Plan for today

Saturday, 14 August 2021 10:14 AM

Morning

✓ 1. construct Binary Tree using Inorder and level order

✓ 2. Construct BST using Inorder }

✓ 3. Construct BST using Pre Order

✓ 4. construct BST using Postorder

Evening (from 7:00 PM)

→ 1. Cameras in Binary Tree

→ 2. House Robber in B.Tree

→ 3. longest zigzag path

→ 4. Validate BST

→ 5. Recover BST

skip → construct BST using level order

Construct Binary Tree Using Inorder and Levelorder

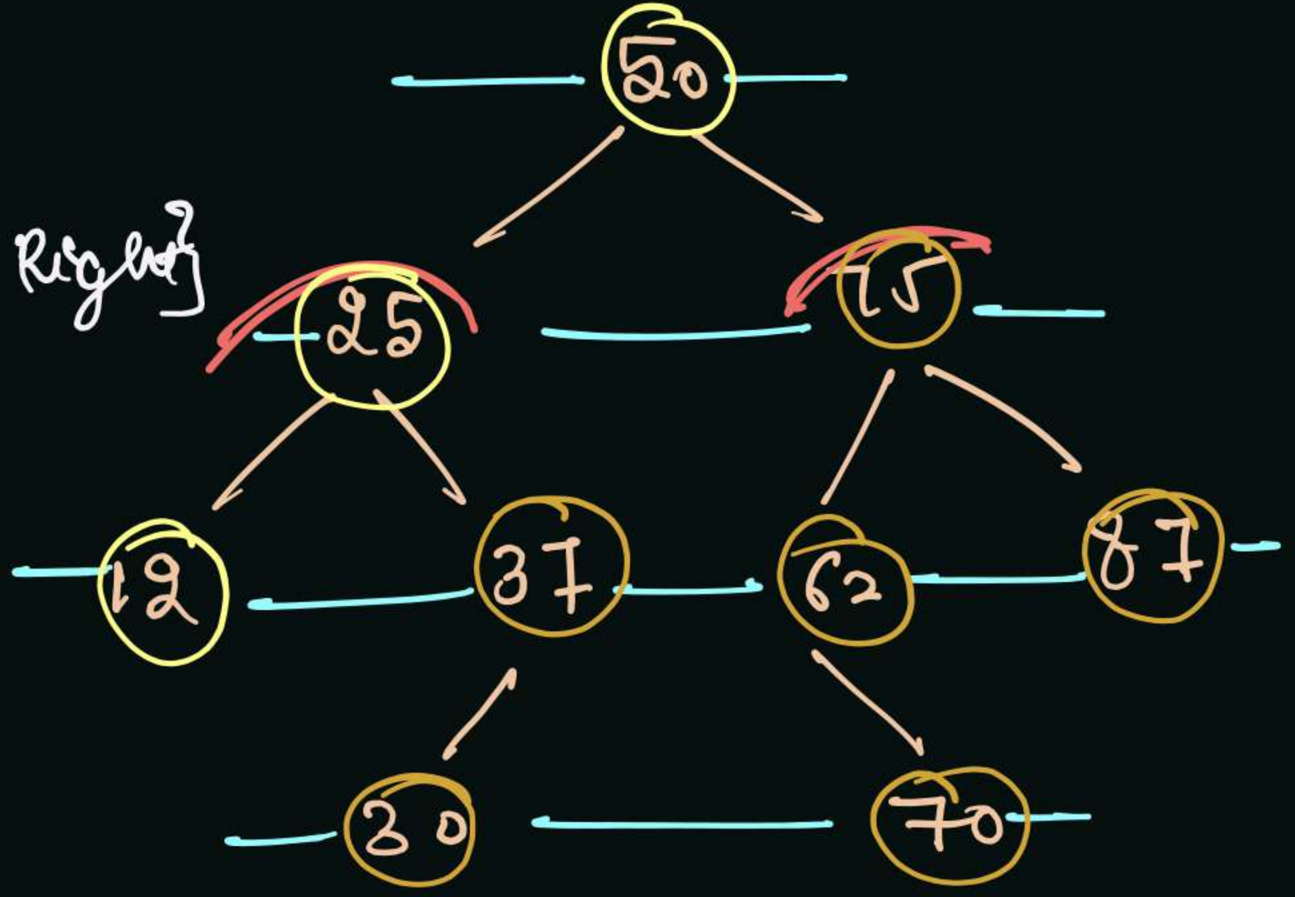
Saturday, 14 August 2021 10:12 AM

Binary Tree

Inorder → 12 25 30 37 50 62 70 75 87 {left Root Right}

level Order → 50 25 75 12 37 62 87 30 70

{level wise}



level=length=0

level.length=0

Hashset

★ In → 12

level → 12

In → 30

level → 30

In →

level = 0

In → 30 37

