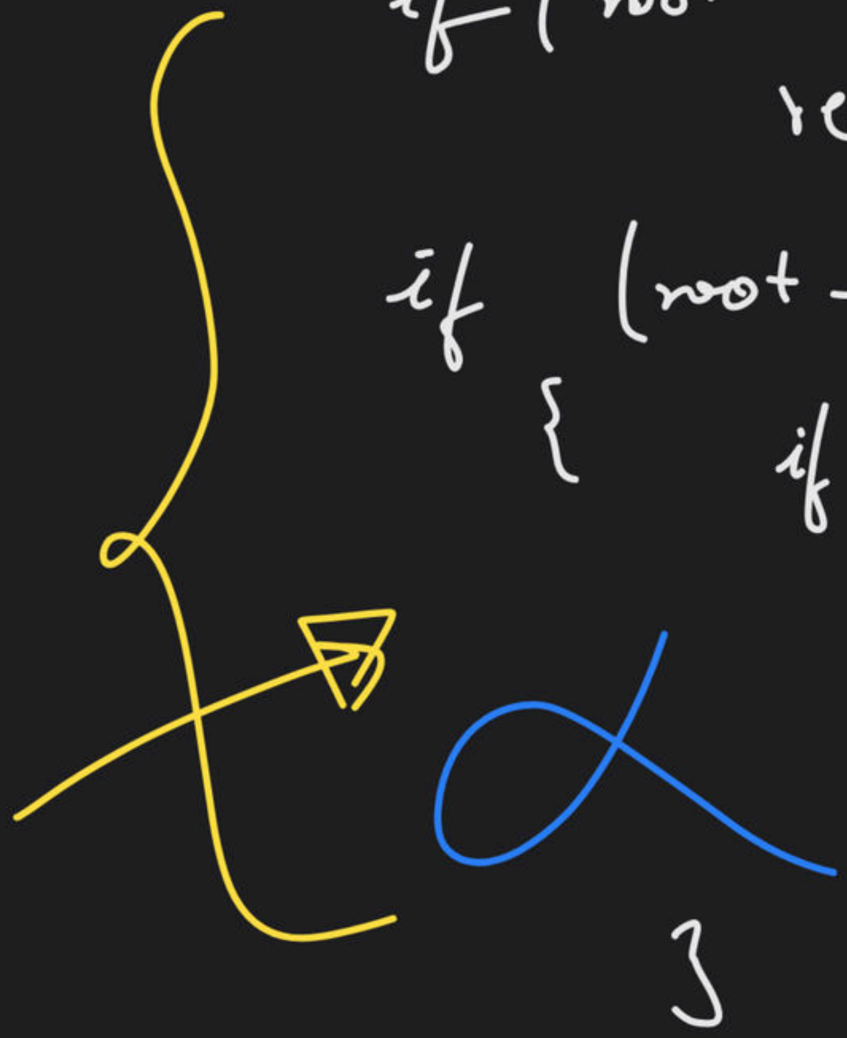
```
void solve (Node * root, int data, int K, vector<int> &ans)
{
    //B.C ->  if (root == NULL) return NULL;

    //B.C     if (root -> val == data)
                {
                    cout << ans[ans.size() - K - 1];
                    return;
                }

    ans.push_back (root -> val);
    solve (root -> left, data, K, ans);
    solve (root -> right, data, K, ans);
    ans.pop_back()
}
```


node = 7
k = 2
pop
pop
int index arr.size() - k;
cout << arr[arr.size() - k]
2 - 3 = -1
Size - K = 2 - 6
node = 5
K = 2
3 - 2 = 1


flag = false;
true
k--
k==0
cout<<
root
args
1
2
3
4
5
6
7
8
9
11
k = 6
2 nodes
n = 6
k = 2
flg = True
(I) root==NULL
return;
(II) if(root->val==data)
{ flag = true;
k--; return;
}
(III)
(IV) if(k==0)
cout <<root
(V)
left call
;
right call
if(flg==True)
k--;


