Trees

→ i/p

L N R → root

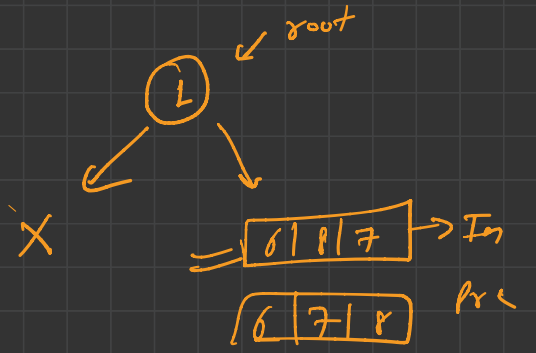
Left Root Right

Inorder → [1 | 6 8 7]

PreOrder → [1 | 6 7 8]

NLR → Root Left Right

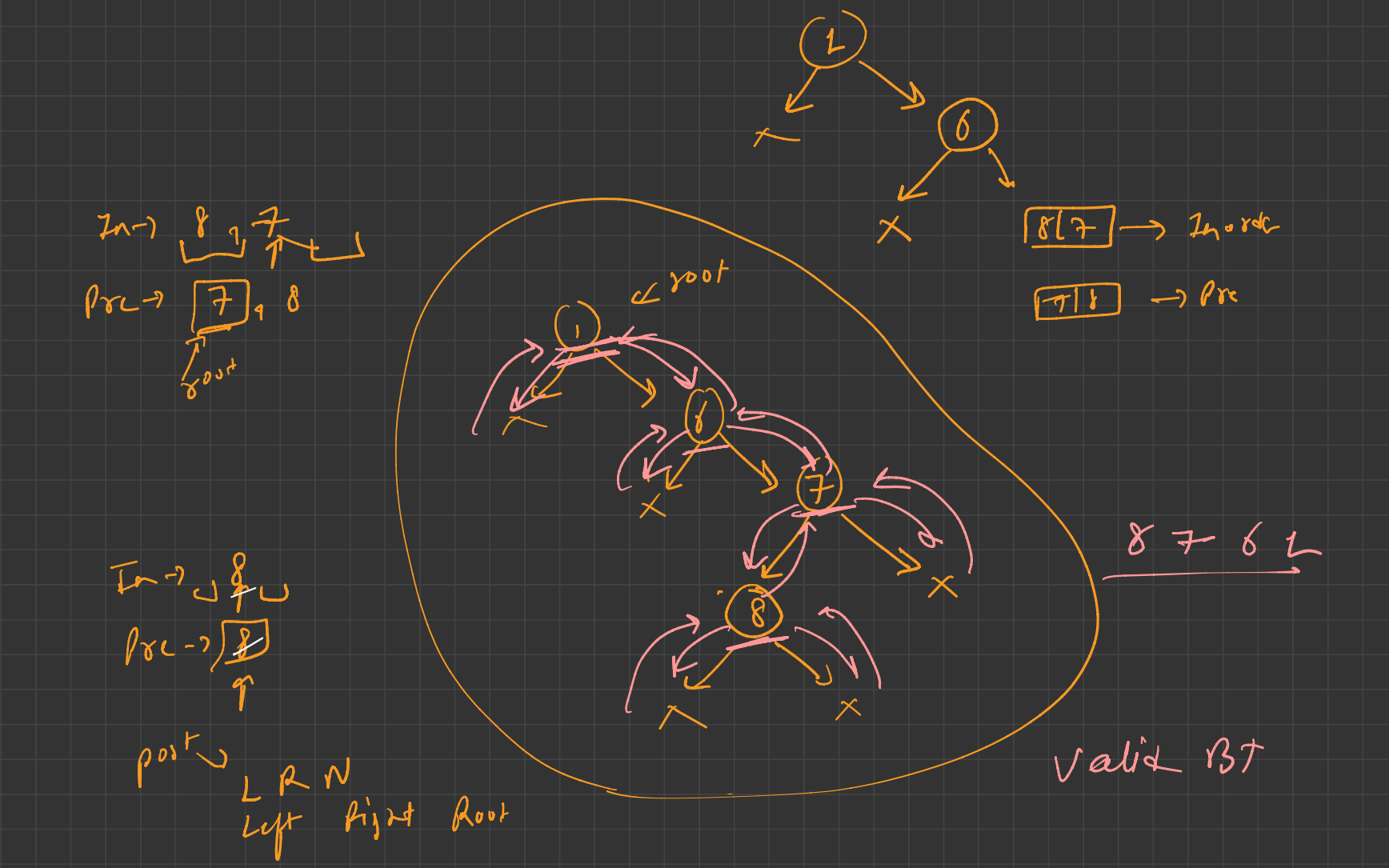
o/p →

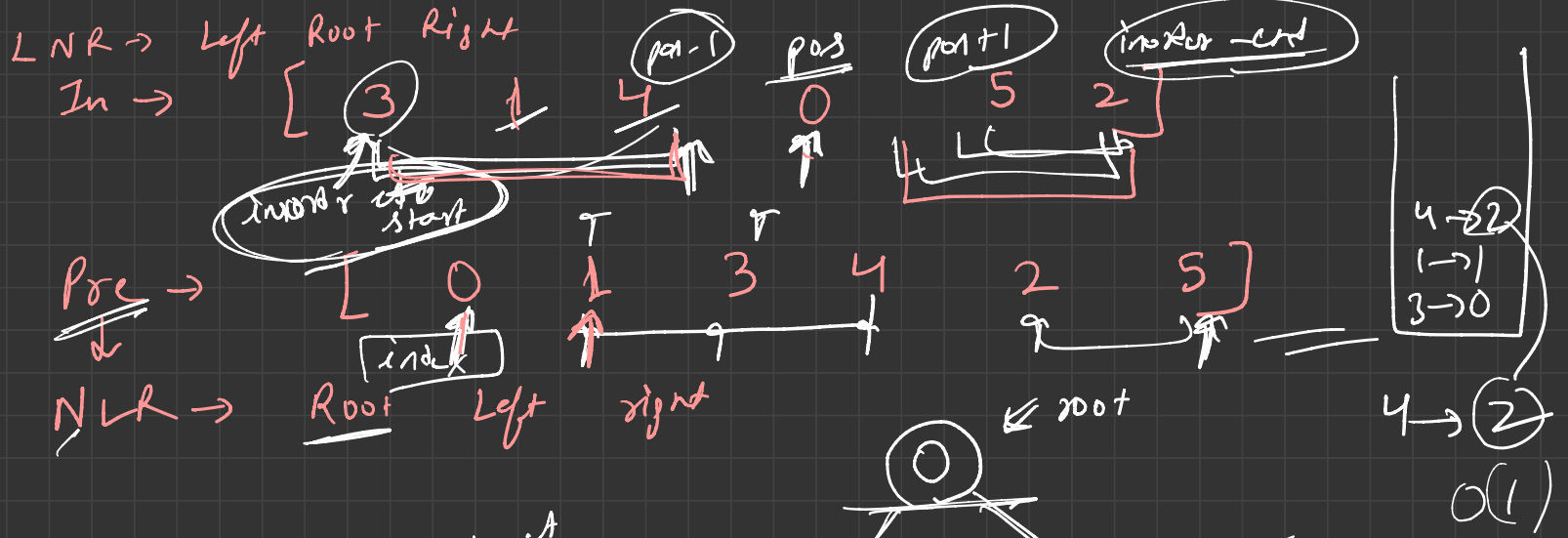


1

In → 6 | 8 7

Pre → 6 | 7 8


In→ 8 4 7
Pre→ 7 4 8
root
8 | 7 → Inorder
7 | 8 → Pre
root
8 7 6 4
In→ 8
Pre→ 8
post → L R N
Left Right Root
Valid BT

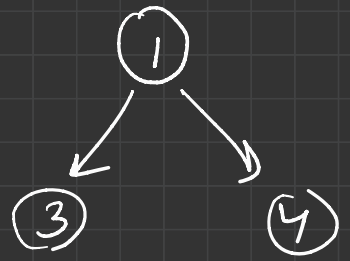


Algo:-

- take index element as root element
- find root element ki position in inorder
- recursion
  - root → left (inorder start → pos - 1)
  - root → right (pos + 1, inorder end)