level → 37 30

level.length=0

In → 70

level → 70

In → 62 70

level → 62 70

In → 87

level → 87

Hashset

12

Hashset

12

25

30

37

left

In → 12 25 30 37

level → 25 12 37 30

right

In → 62 70 75 87

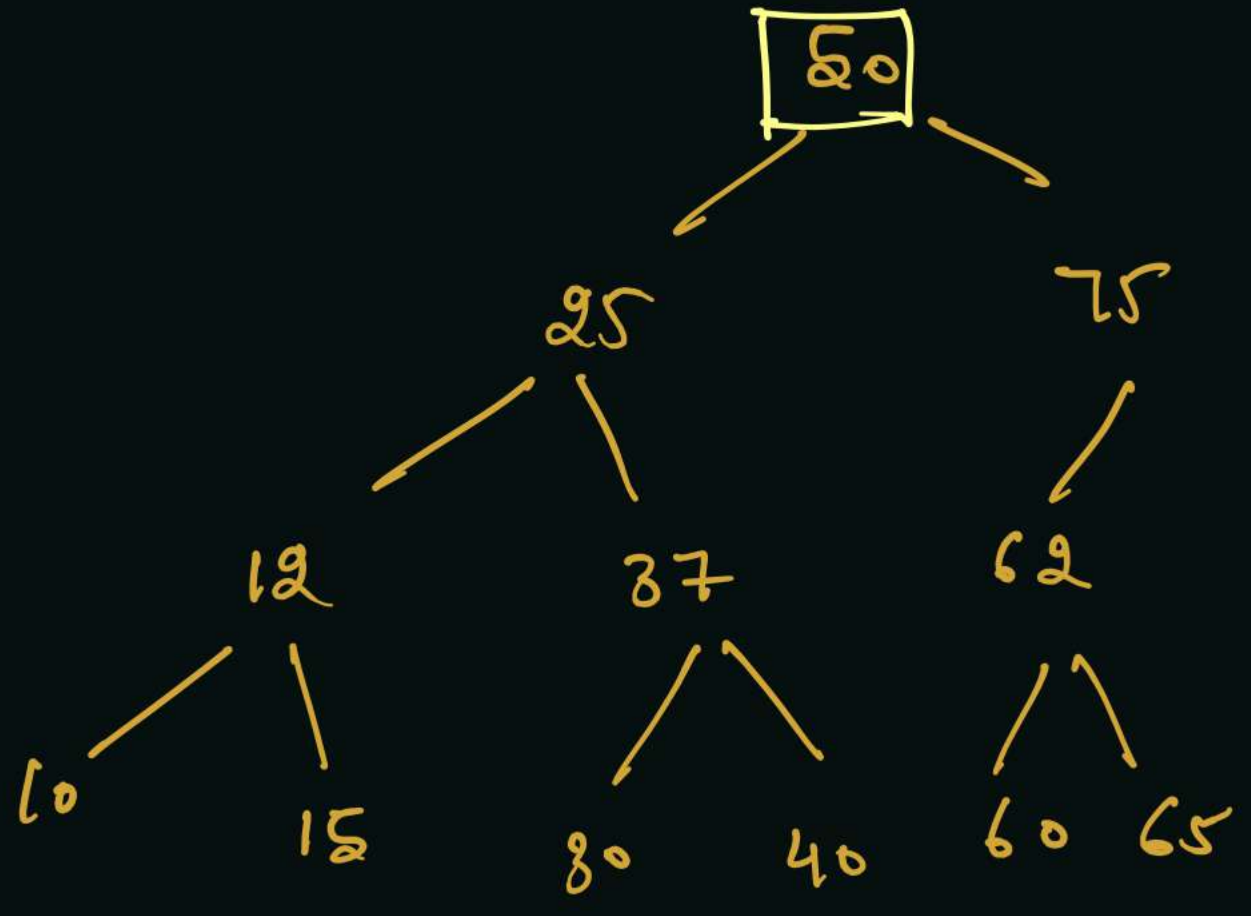
level → 75 62 87 70

In → 12 25 30 37 50 62 70 75 87

level → 50 25 75 12 37 62 87 30 70

Inorder →

0	1	2	3	4	5	6	7	8	9	10	11
10	12	15	25	30	37	40	50	60	62	65	75
↑ Instart Inclue							↑ Indx Exclude				↑ InEnd



Indx = 7 [Index of Root]

Element Count [left Element]

Element Count = Indx - Instarting] difference // no. of Elements b/w Indx and In starting Index

i ↓				i ↓						j j ↓	
0	1	2	3	4	5	6	7	8	9	10	11
a	b	c	d	e	f	g	h	i	j	k	l

Indx → i = 4] No. of Elements b/w i & j → i → Include, j → Exclude. j - i Case - I

