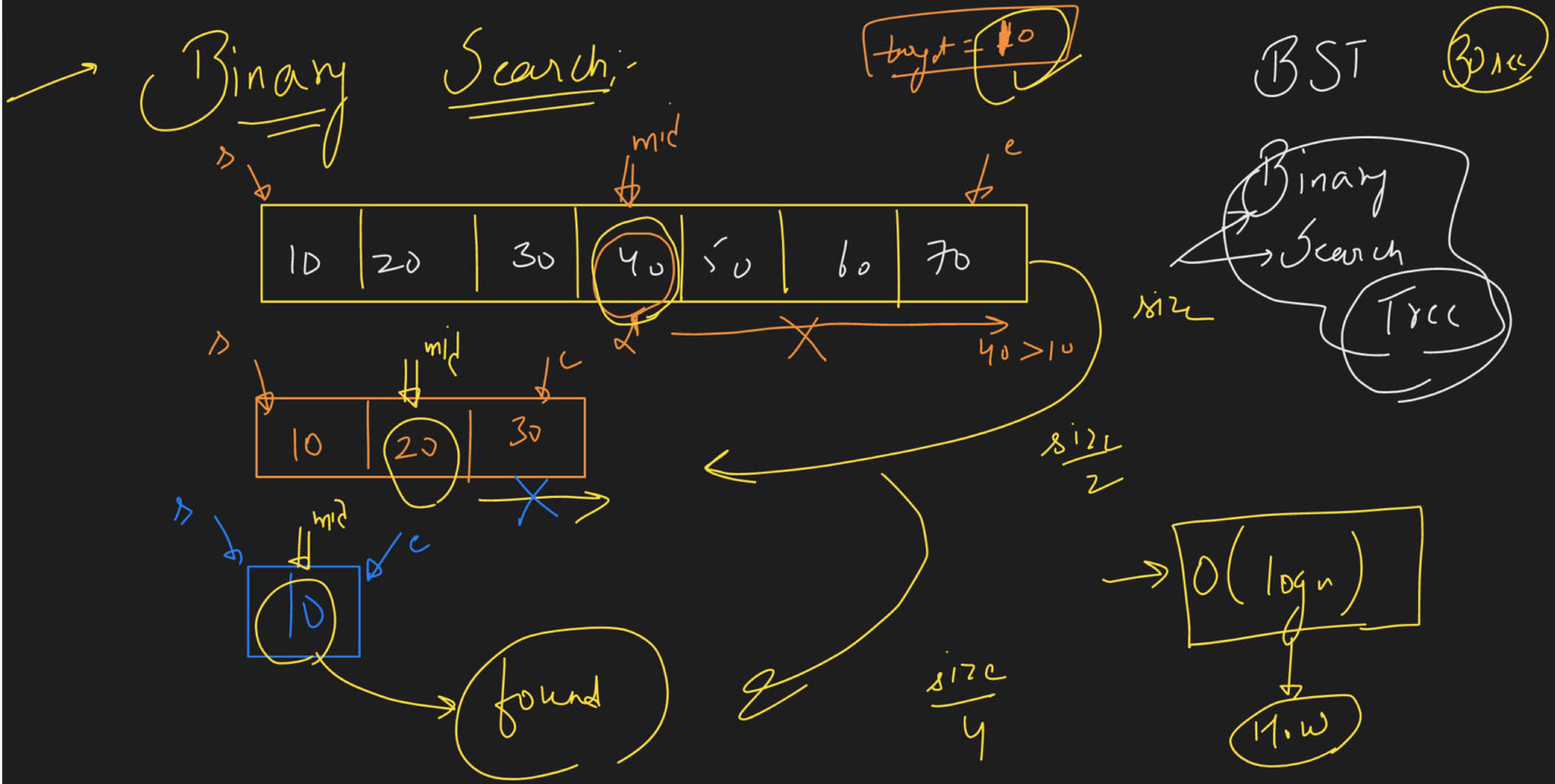
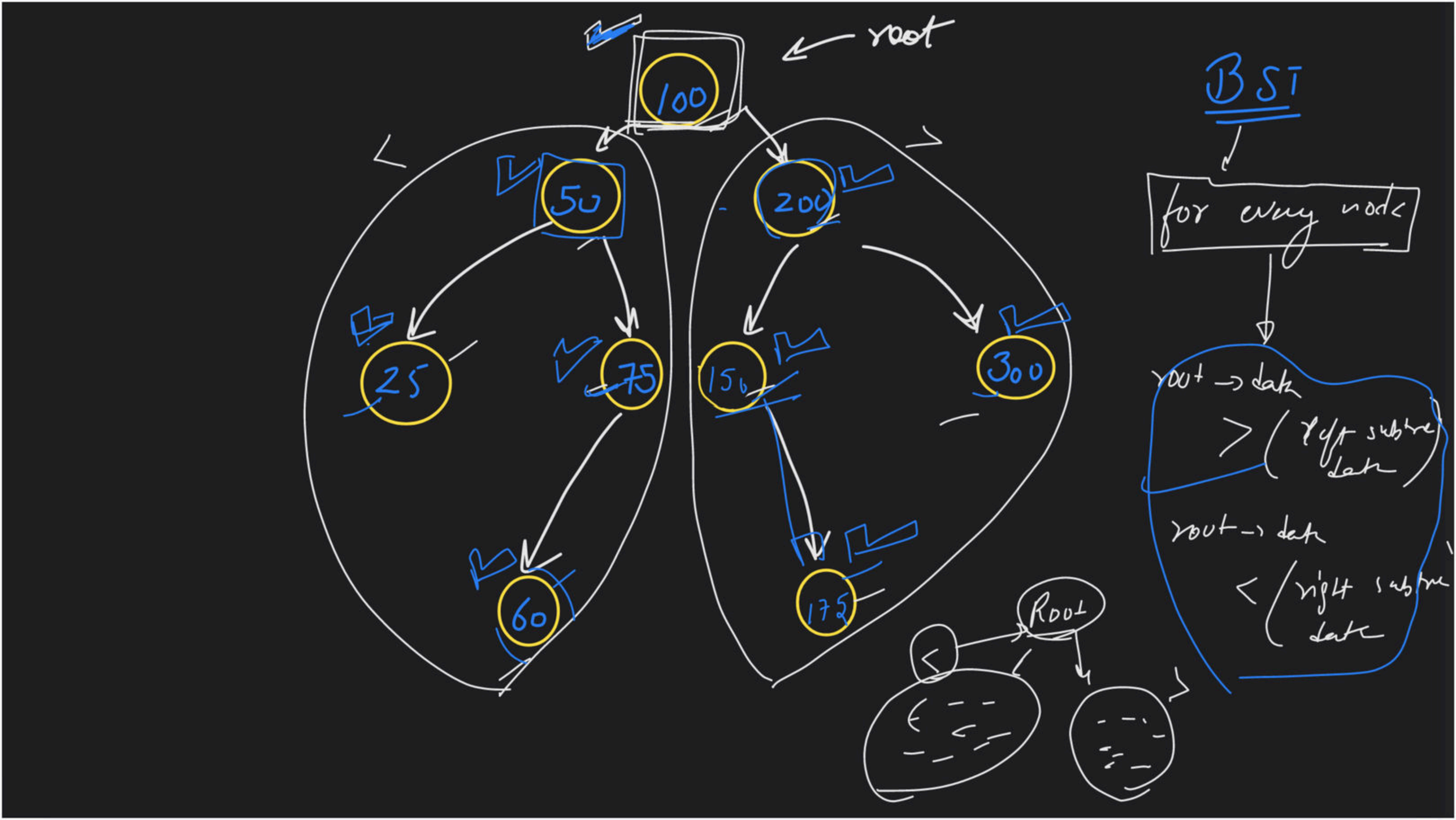
BST Class - 1

