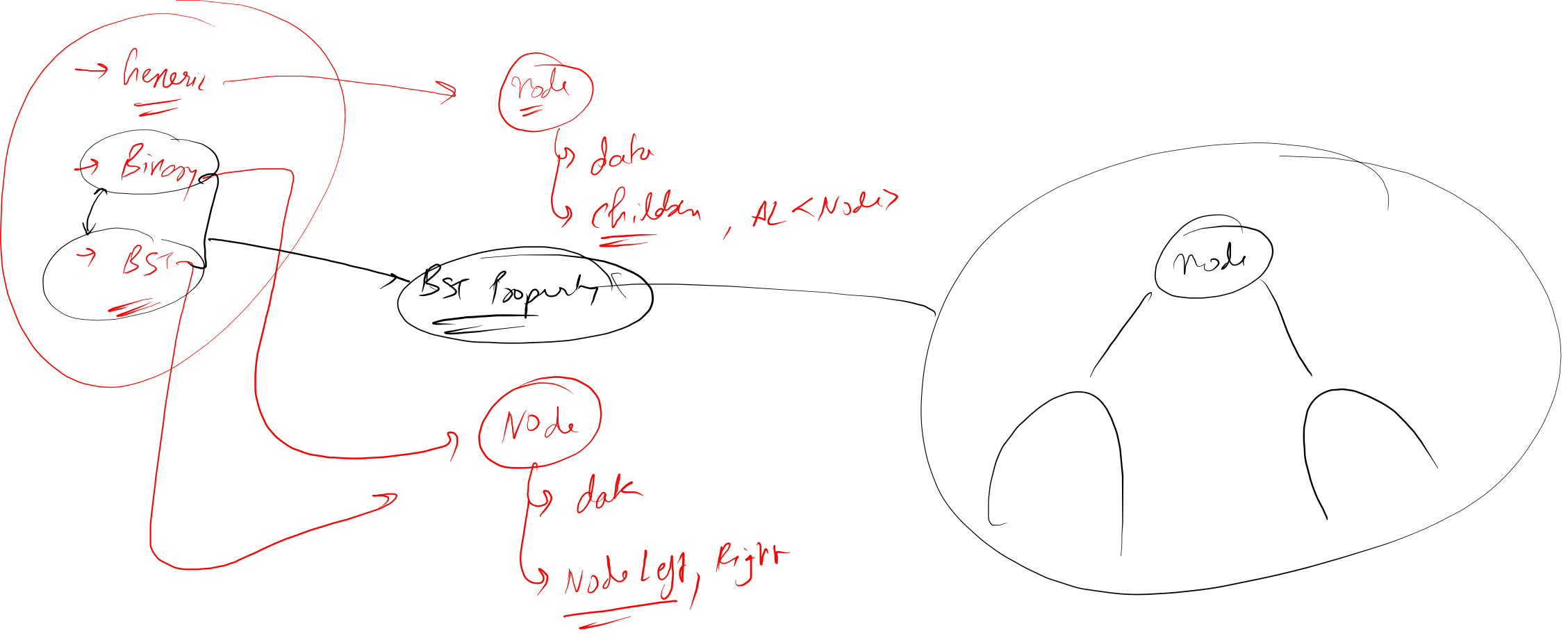
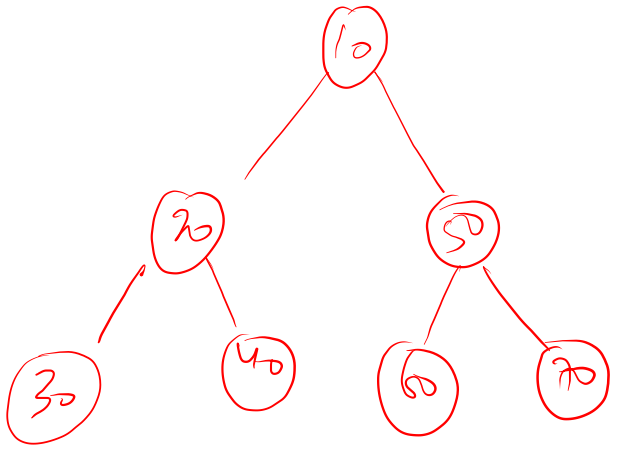

