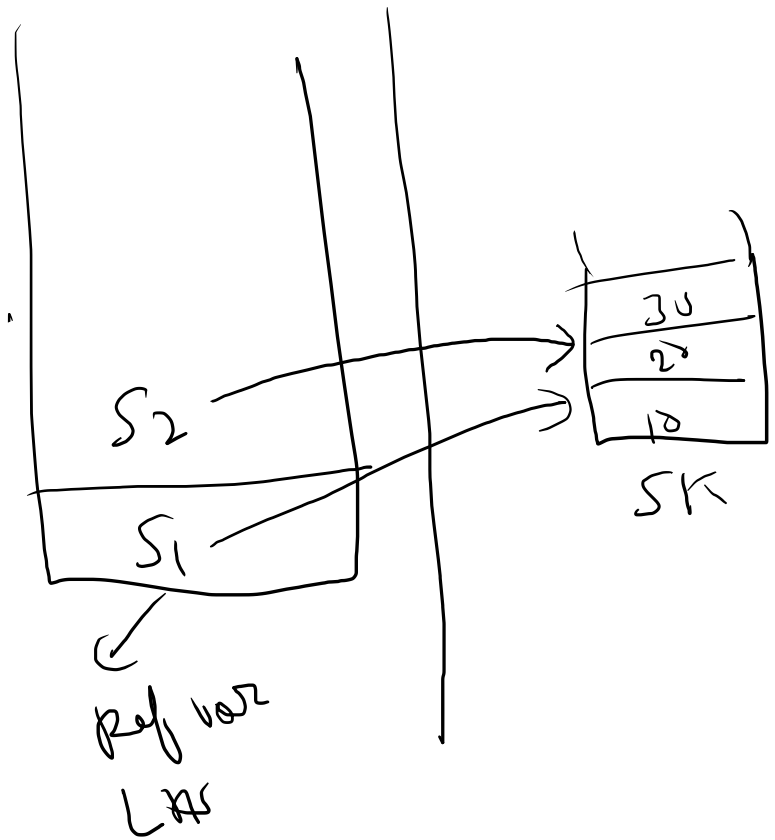
AL 12

Ref vars

LHS RHS

Stack S_1 = new Stack

Stack S_2 = S_1



Java

ctt

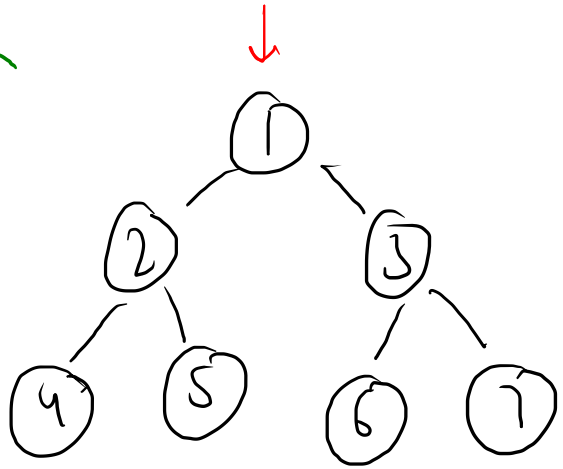
Construct BT from Inorder & Post order

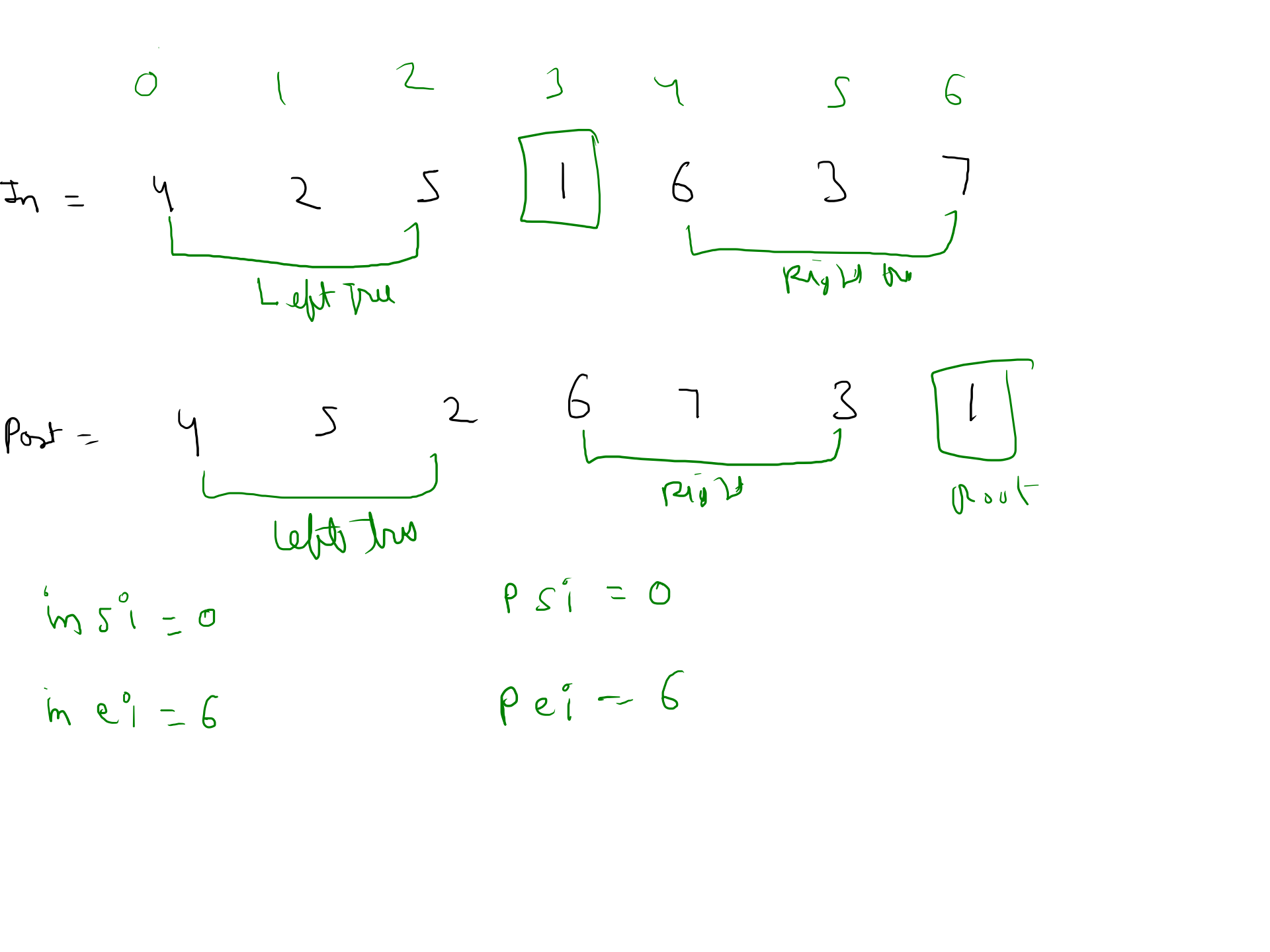