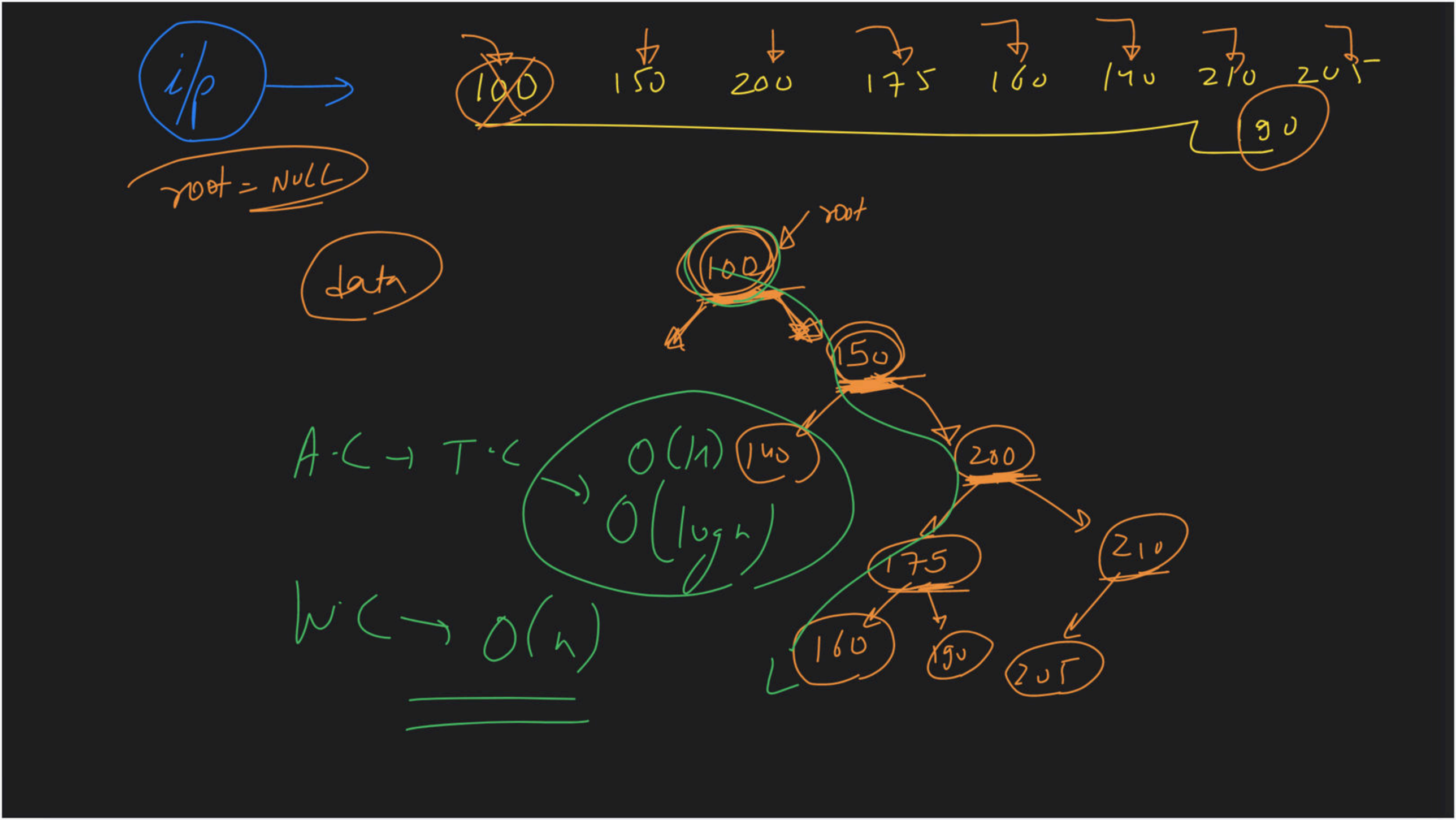
Special class

Love Babbar • May 14, 2023



Binary Search:-
target = 10
BST
30 sec
s
mid
e
10
20
30
40
50
60
70
40 > 10
size
Binary
Search
Tree
size/2
size/4
O(log n)
H.W
found


root
100
50
200
25
75
150
300
60
175
BST
for every node
root -> data
> (left subtree data)
root -> data
< (right subtree data
ROOT


i/p
100
150
200
175
160
140
210
205
90
root = NULL
data
root
100
150
140
200
175
210
160
190
205
A.C → T.C
O(1)
O(log n)
W.C → O(n)