bool found = false
root
1
2
3
4
5
6
7
8
X
X
X
X
X
X
count
K=0
if (K==0)
cout << root -> val
K = INT_MAX
if (K==0)
K--
K=1
K=2
node = 6
K = 3
ans = 3
if (K==0)
K = 3 2 1 0
if (K==0)
cout << root -> val
K--;
very good
through
code
KMvD
BD

```
void solve ( root , data , K )
{
    if (root == NULL)
        return;

    if (root -> val == data)
        found = true;
    {   if (K == 0)
        { cout << root -> val;
          K = INT_MAX; }

    K--;
    return
}
```



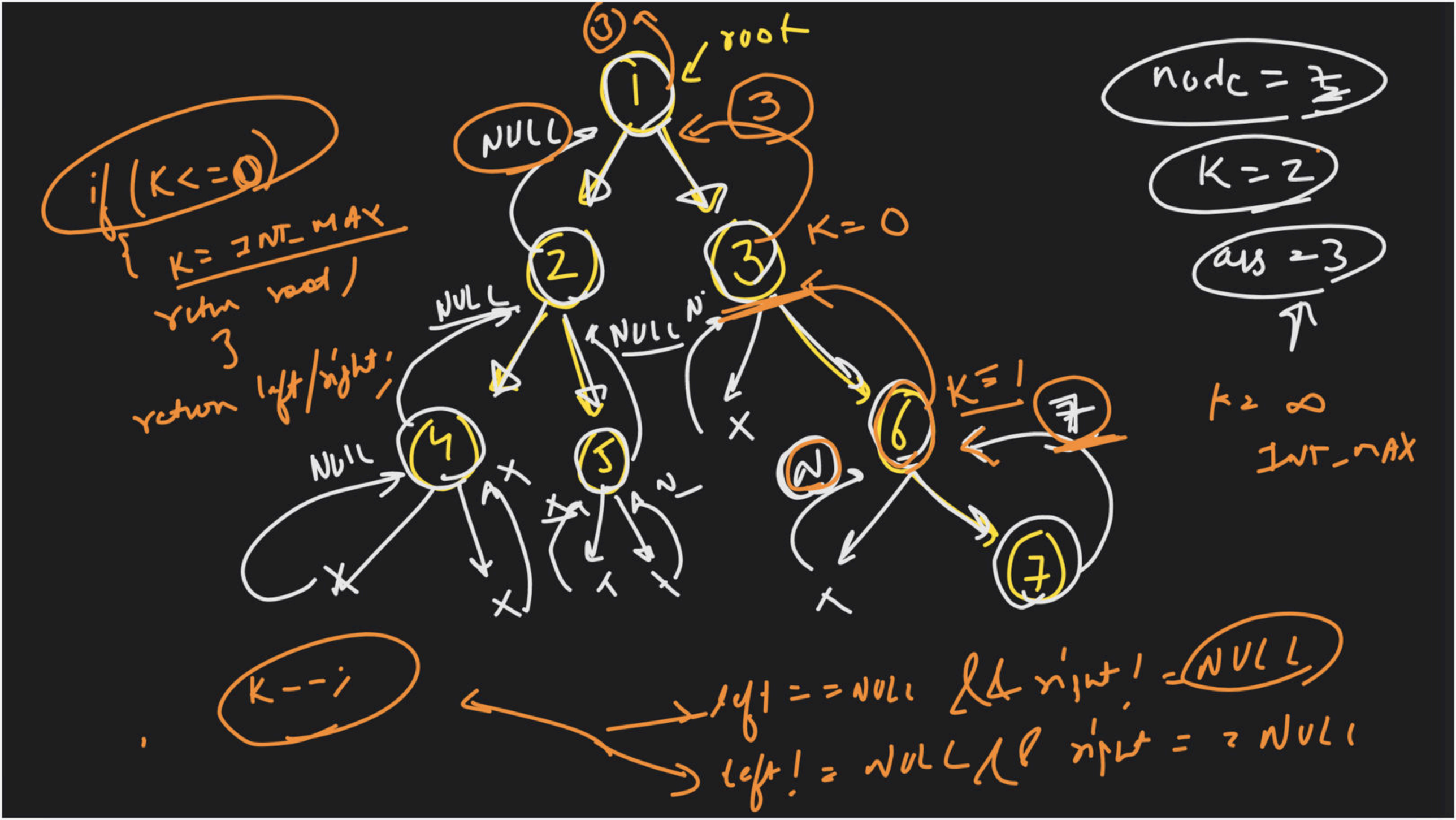
solve (root -> left , data , k);