Generic
Binary
BST
Node
data
children, AL<Node>
BST Property
Node
data
Node left, Right
node



Pre, In, Post, level

Pre ← 10 20 30 40 50 60 70 [NLR]

In ← 30 20 40 10 60 50 70 [LNR]

Post ← 30 40 20 60 70 50 10 [LRN]

level ← 10 20 50 30 40 60 70

psi pei

	0	1	2	3	4	5	6	
Pre	4	2	1	3	6	5	7	[NLR]
In	1	2	3	4	5	6	7	[LNR]

isi iei

psi > pei

psi pei isi iei

(preodr, 0, 6, inordr, 0, 6)

val = preodr[psi]

idx

tnel = idx − isi

val

(preodr, psi+1, psi+tnel, inordr, isi, idx−1)

(preodr, psi+tnel+1, pei, inordr, idx+1, iei)

psi pei isi iei

(0, 6, 0, 6)

val = 4 idx: 3

tnel: 3

4

(1, 3, 0, 2)

val = 2, idx: 1, tnel = 1

(4, 6, 4, 6)

val = 6, idx: 5

6 tnel = 1

2

(3, 3, 2, 2)

val = 3, idx = 2

tnel = 0

(2, 2, 0, 0)

val = 1, idx: 0, tnel = 0

1

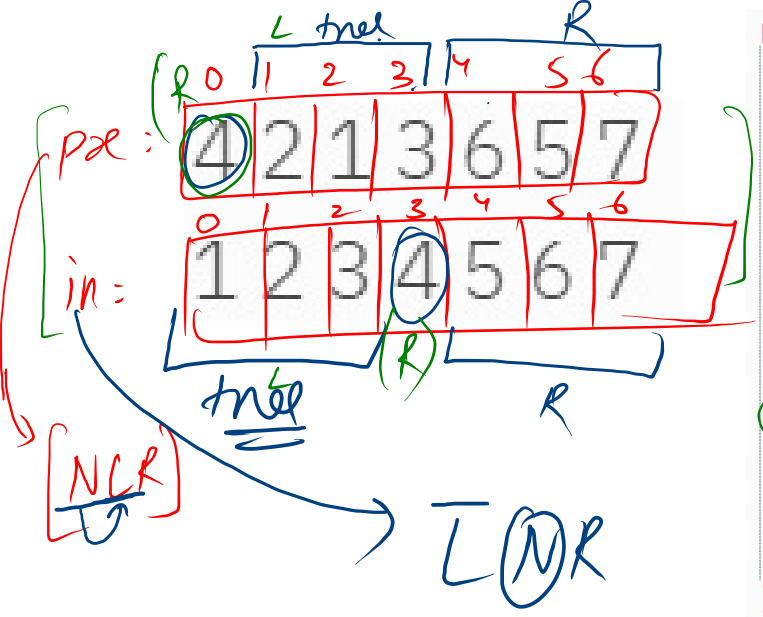
3

(3, 2, 0, −1)

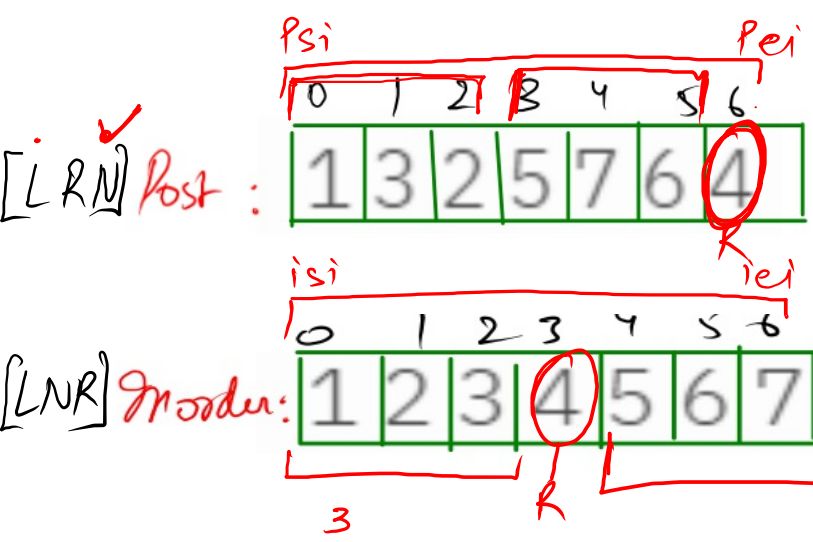
(3, 2, 1, 0)