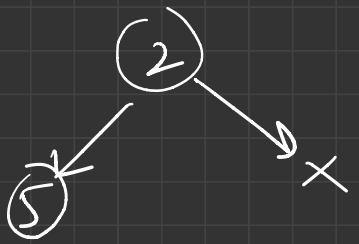
In → 3 1 4

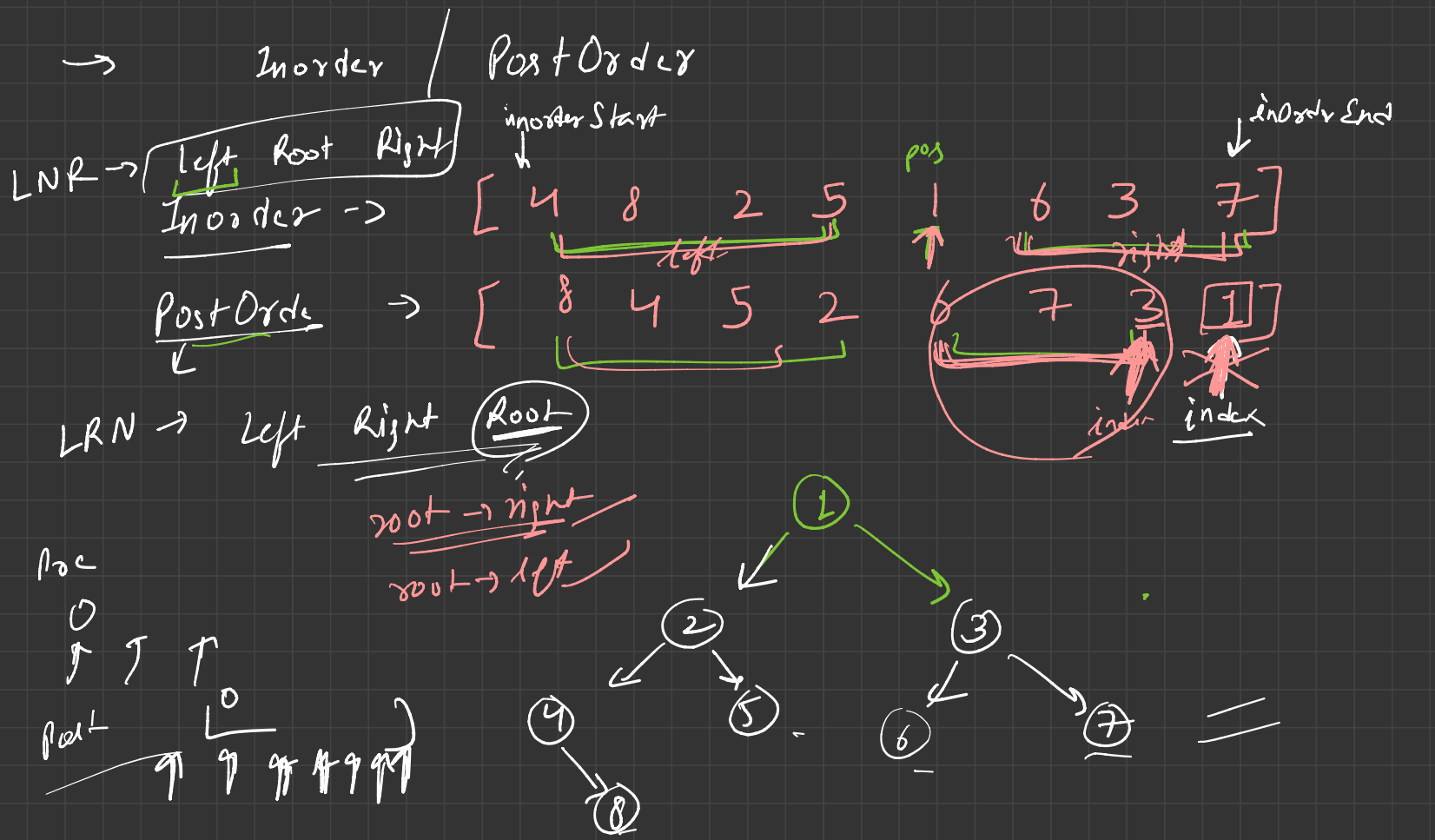
Pre → 1 3 4

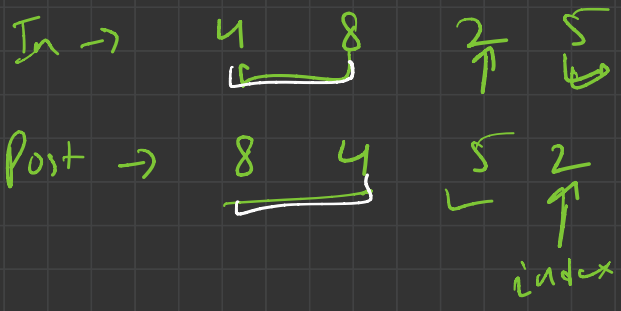


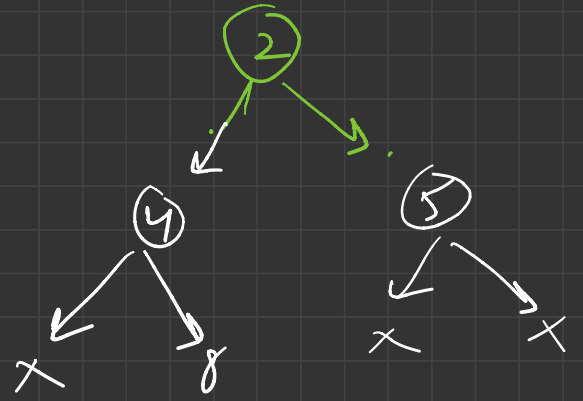
In → 5 2

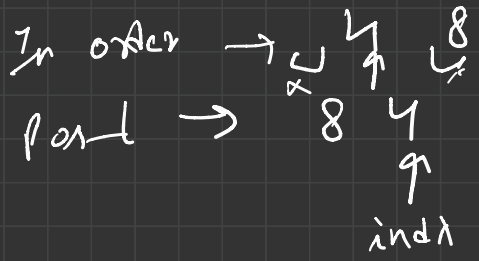
Pre → 2 5

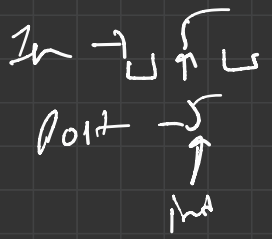


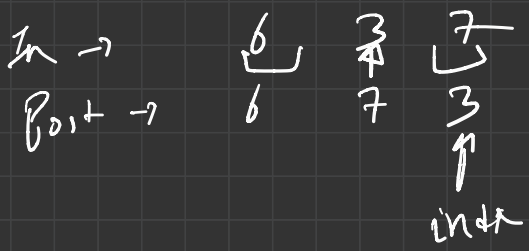

Inorder
Post Order
LNR → Left Root Right
Inorder →
Post Order →
LRN → Left Right Root
inorderStart