i/p

10 20 5 11 17 2 4 8 6 25 15

Observation

Inorder of a BST is always Sorted

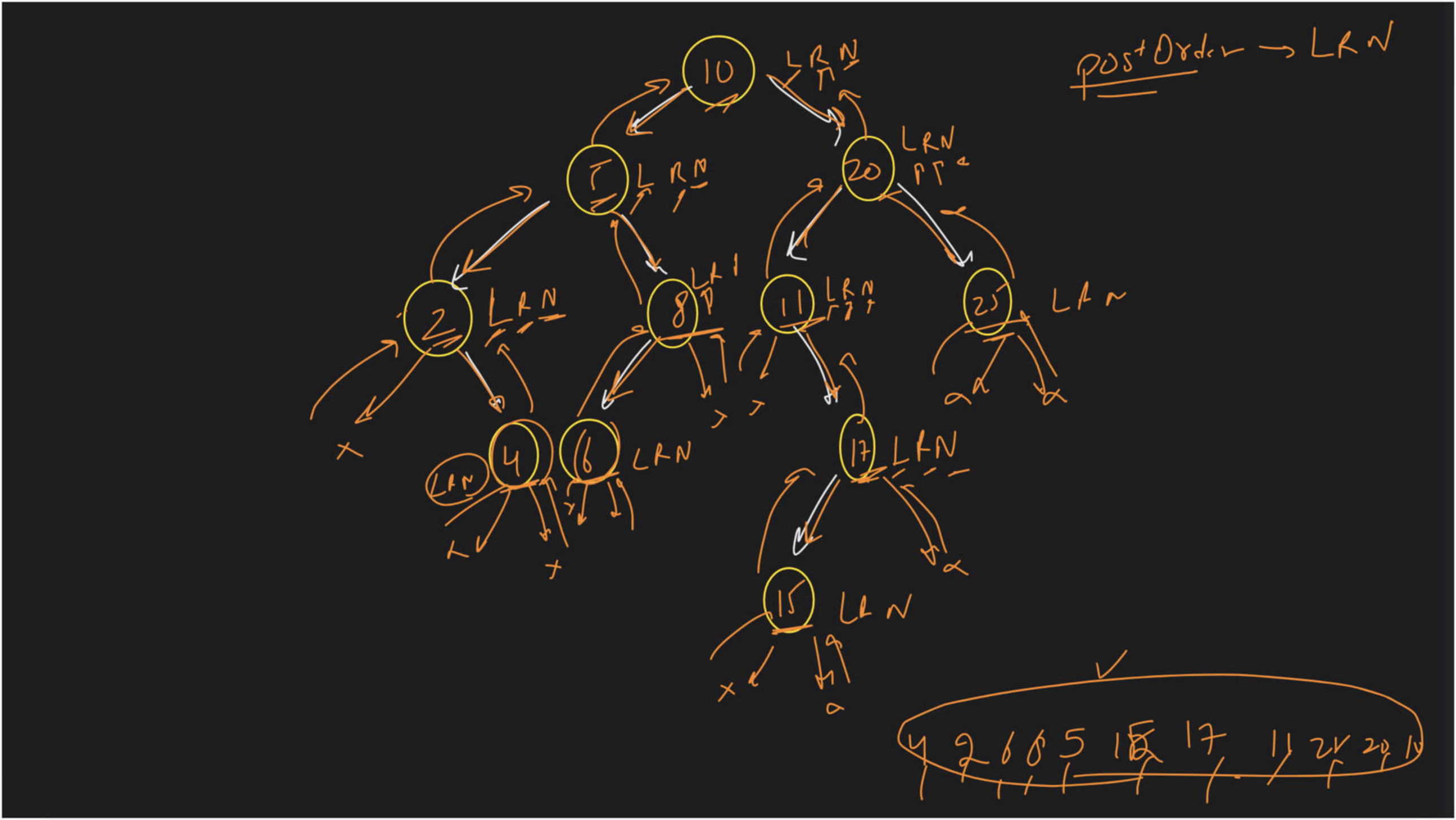
Creation of BST

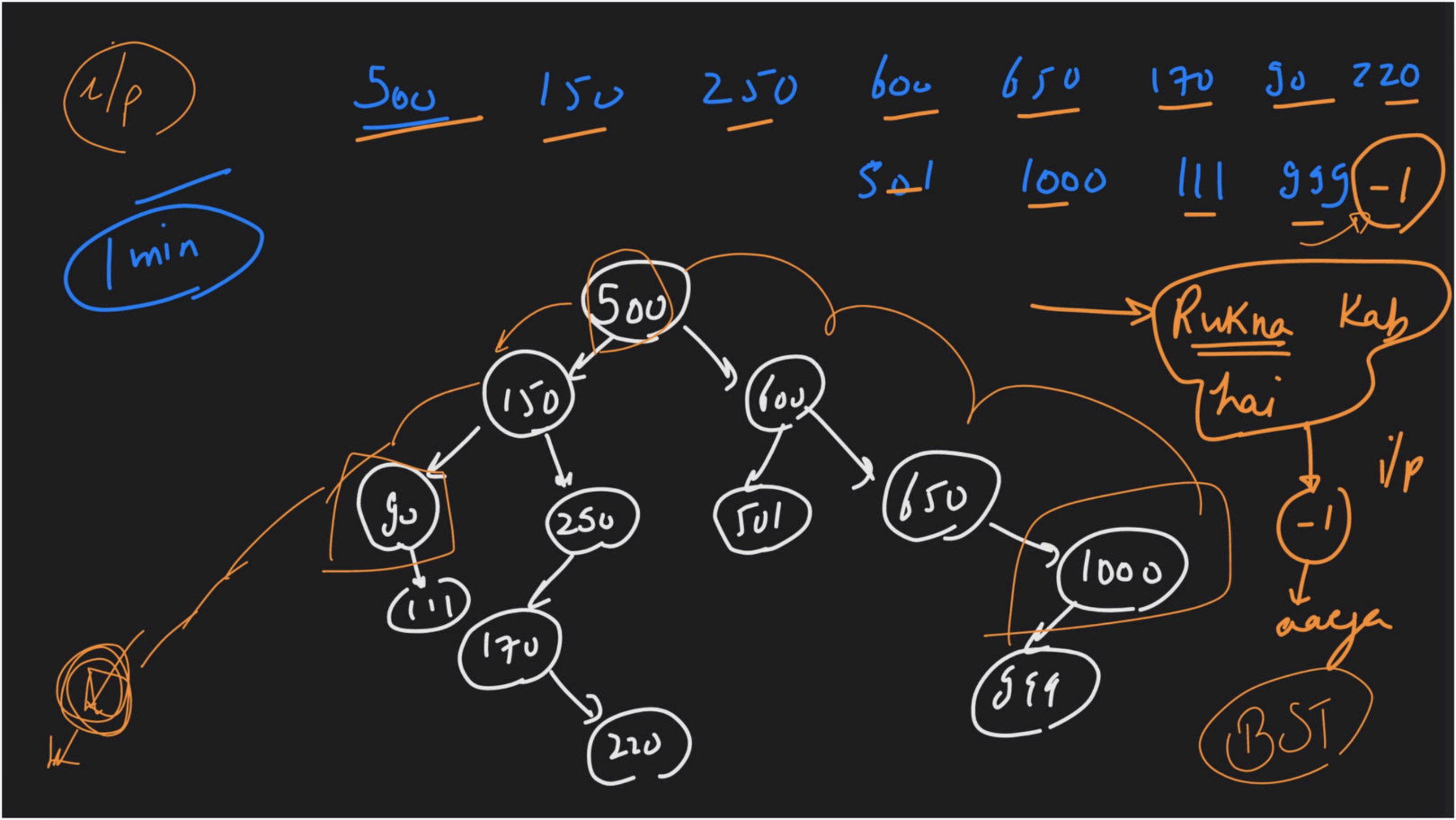
In → LNR

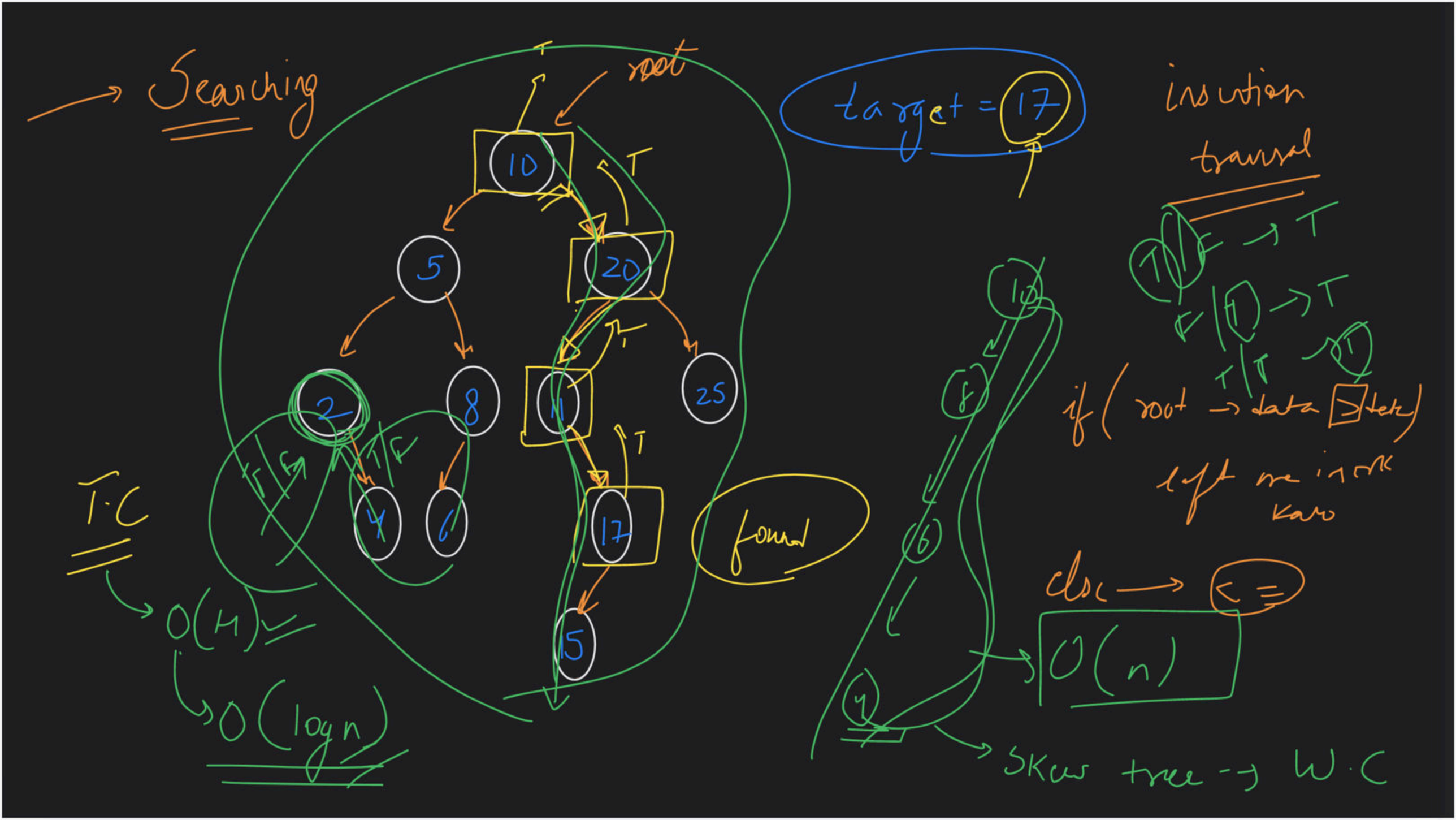
2 4 5 6 8 10 11 15 17 20 25

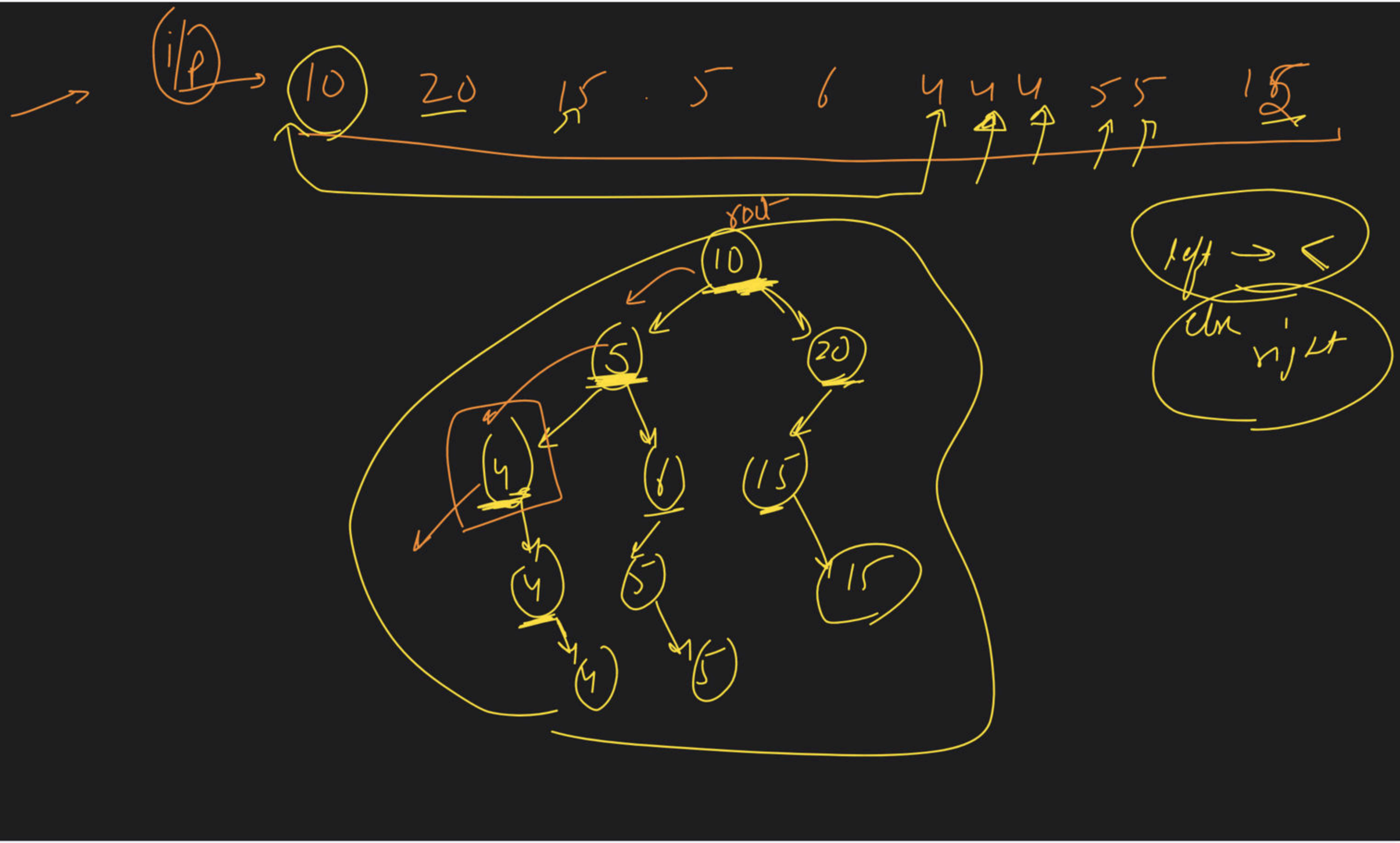
Pre Order →

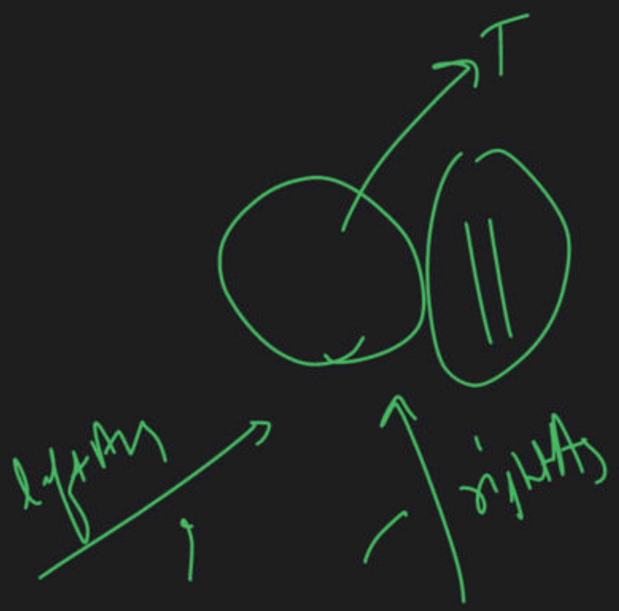
10 5 2 4 8 6 20 11 15 25

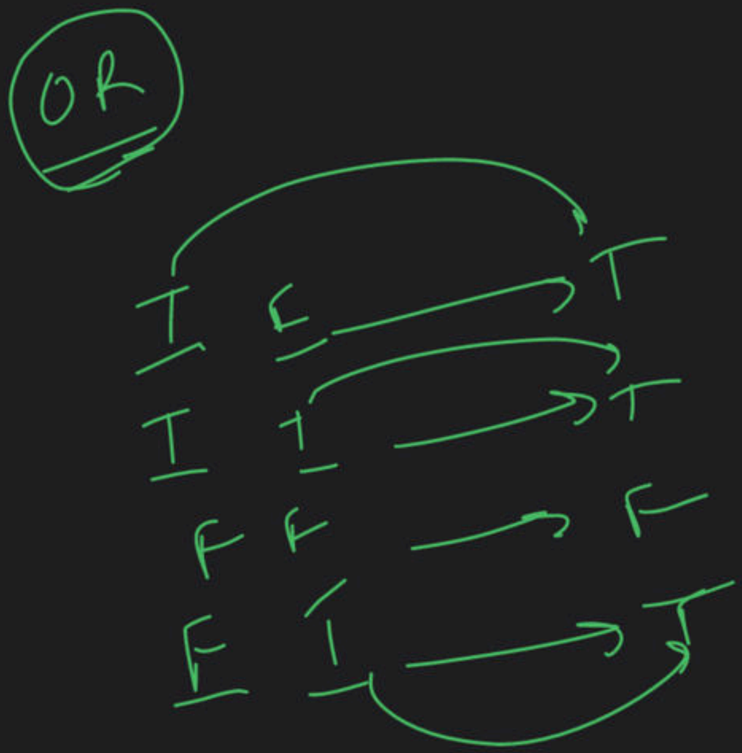

PostOrder → LRN
10
7
20
2
8
11