Indx → j = 10

i → Exclude, j → Exclude j-i-1 Case - II

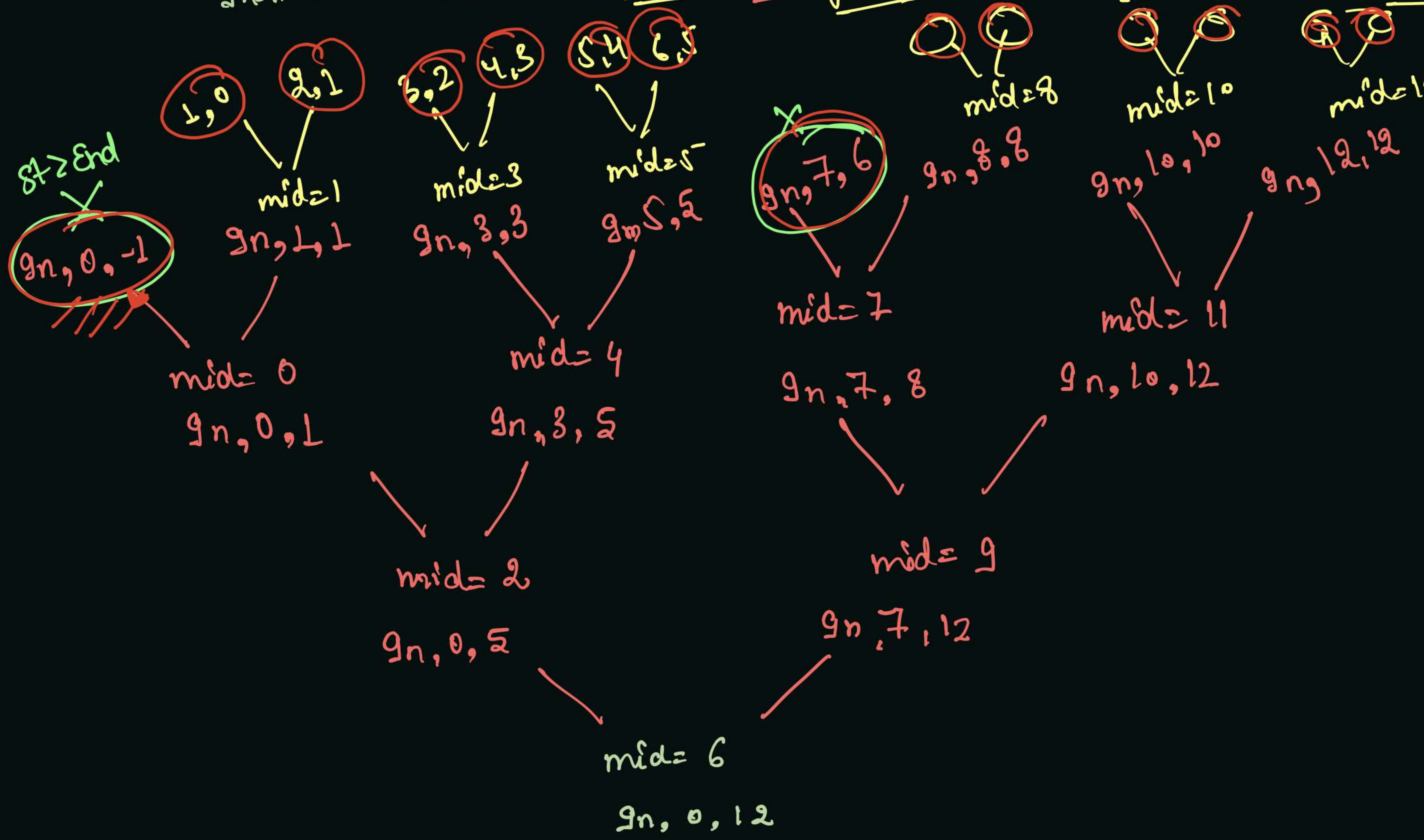
i → Incl. j → Incl. j-i+1 Case-III

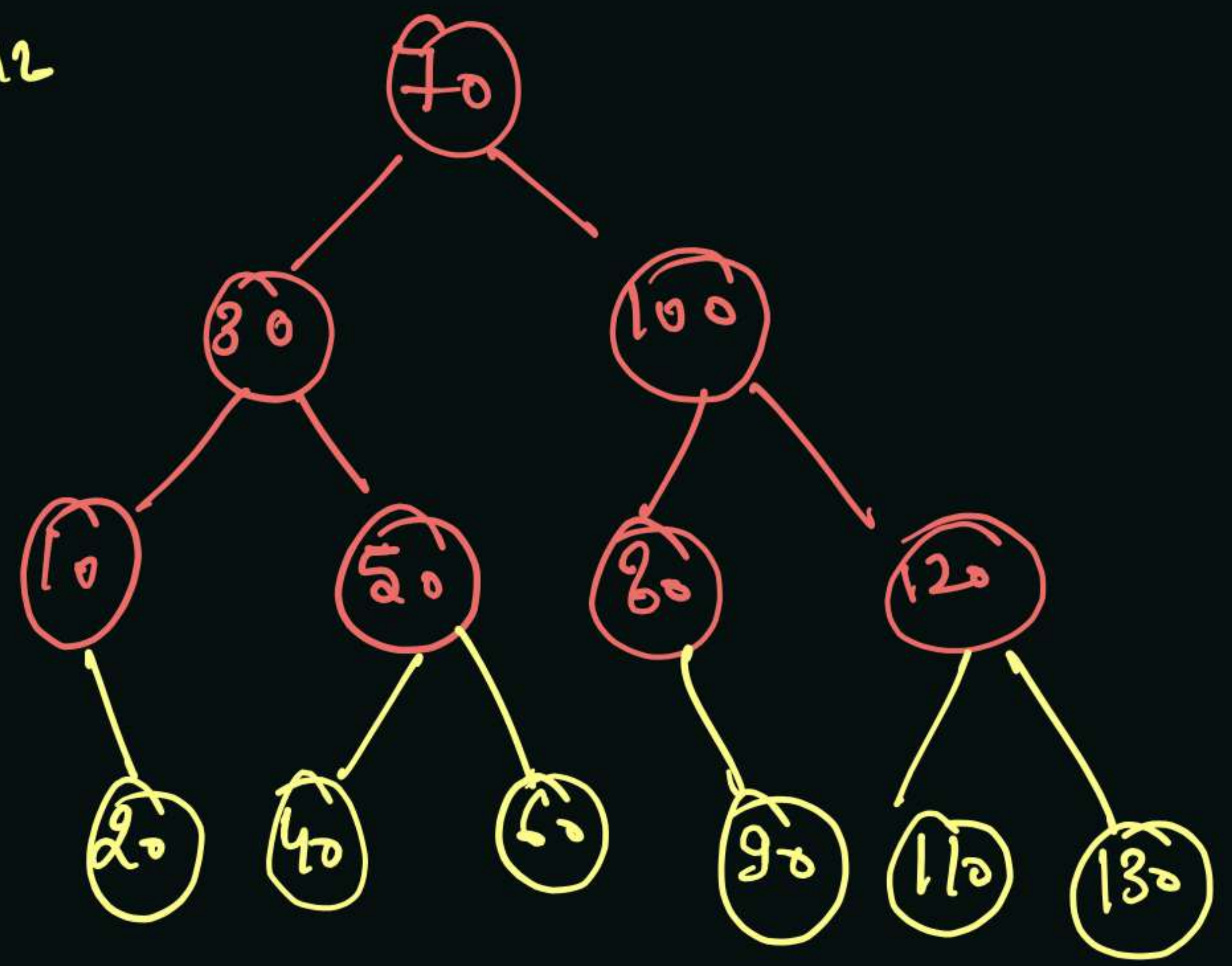
Construct BST Using Inorder

Saturday, 14 August 2021 10:13 AM

Poperty of BST → Inorder is always sorted.

Inorder →	10	20	30	40	50	60	70	80	90	100	110	120	130
Indx →	0	1	2	3	4	5	6	7	8	9	10	11	12