inOrderEnd
pos
4 8 2 5
8 4 5 2
1 6 3 7
6 7 3 1
left
right
root → right
root → left
index
1
2
3
4
5
6
7
8

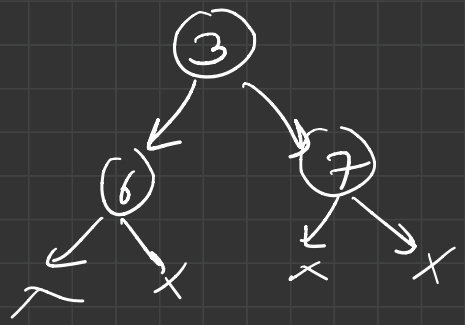

In →
4 8 2 5
Post →
8 4 5 2
index


2
4
5
x
y
x
x


In order →
Post →
8 4
index


In →
Post →
index





→ BT → construct

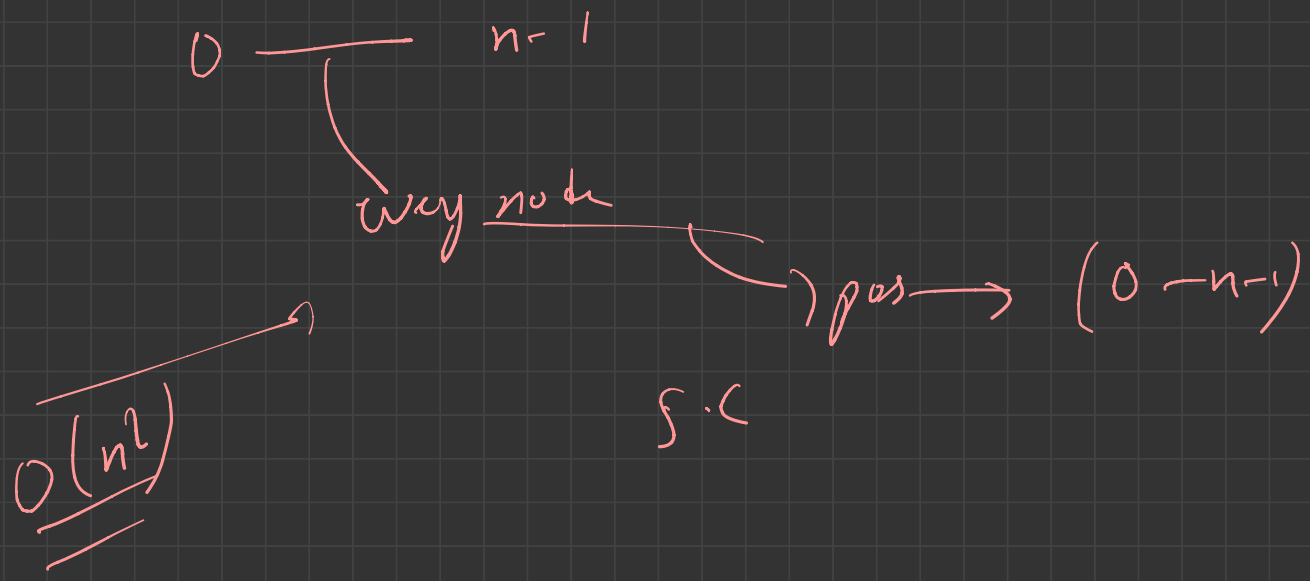
→ In / Pre ⟹ =

→ In / Post ⟹

T.C / S.C

O(nlogn)

= O(n) + O(nlogn) → O(nlogn)


0
n-1
way node
pos → (0 - n-1)
O(n!)
S.C