In = 4 2 5 1 6 3 7

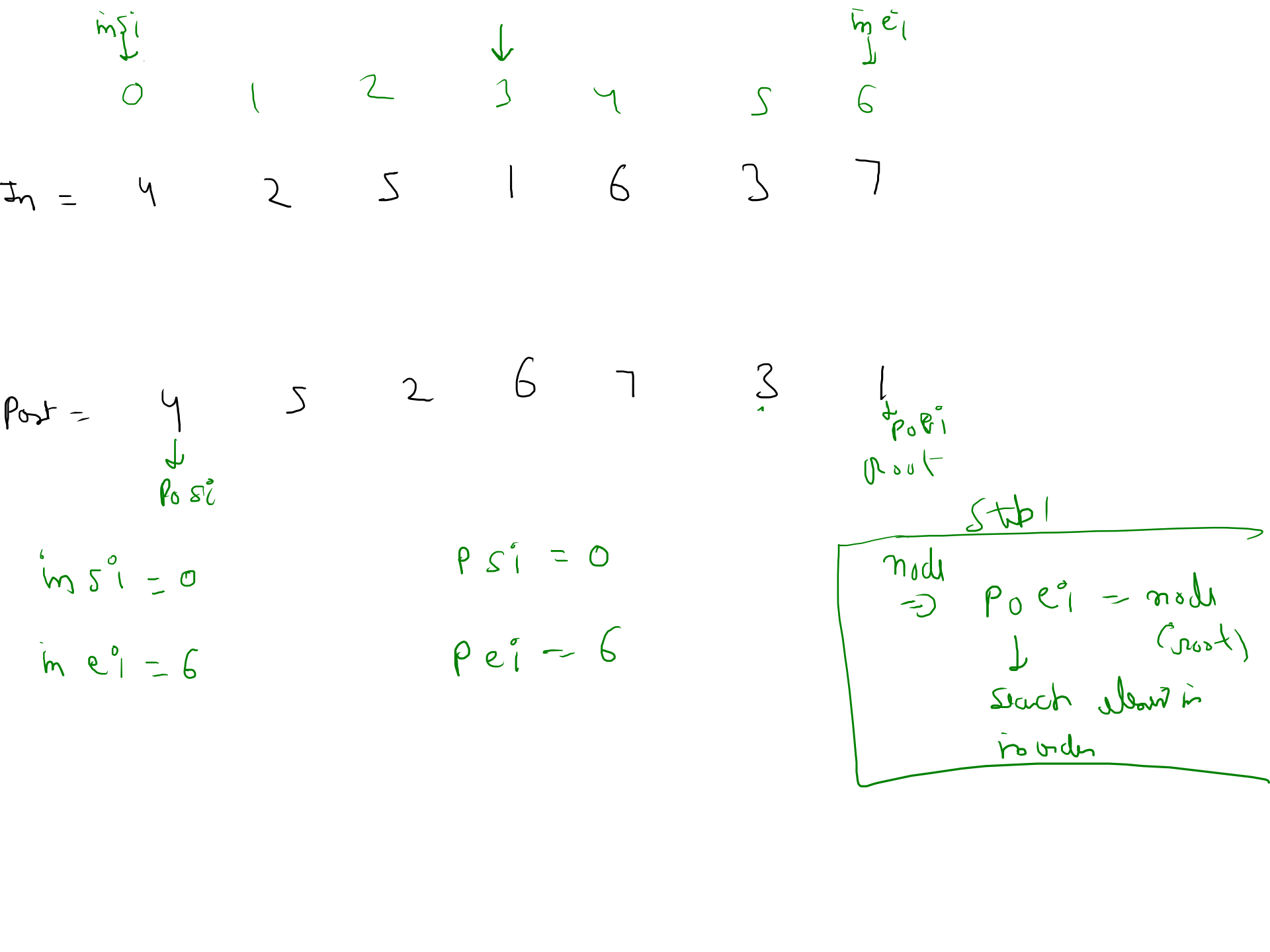
Post = 4 5 2 6 7 3 1

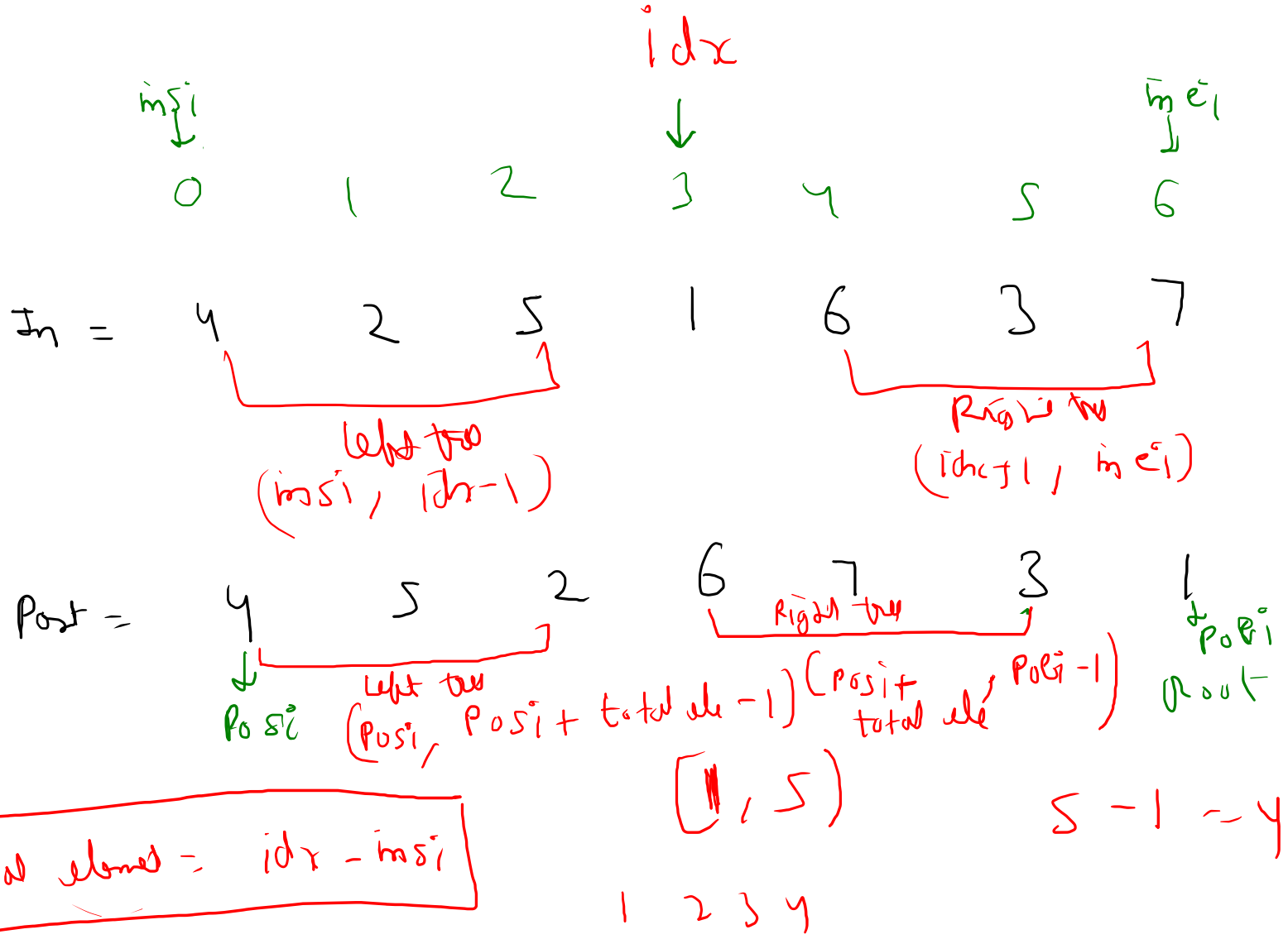
Inorder => L N R

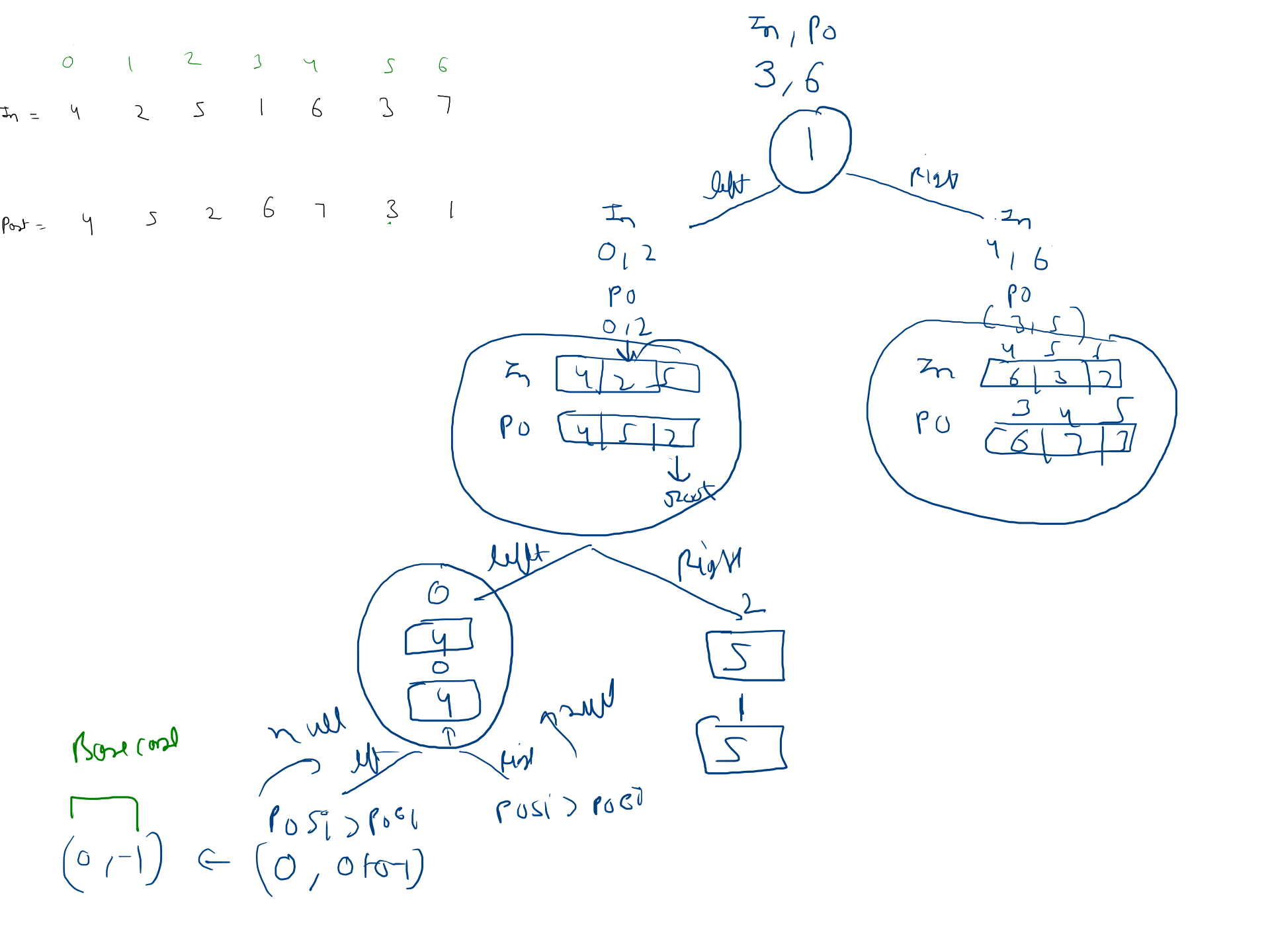
Post = L R N