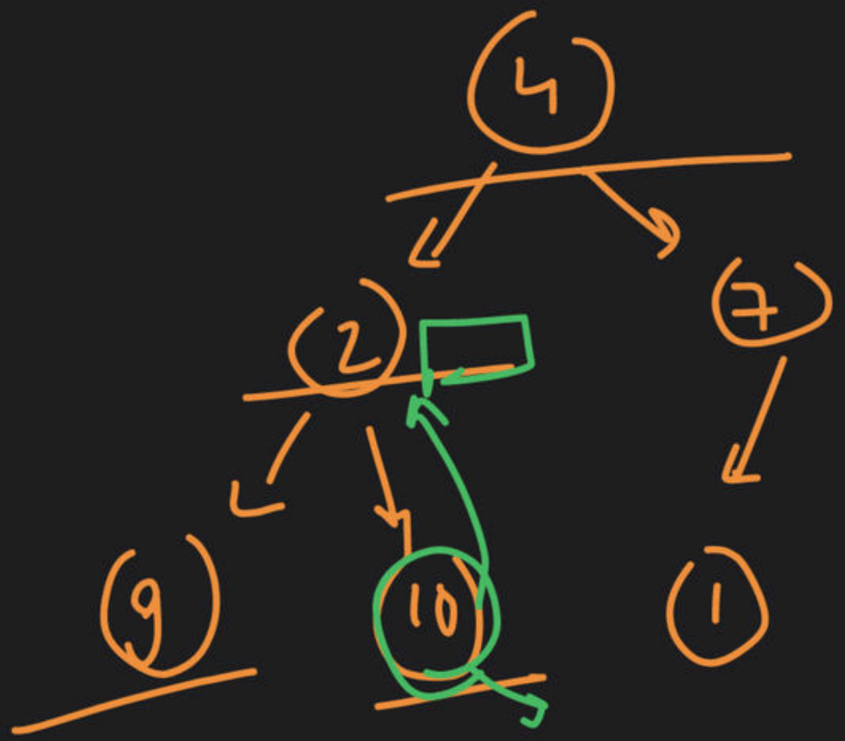
if (found = false)

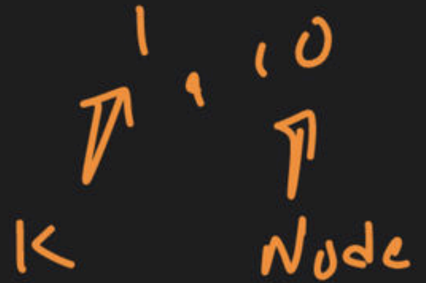
solve (root -> right , data , k);

gadbad

```
if (k == 0 && found == true)
{
    cout << root -> val;
}
else if (found == true)
    k--;
};
```