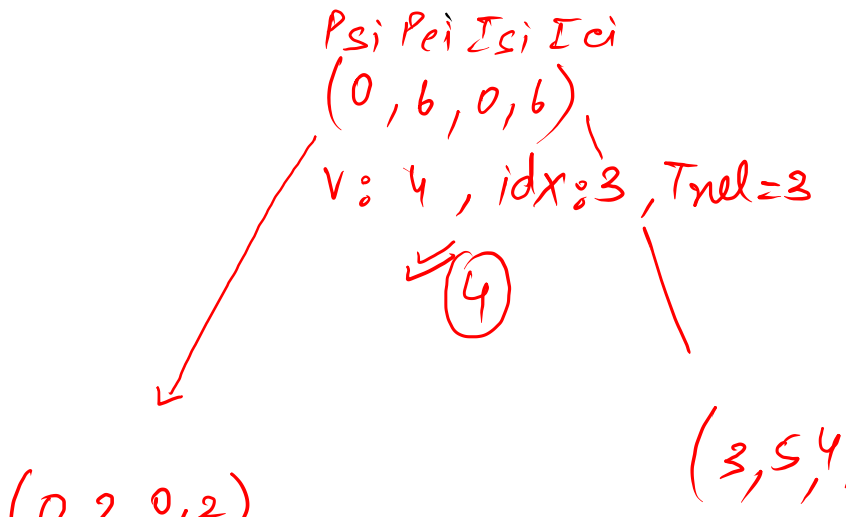
(4, 3, 2, 1)

(4, 3, 3, 2)




```java
private static TreeNode buildTree(int preorder[],int psi, int pei, int inorder[],int isi,int iei){
    if(psi > pei){
        return null;
    }

    int val = preorder[psi];
    int idx = isi;
    while(inorder[idx] != val){
        idx++;
    }
    int tnel = idx - isi;
    TreeNode node = new TreeNode(val);

    node.left = buildTree(preorder,psi+1,psi+tnel,inorder,isi,idx-1);
    node.right = buildTree(preorder,psi+tnel+1,pei,inorder,idx+1,iei);

    return node;
}
public static TreeNode buildTree(int[] preorder, int[] inorder) {

    return buildTree(preorder,0,preorder.length-1,inorder,0,inorder.length-1);
}
```