25
4
6
17
15
4 2 6 8 5 10 17 11 25 20 10

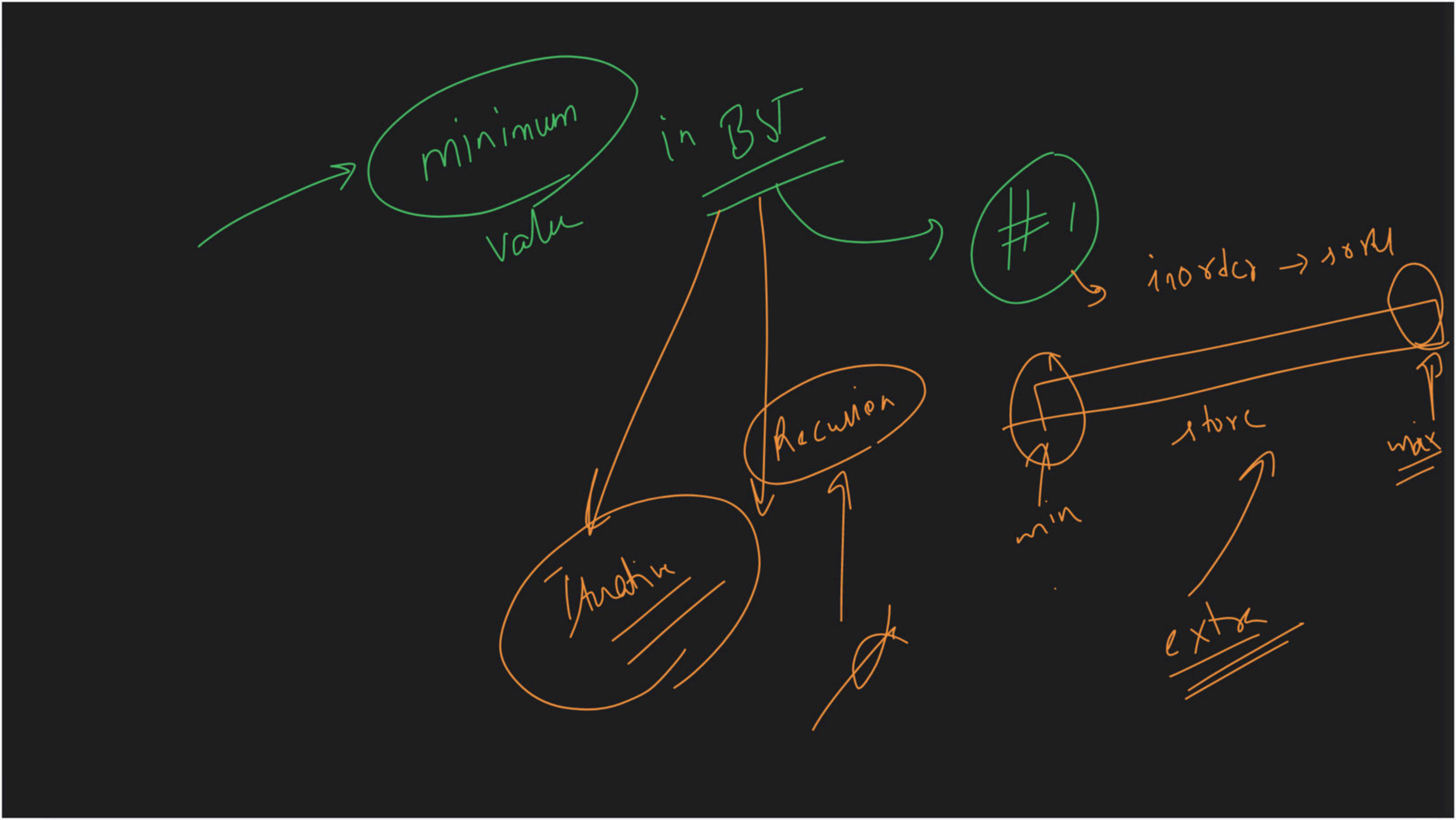

i/p
1 min
500 150 250 60 650 170 90 220
501 1000 111 999 -1
500
150
60
90
250
501
650
111
170
1000
220
999
Rukna Kab hai
i/p
-1
aaya
BST

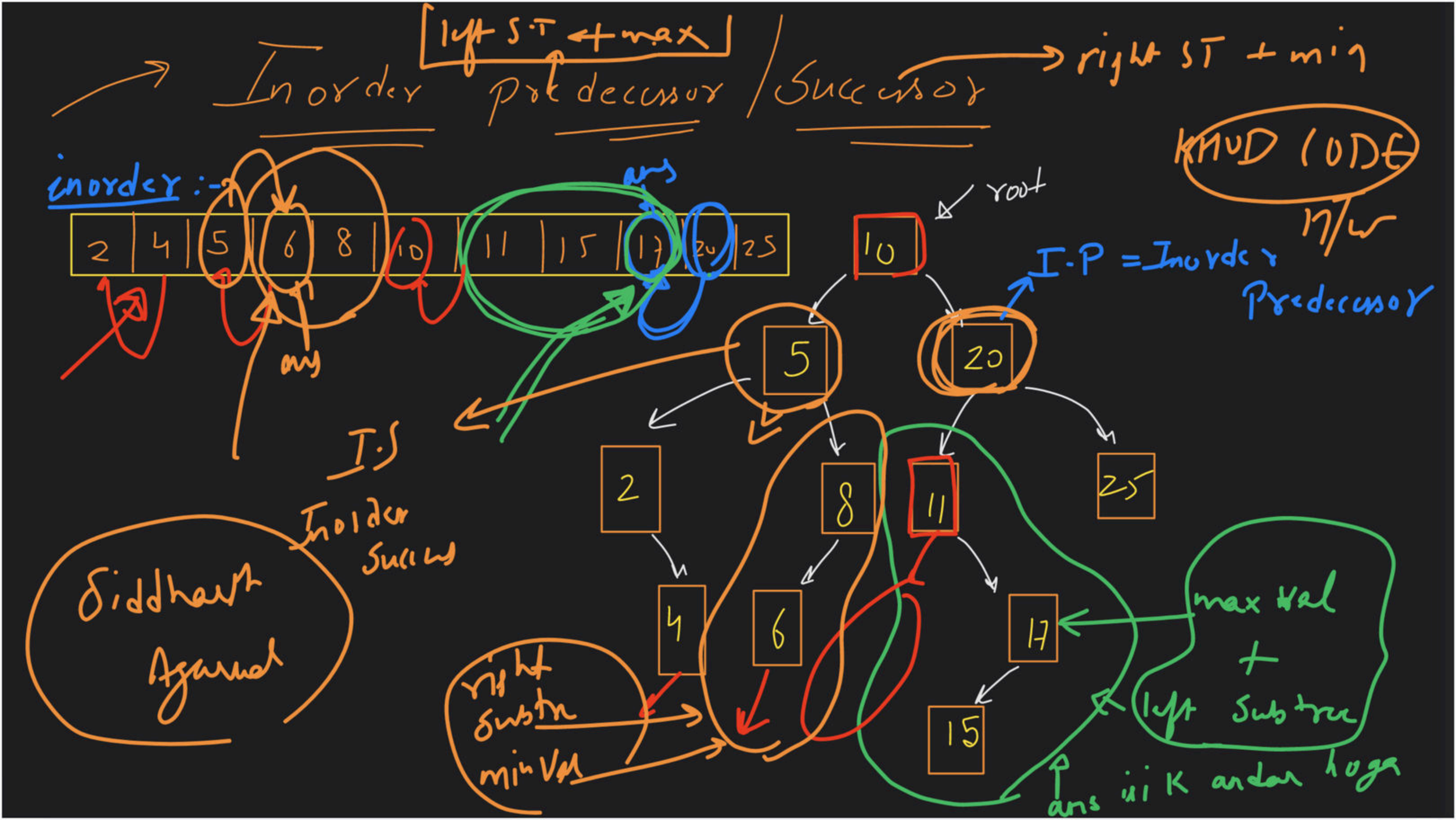

Searching
root
target = 17
insertion traversal
10
5
20
2
8
11
25
4
6
17
15
found
T
F
T.C
O(H)
O(log n)
if(root -> data > key)
left me insert karo
8
6
4
class
O(n)
Skew tree -> W.C


i/p
10
20
15
5
6
4 4 4
5 5
15
root
10
5
20
4
1
15
4
5
15
4
5
left -> <
right


T


OR
T
T
T
F
F
F
T


minimum
value
in BST
#1
inorder → root
Recursion
Iterative
root
min
max
extra


left ST → max
Inorder Predecessor / Successor
right ST → min
KHUD CODE
inorder :-
2 4 5 6 8 10 11 15 17 20 25
ans
ans
root
10
5
20
I-P = Inorder Predecessor
I.S
Inorder Successor
2
8
11
25
4
6
17
15
Siddharth Agarwal
right Subtree minVal
max Val + left Subtree
ans ki K andar hoga