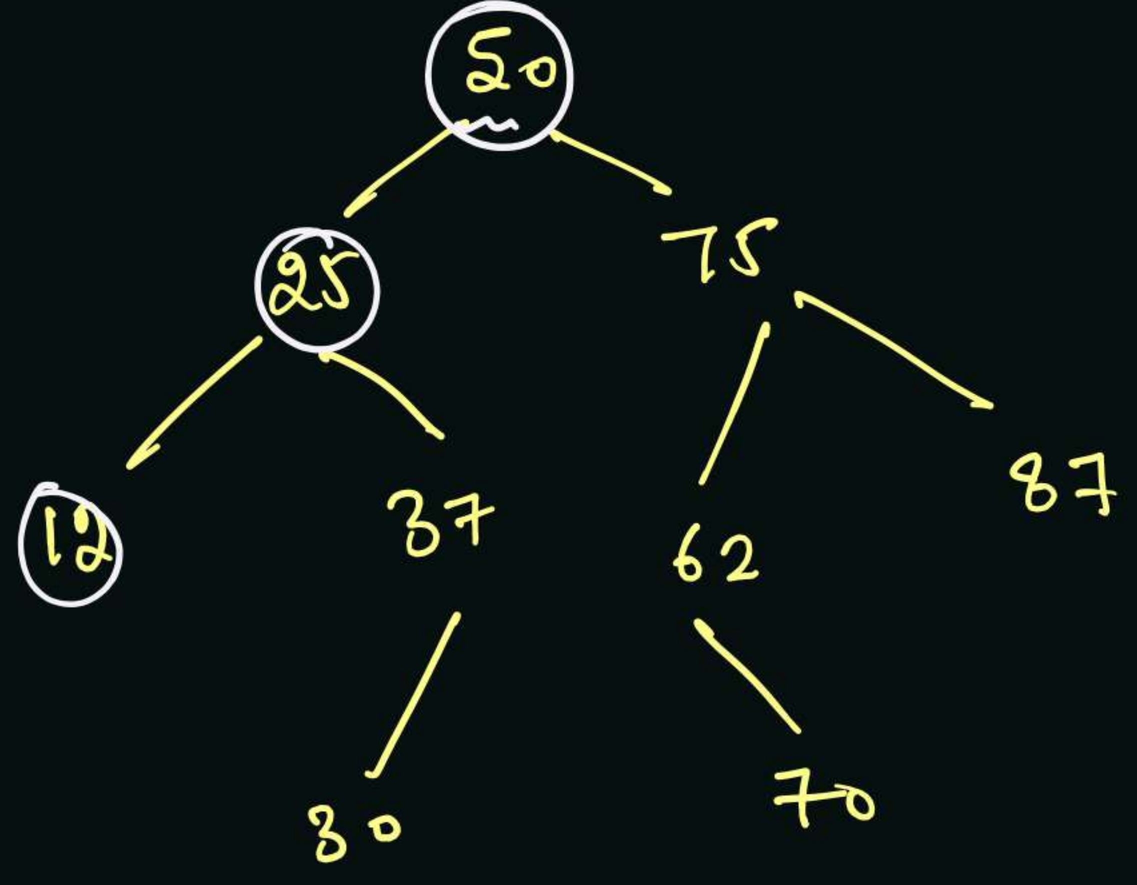
Construct BST Using Preorder

Saturday, 14 August 2021 10:13 AM

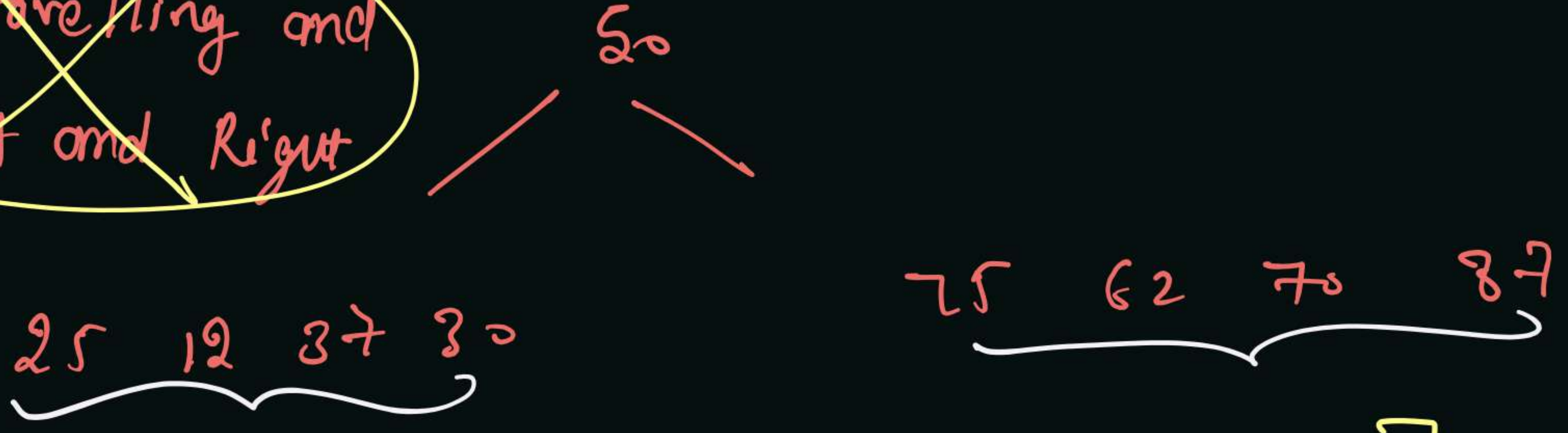
BST

PreOrder → 50, 25, 12, 37, 30, 75, 62, 70, 87

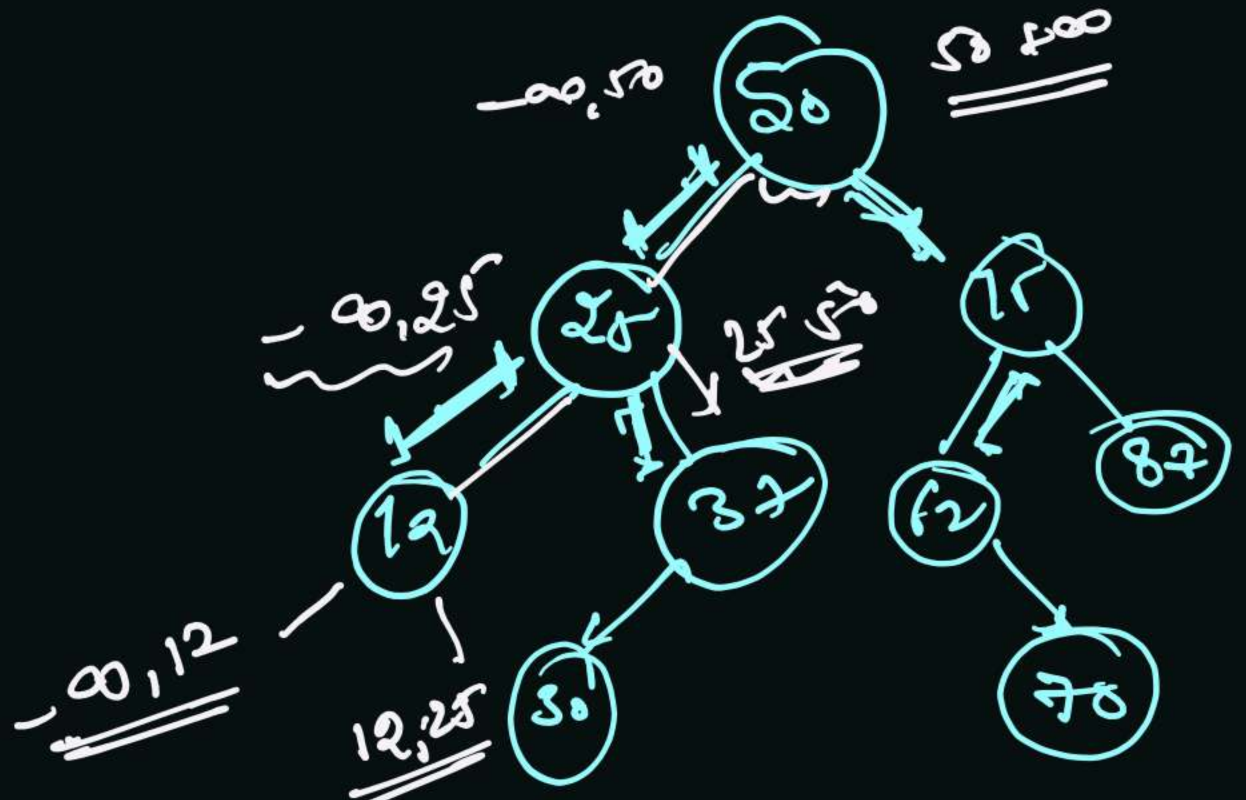
Root left Right



Method 1 → ~~# travelling and left and Right~~



Method 2 → Sort PreOrder and solve for InOrder] X



Methods → solve for first Element of preOrder, and figure out correct position for next element.