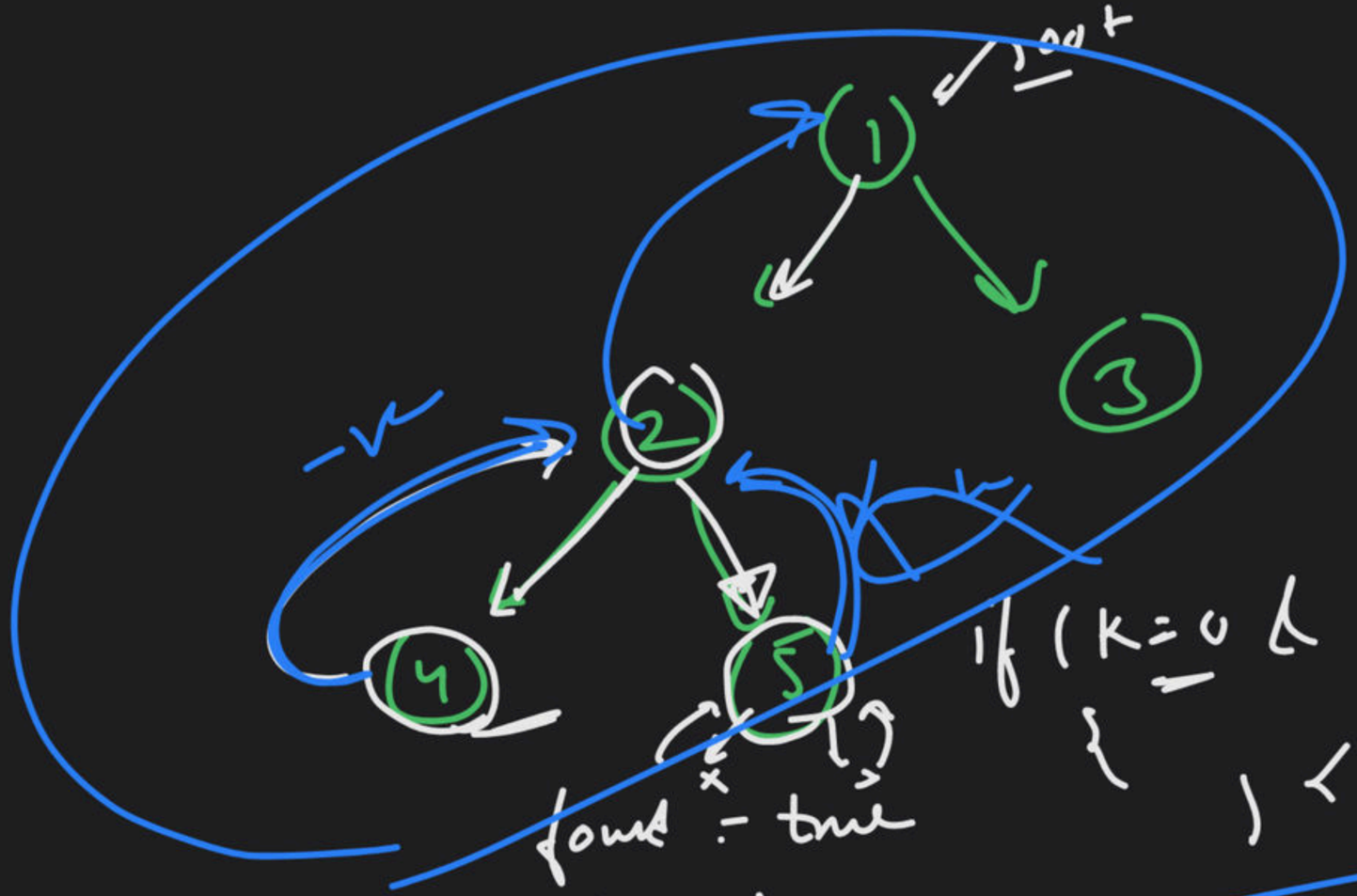

root
NULL
K = 0
NULL
NULL
K = 1
K--;
left == NULL && right != NULL
left != NULL && right == NULL
node = 7
K = 2
ans = 3
K = ∞
INT_MAX





K = INT_MAX

K = = INT_MAX;



DRY

KVN

node = 4

k = 2

if (k == 0 && f == true)