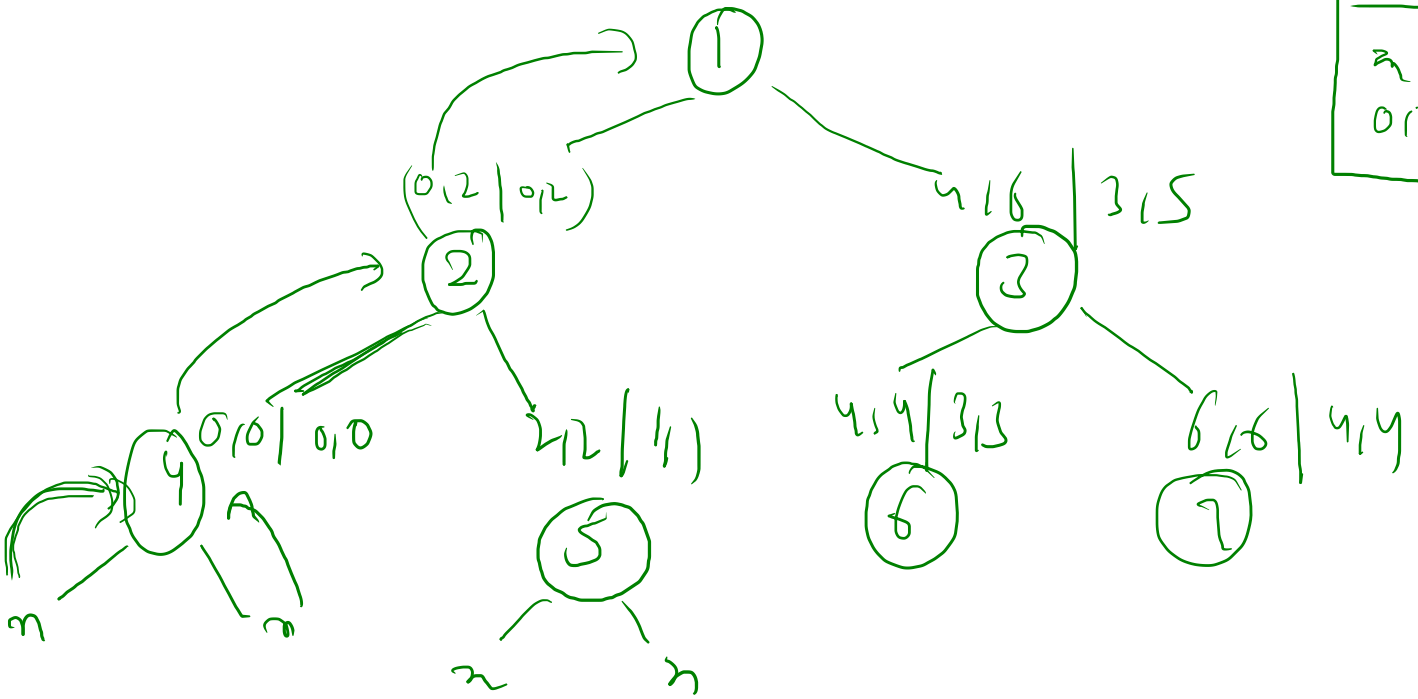

0 1 2 3 4 5 6
In = 4 2 5 1 6 3 7
Left Tree
Right Tree
Post = 4 5 2 6 7 3 1
Left Tree
Right
Root
ins i = 0
ine i = 6
psi = 0
pei = 6


ins
ine
0 1 2 3 4 5 6
In = 4 2 5 1 6 3 7
Post = 4 5 2 6 7 3 1