	0	1	2	3	4	5	6	7	8
PreOrder →	50,	25,	12,	37,	30,	75,	62,	70,	87

Indx

75, ∞,

~~70, 75~~

~~62, 70~~

(8) 87 ~~62, 75, R~~

(6) 62 ~~50, 75, R~~

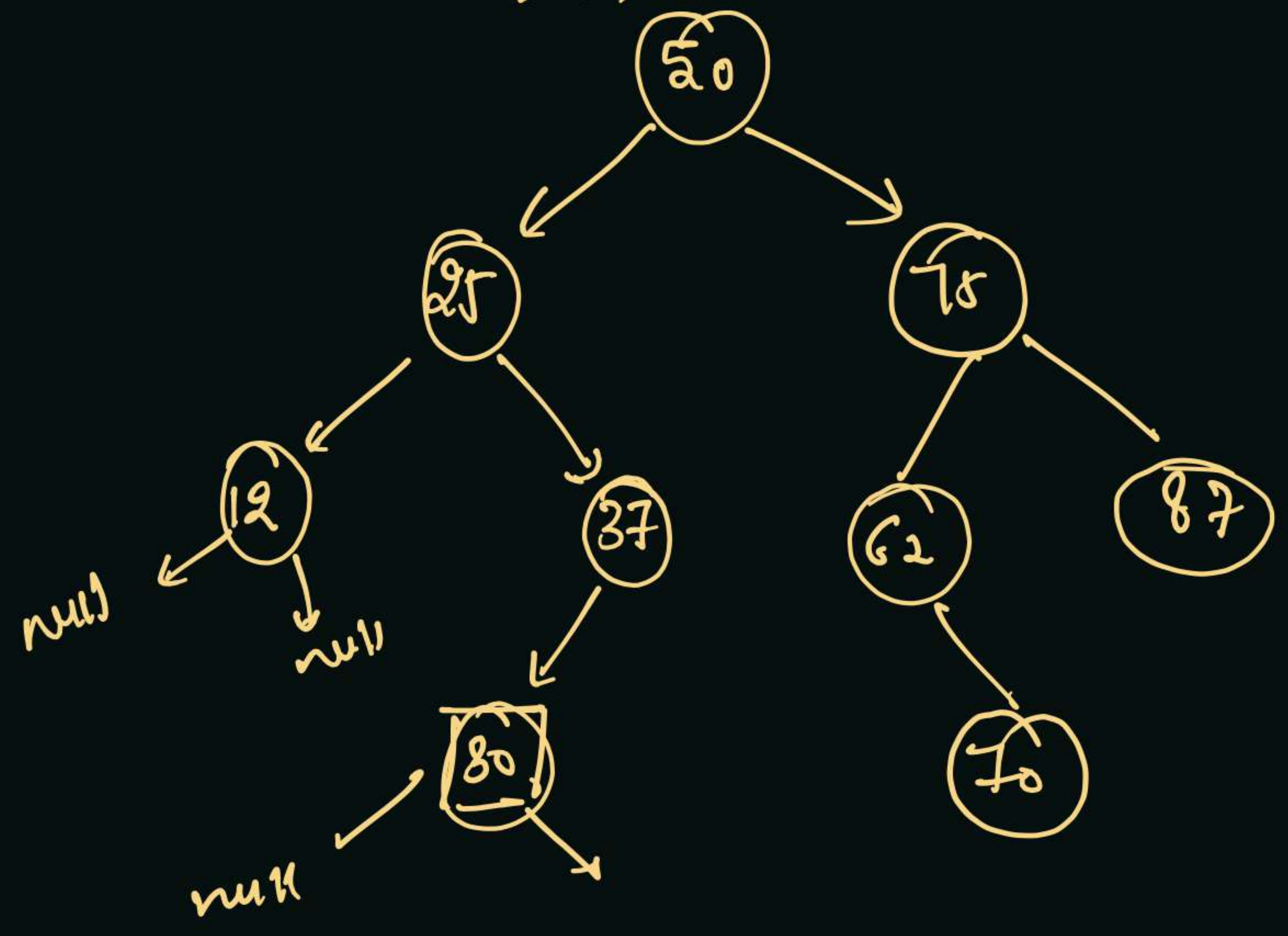
(5) 75

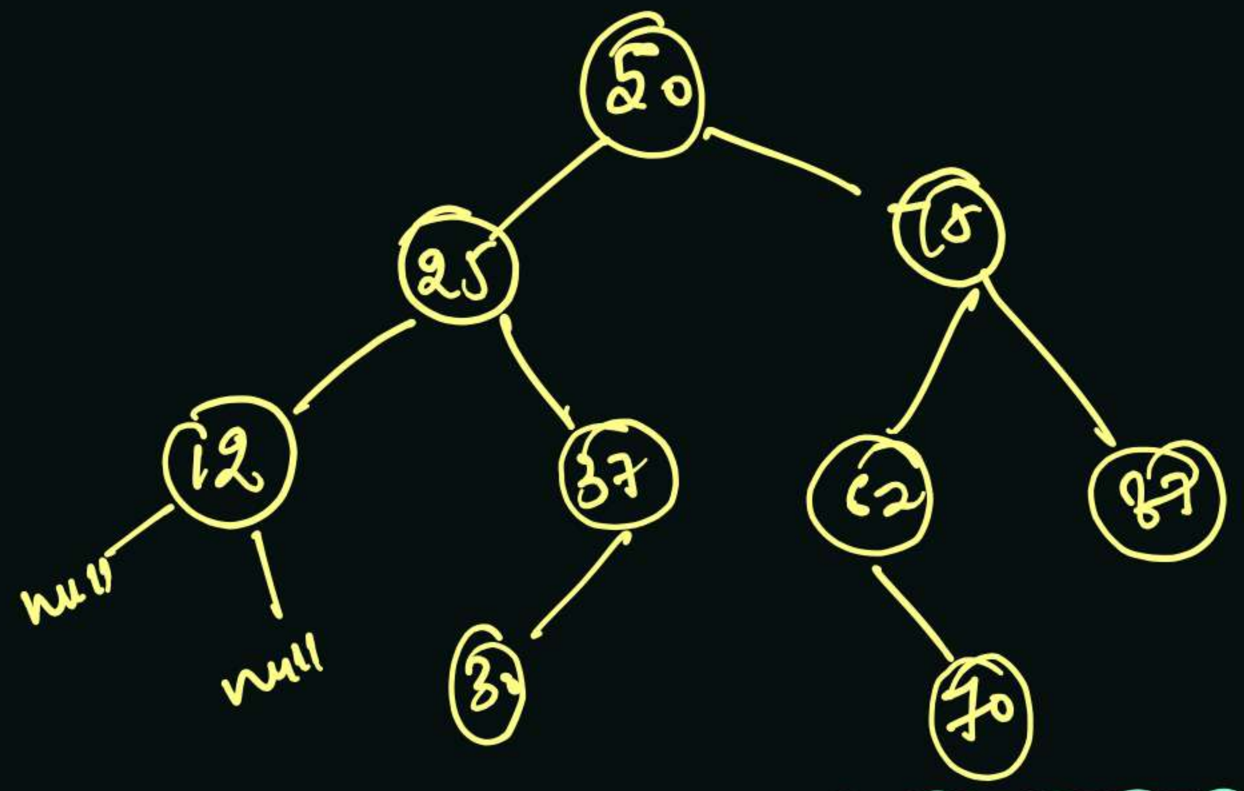
50, ∞, R

(0) 50 -∞, ∞, L

Basic Structure

1. Base case
   - Index out of Bond
   - Invalid Range of value
2. Recursive part
   - Tree Node
   - Indx ++
   - set left and right
   - Return Node