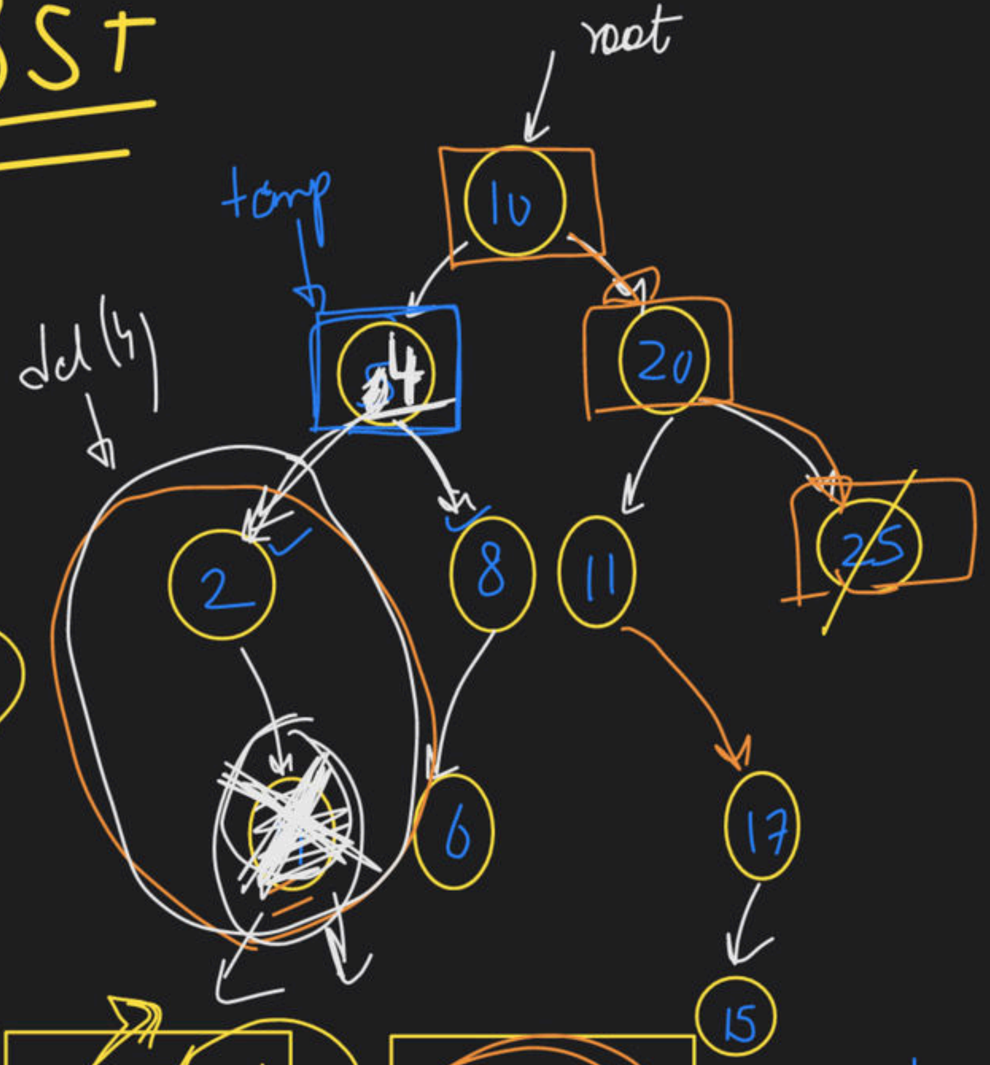
Deletion in BST

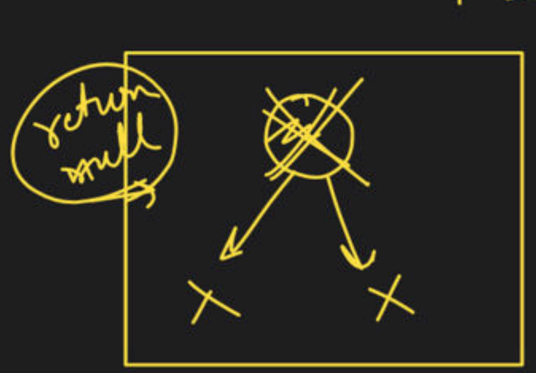
target = 25

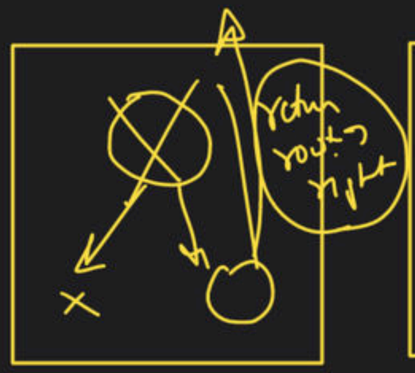
(1) 25 talk pukaro

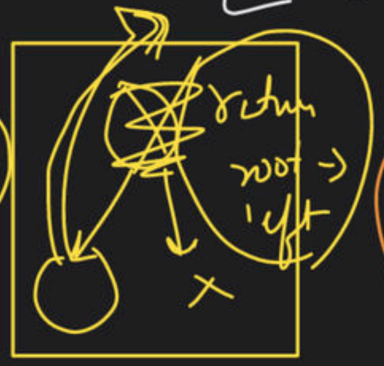
(2) 25 ko delete karna h

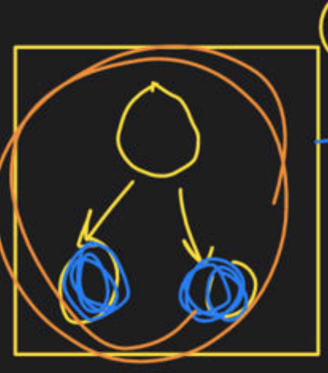
4 cases









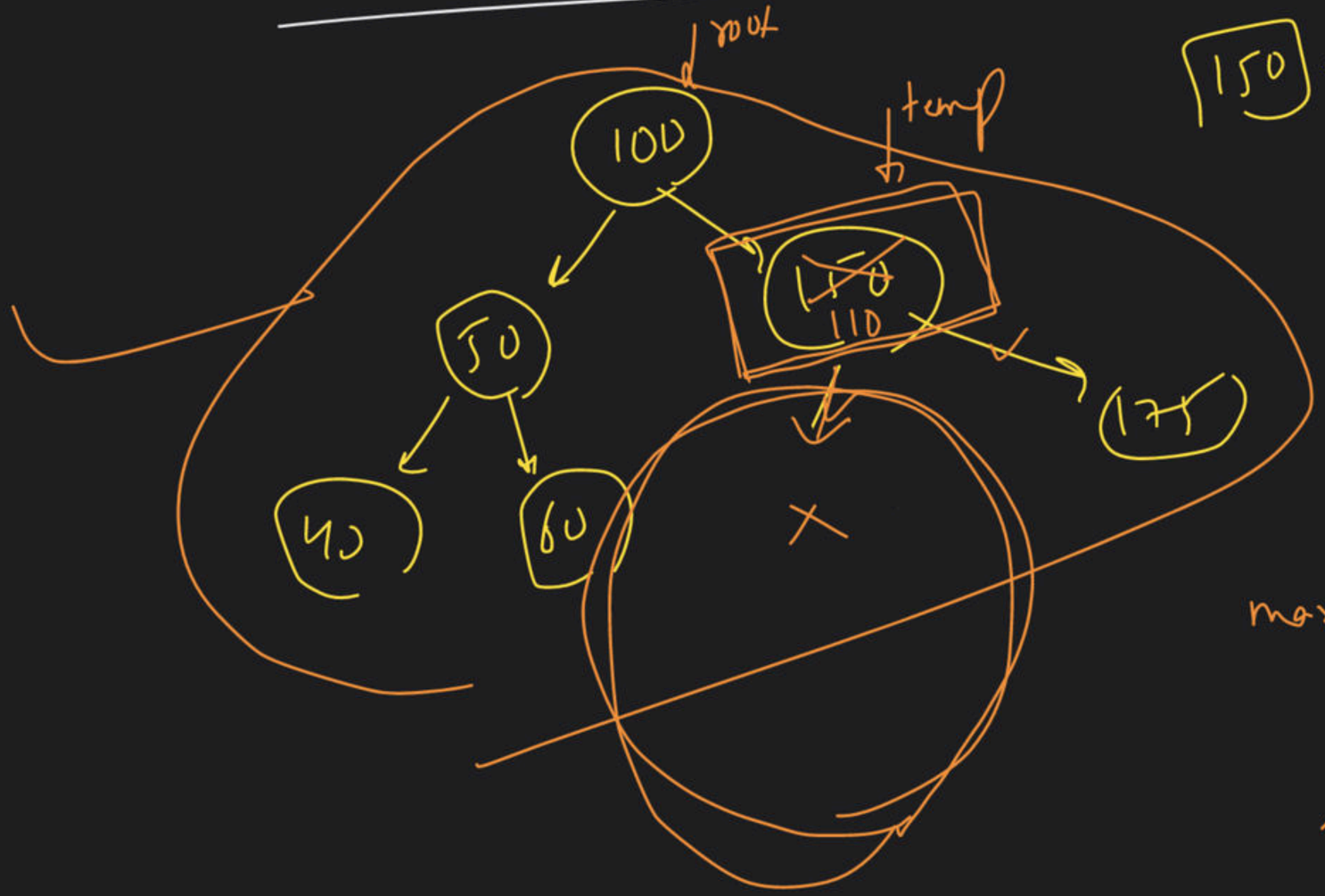


root -> data

T.P

T.S

100 50 150 40 60 175 110 -|