Posi
PoEi
Root
ins i = 0
ine i = 6
Psi = 0
Pei = 6
Stbl
node
⇒ Poei = node (root)
↓
search node in inorder

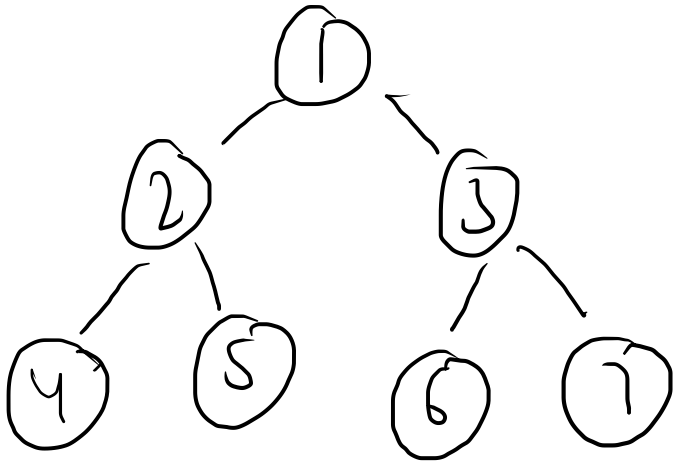

idx
insi
inei
0 1 2 3 4 5 6
In = 4 2 5 1 6 3 7
left tree
(insi, idx-1)
Right tree
(idx+1, inei)
Post = 4 5 2 6 7 3 1
Posi
left tree
(Posi, Posi + total ele - 1)
Right tree
(posi + total ele, Posi - 1)
Root
total elements = idx - insi
[1, 5)
5 - 1 = 4
1 2 3 4


0 1 2 3 4 5 6
In = 4 2 5 1 6 3 7
Post = 4 5 2 6 7 3 1
In, Po
3, 6
1
left
right
In
0, 2
Po
0, 2
In
4 2 5
Po
4 5 2
root
In
4, 6
Po
(3+5)
4 5 6
In
6 3 7
Po
3 4 5
6 7 7
left
right
0
4
0
4
2
5
1
5
null
left
right
Posi > Poei
Posi > Poei
Base cond
(0, -1) ← (0, 0-1)