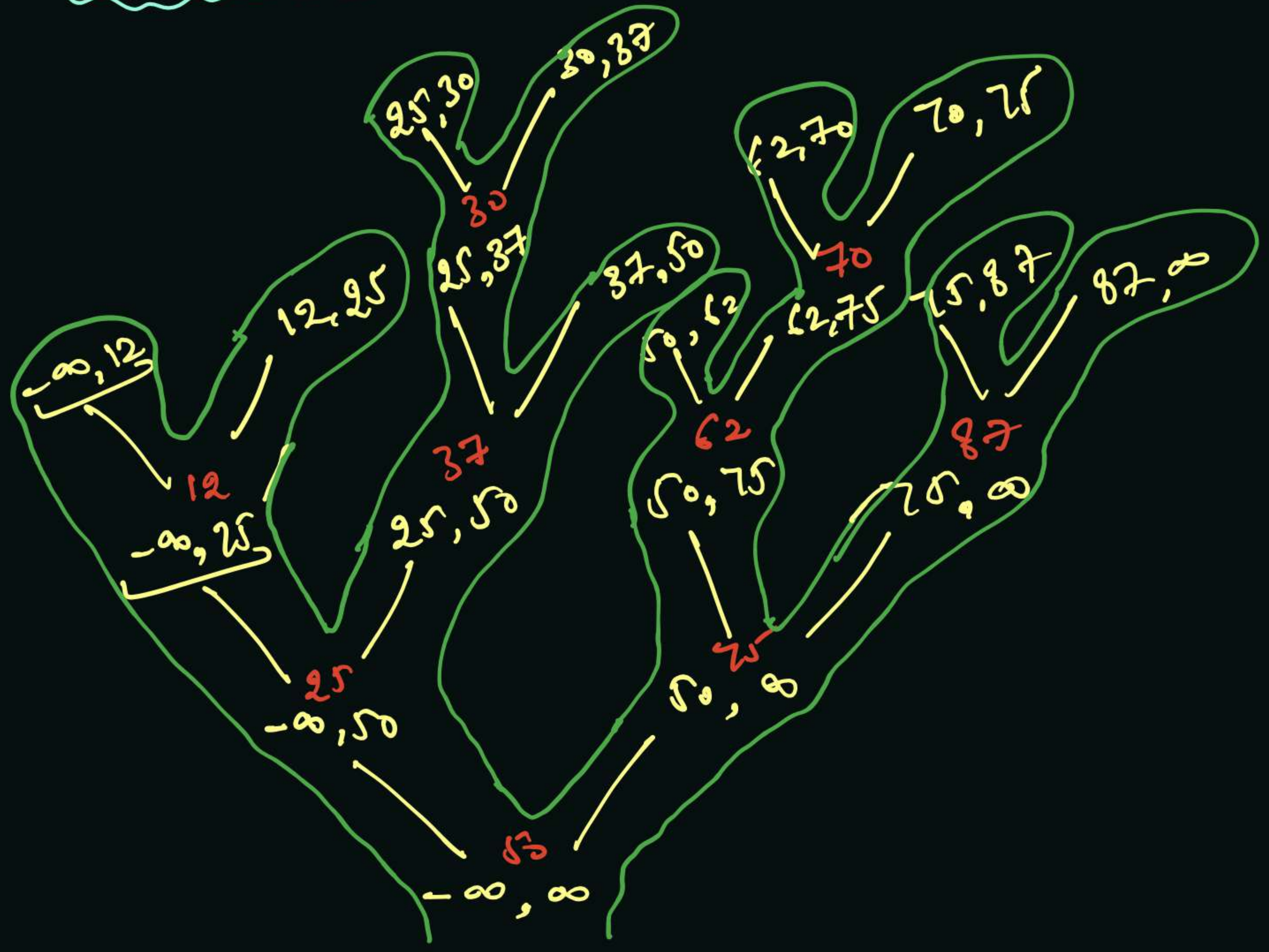




Pre Order → 50, 25, 12, 37, 30, 75, 62, 70, 87

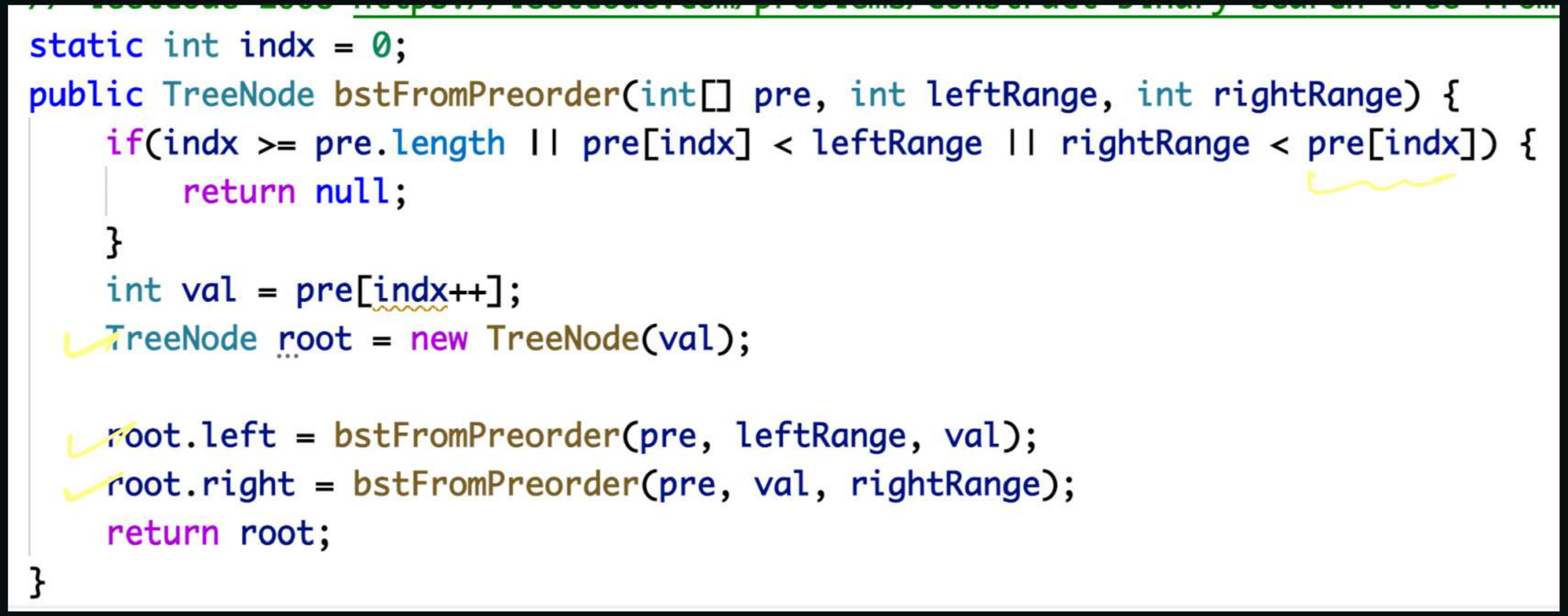
0 1 2 3 4 5 6 7 8

↑
Indx



Indx = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9

val = ~~50~~ ~~25~~ ~~12~~ ~~37~~ ~~30~~ ~~75~~ ~~62~~ ~~70~~ ~~87~~



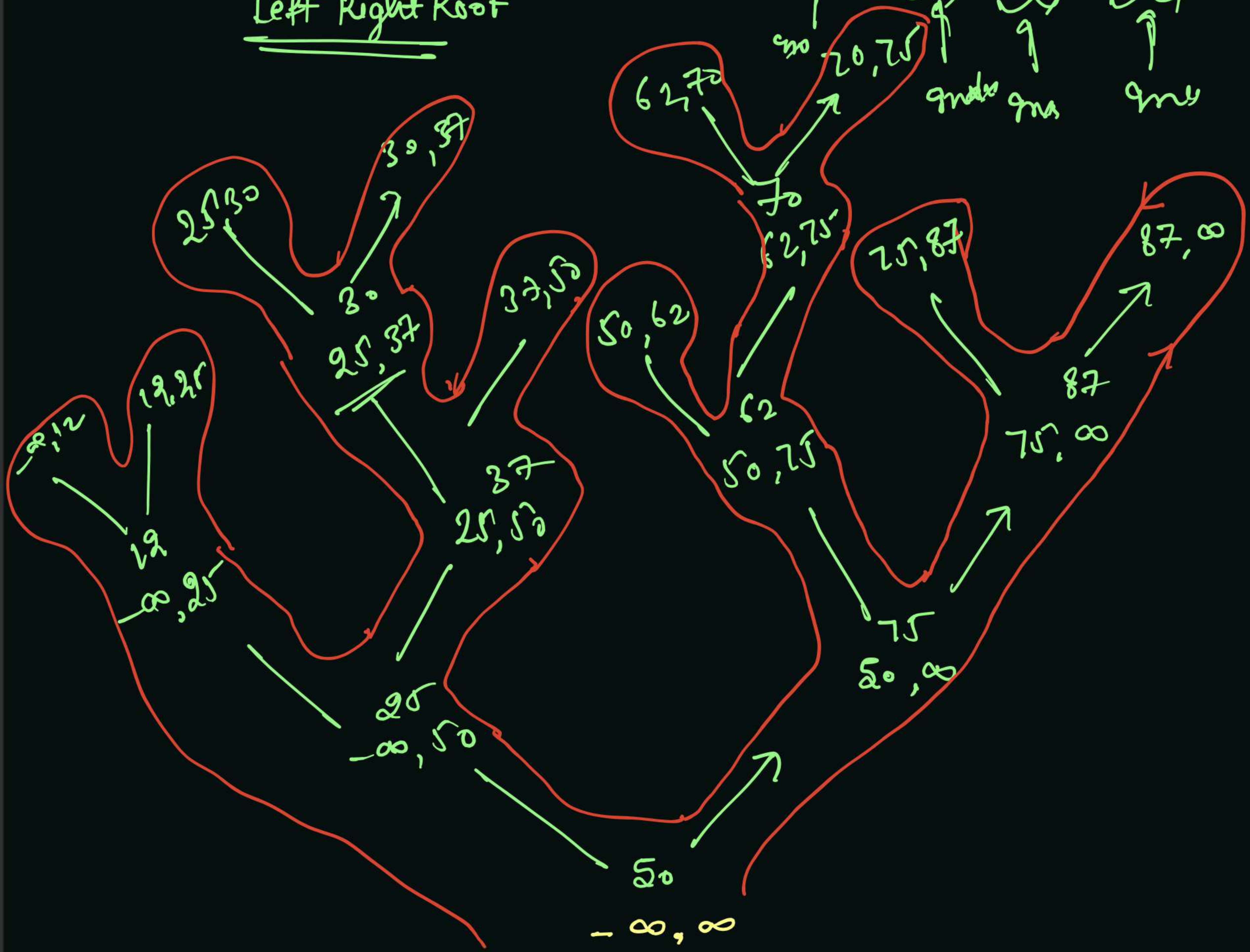
```
// leetcode 1008 https://leetcode.com/problems/construct-binary-search-tree-from
static int indx = 0;
public TreeNode bstFromPreorder(int[] pre, int leftRange, int rightRange) {
    if(indx >= pre.length || pre[indx] < leftRange || rightRange < pre[indx]) {
        return null;
    }
    int val = pre[indx++];
    TreeNode root = new TreeNode(val);

    root.left = bstFromPreorder(pre, leftRange, val);
    root.right = bstFromPreorder(pre, val, rightRange);
    return root;
}
```