tnel: 3

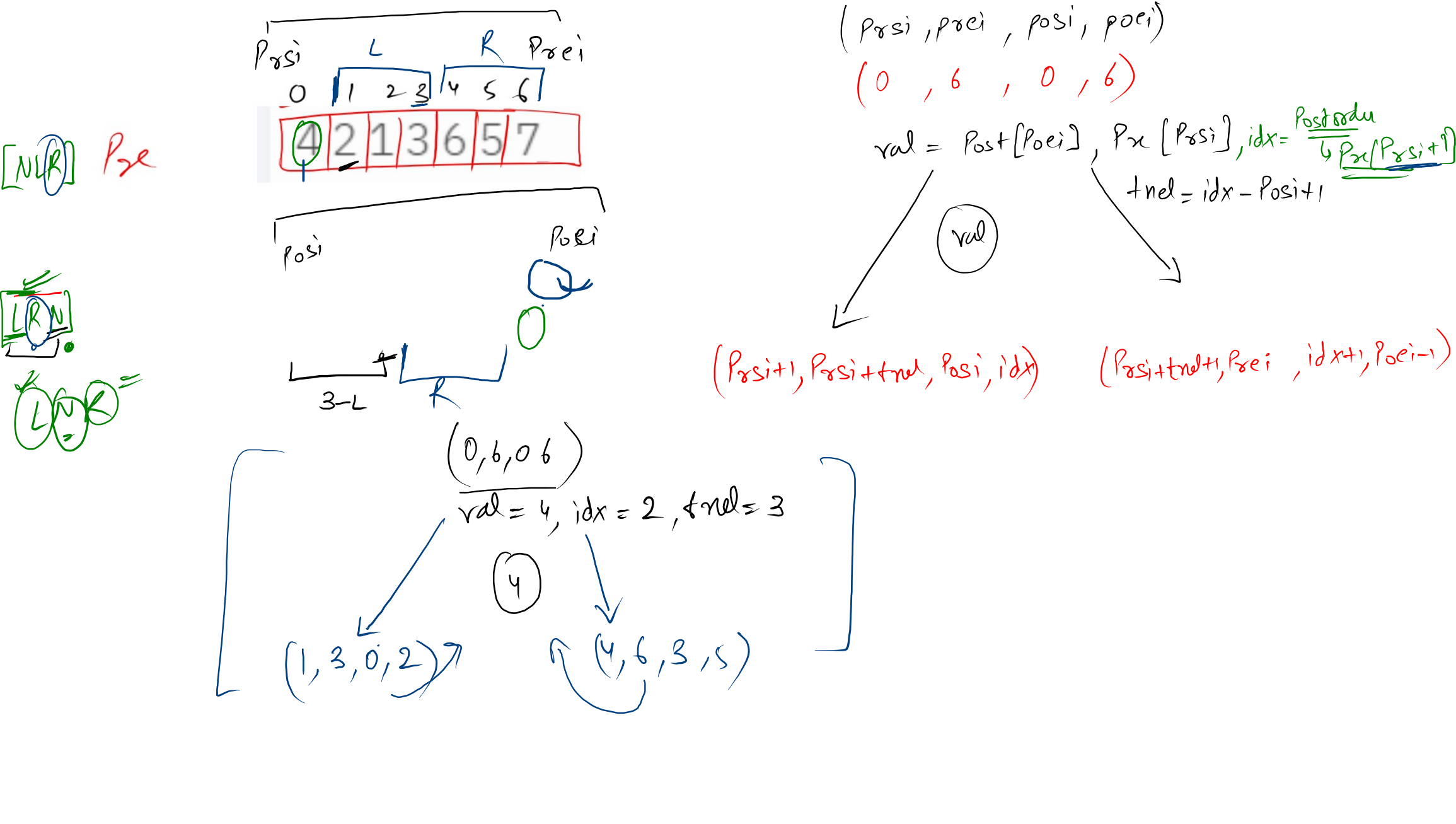
Psi Pei isi iei

(0, 6, 0, 6)

val = post[pei], idx, tnel : idx - isi

(psi, psi+tnel-1, isi, idx-1) (psi+tnel, pei-1, idx+1, iei)

```java
public static TreeNode buildTree(int[] inorder,int isi,int iei, int[] postorder,int psi,int pei) {
    if(psi > pei){
        return null;
    }

    int val = postorder[pei];
    TreeNode node = new TreeNode(val);

    int idx = isi;
    while(inorder[idx] != val){
        idx++;
    }

    int tnel = idx - isi;

    node.left = buildTree(inorder,isi,idx-1,postorder,psi,psi+tnel-1);
    node.right = buildTree(inorder,idx+1,iei,postorder,psi+tnel,pei-1);

    return node;
}
```


Prsi
L
R
Prei
0 1 2 3 4 5 6
4 2 1 3 6 5 7
[N L R] Pre
Posi
Poei
LRN
LNR
3-L
R
(0,6,0 6)
val = 4, idx = 2, tnel = 3
4
(1, 3, 0, 2)
(4, 6, 3, 5)
(Prsi, Prei, Posi, Poei)
(0, 6, 0, 6)
val = Post[Poei], Pre[Prsi], idx = Postordu / Pre[Prsi+1]
tnel = idx - Posi + 1
val
(Prsi+1, Prsi+tnel, Posi, idx)
(Prsi+tnel+1, Prei, idx+1, Poei-1)