	1	2	3	4	5	6
In =	4	2	5	1	6	3	7
Post =	4	5	2	6	7	3	1



Construct Post & Pre order

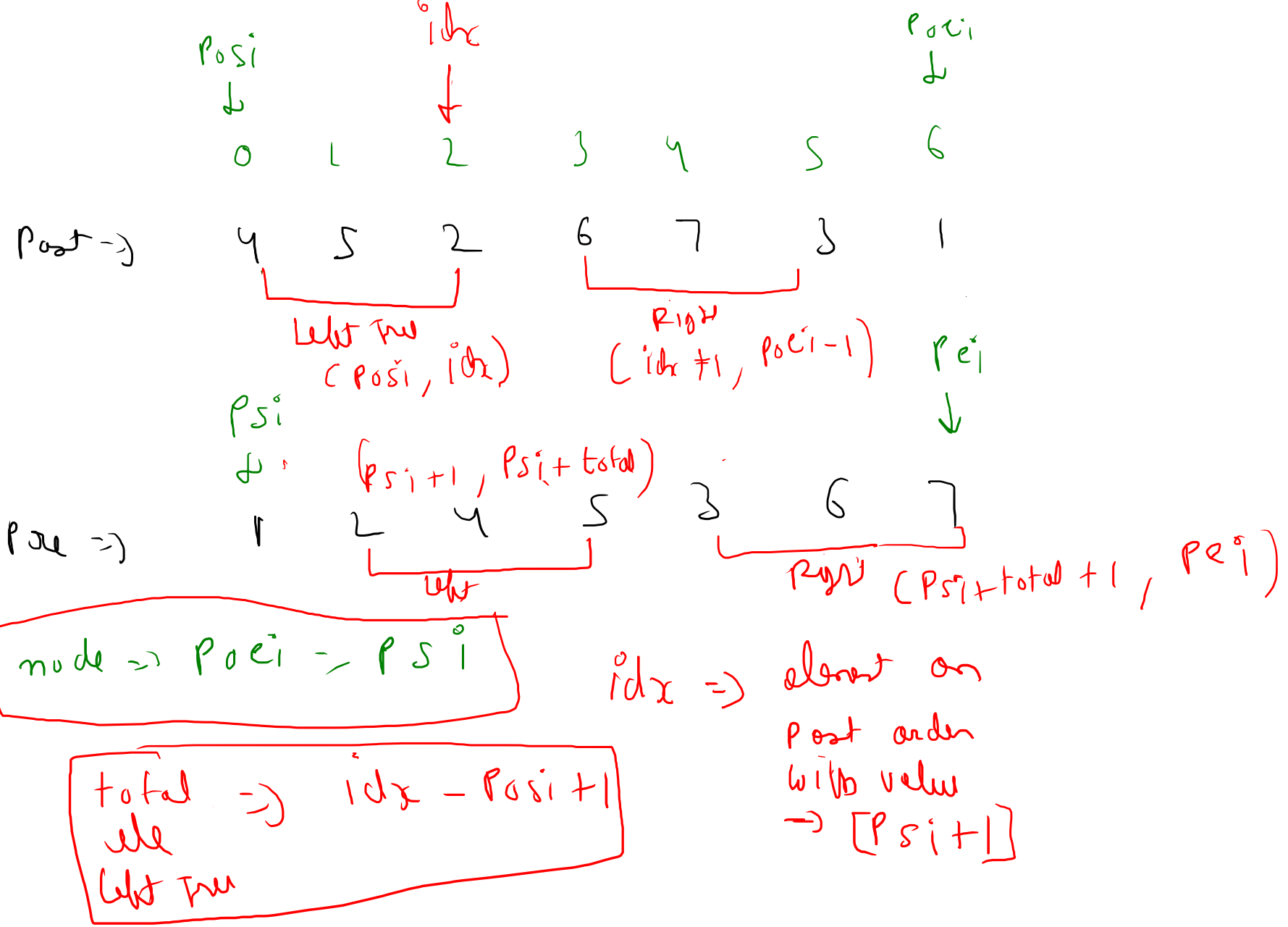


Post -> 4 5 2 6 7 3 1

Pre -> 1 2 4 5 3 6 7

Post -> L R N

Pre -> N L R


posi
idx
poei
0 1 2 3 4 5 6
post =>
4 5 2 6 7 3 1
Left Tree
(posi, idx)
Right
(idx+1, poei-1)
pei
psi
(psi+1, psi+total)
pre =>
1 2 4 5 3 6 7
Left
Right
(psi+total+1, pei)
node => poei - psi
idx => element on post order with value => [psi+1]
total => idx - posi + 1 ie Left Tree