Construct BST Using Postorder

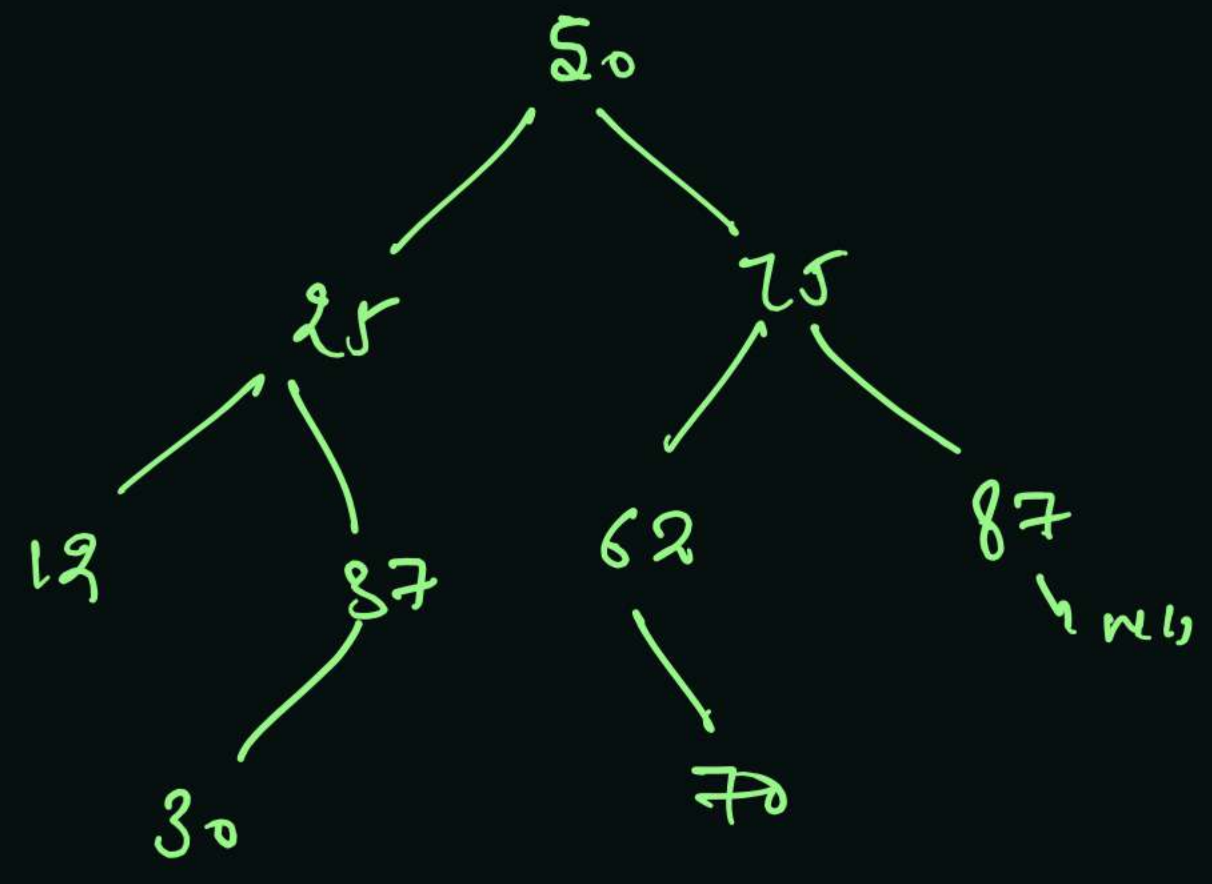
Saturday, 14 August 2021 10:13 AM

Left Right Root

	0	1	2	3	4	5	6	7	8
Postorder →	12,	30,	37,	25,	70,	62,	87,	75,	50
	indx	indx	indx	indx	indx	ix	Indx		Indx



-∞, 12
12, 25
12
-∞, 25
25, 30
30, 37
30
25, 37
37, 50
37
25, 50
25
-∞, 50
62, 70
70, 75
70
62, 75
50, 62
62
50, 75
75, 87
87, ∞
87
75, ∞
75
50, ∞
50
-∞, ∞



50
25
75
12
37
62
87
30
70
null