{

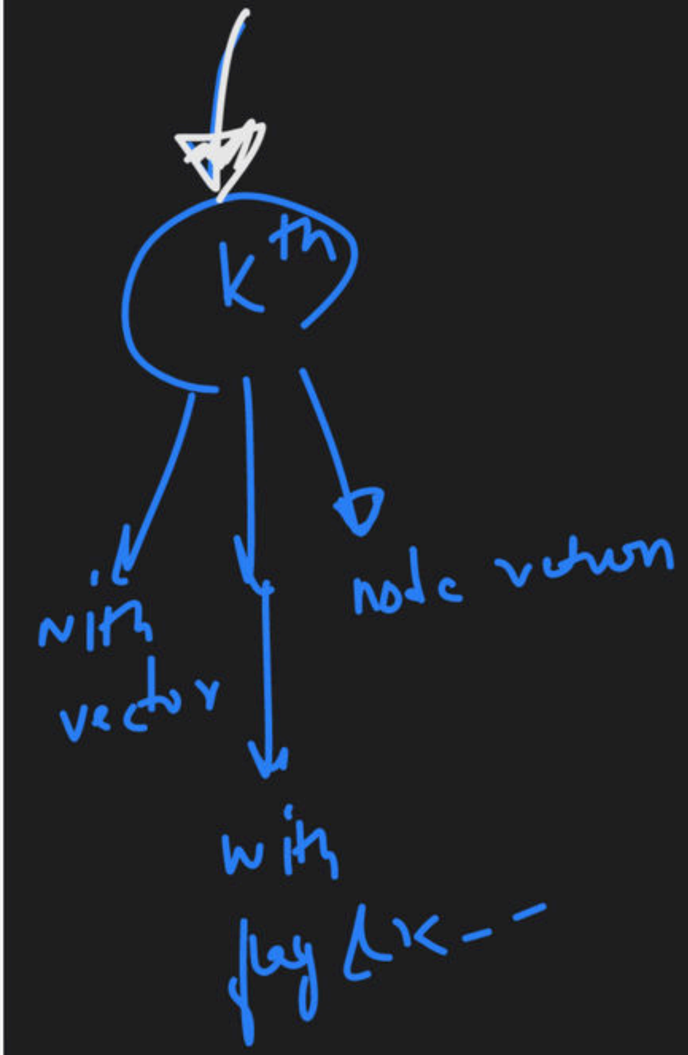
}

else if ( )

k = -1

next

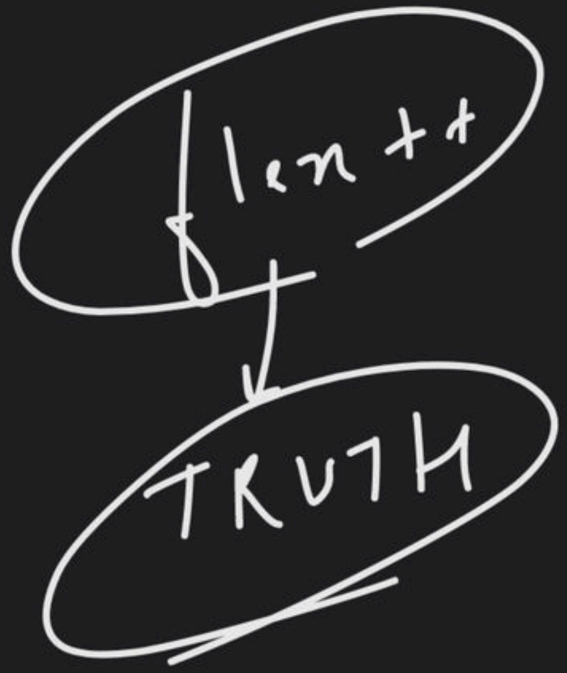
```
void solve (root, data, k, found)
{
    if (root == NULL)
        return;

    if (root -> data == data)
    {
        found = true;
        if (k == 0)
        {
            ans = root -> data;
            k = INT_MAX;
        }
        k--;
    }
}
```




```
solve (root->left, data, k, found)

if (found == false)
    solve (root->right, data, k, found)

if (k==0 && found == true)
{
    ans = root->data;
    k = INT_MAX;
}
else if (found = true)
{ k--; }
```

Paytm

Great UPttt



1.5 hr

2 hr 14 min