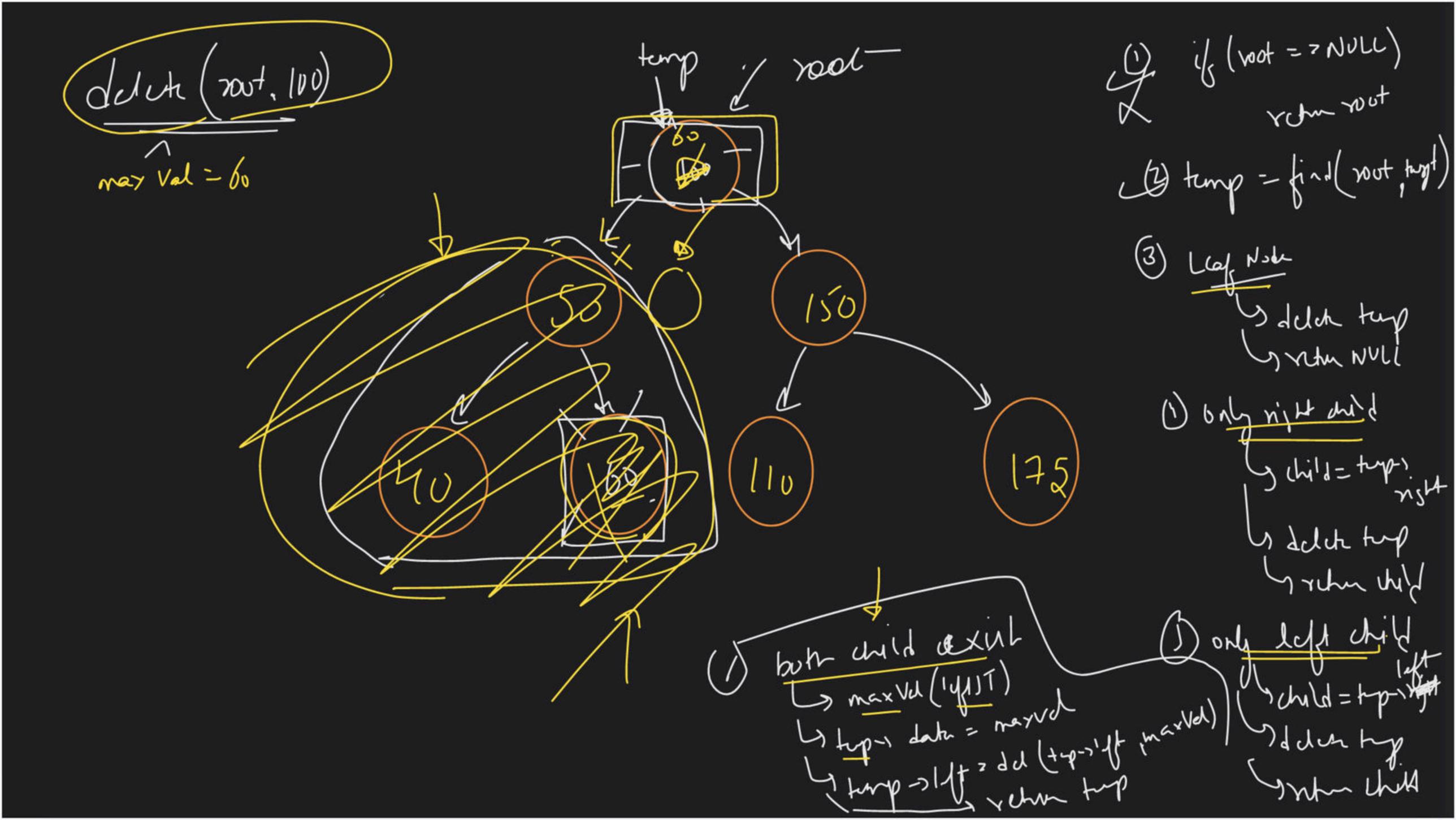


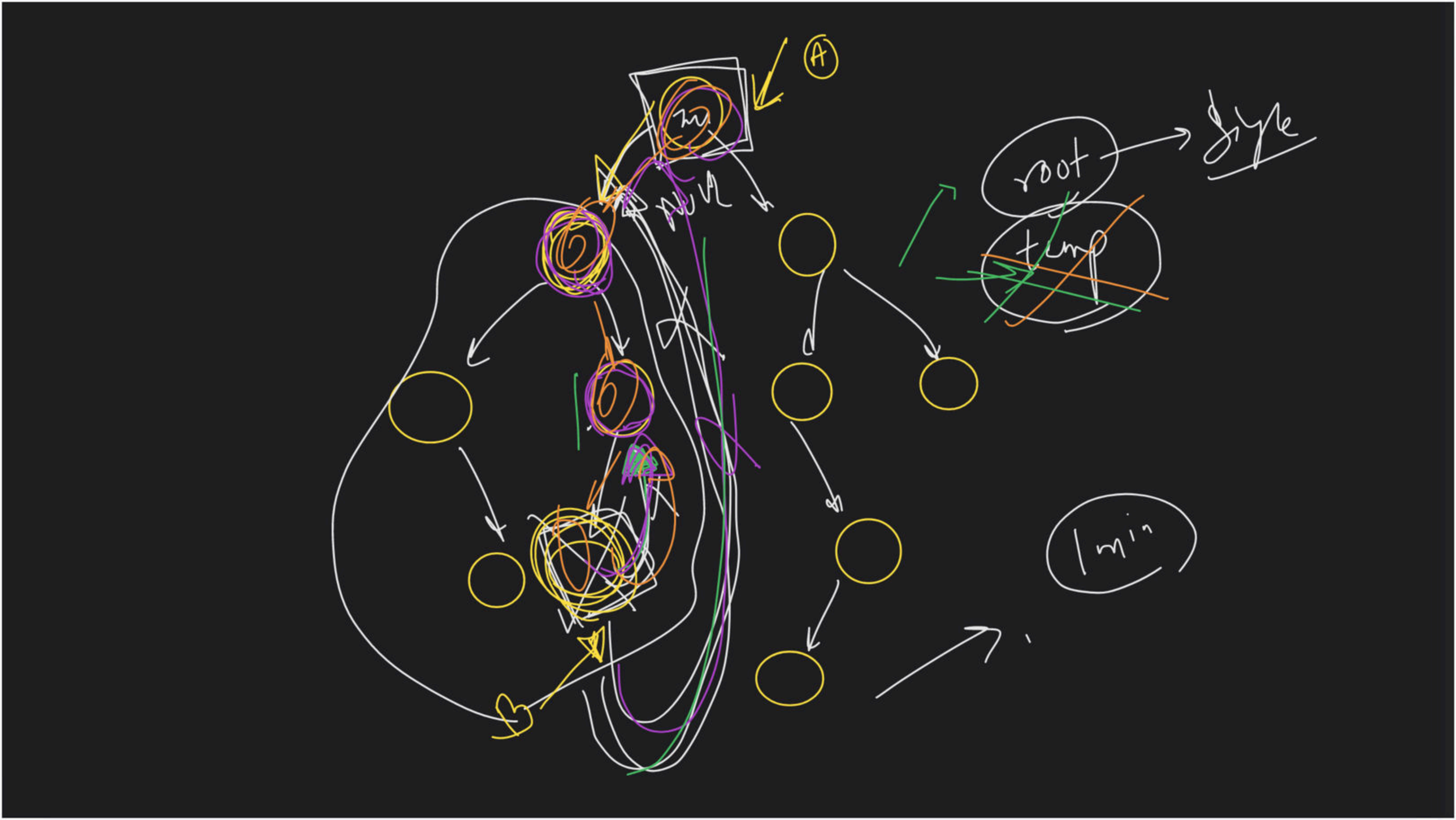
150 ← target

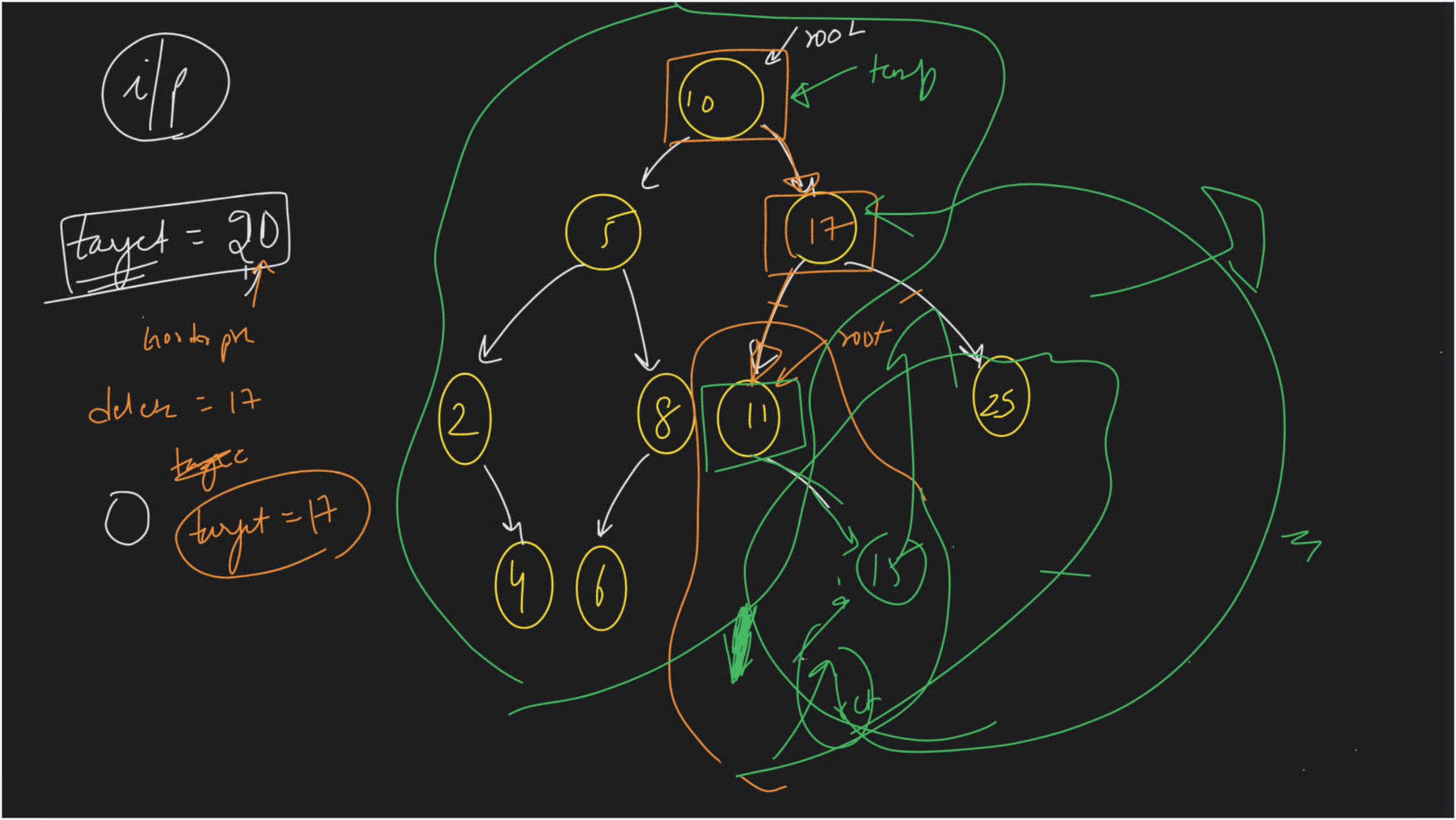
(A)
maxVal = 110

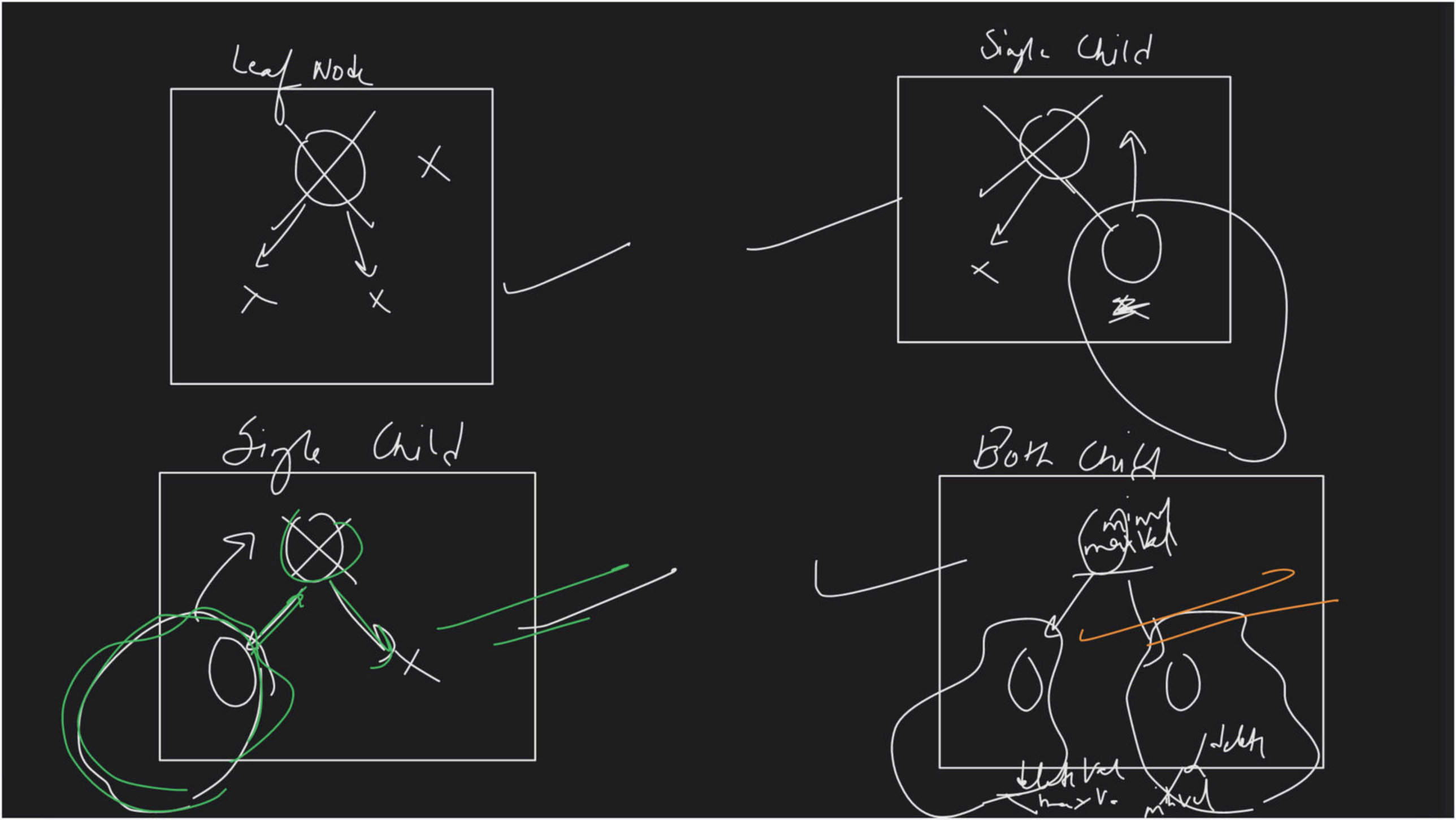
(B)
~~150~~ → 110

(C)
delete(110, leftSubtree)


delete (root, 100)
maxVal = 60
temp
root
60
50
150
40
60
110
175
① if (root == NULL)
return root
② temp = find(root, target)
③ Leaf Node
delete temp
return NULL
① only right child
child = temp->right
delete temp
return child
① only left child
child = temp->left
delete temp
return child
① both child exist
maxVal (leftST)
temp->data = maxVal
temp->left = del (temp->left, maxVal)
return temp


A
root
temp
1 min


i/p
target = 20
inorder
delete = 17
target
target = 17
root
temp
10
5
17
2
8
11
25
4
6
15


Leaf Node
Single Child
Single Child
Both Child