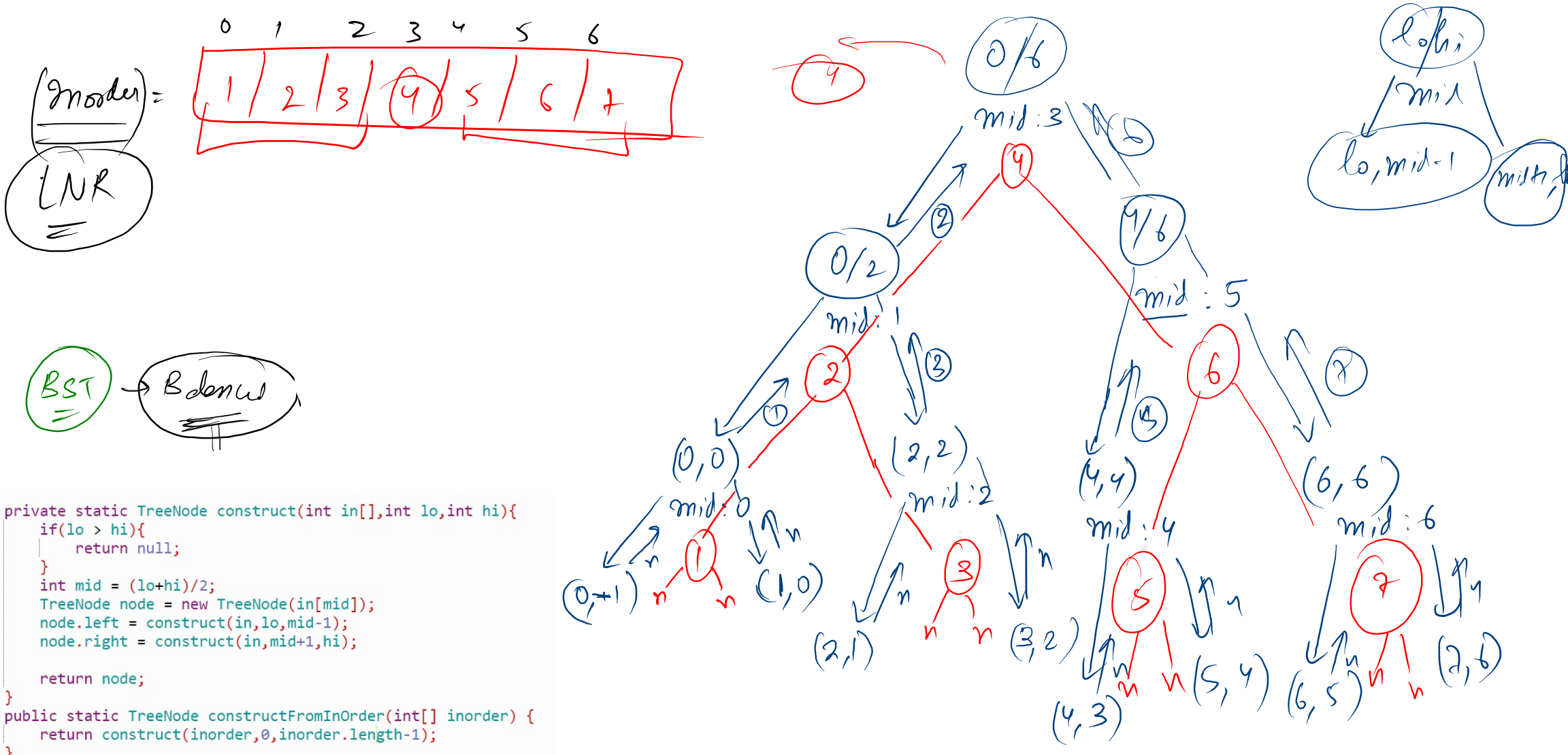
```java
public static TreeNode constructFromPrePost(int[] pre, int prsi, int prei,int[] post, int posi, int poei)
    if(prsi > prei){
        return null;
    }
    int val = post[poei];
    TreeNode node = new TreeNode(val);

    if(prsi == prei){
        return node;
    }

    int idx = posi;
    while(post[idx] != pre[prsi+1]){
        idx++;
    }

    int tnel = idx - posi + 1;
    node.left = constructFromPrePost(pre,prsi+1,prsi+tnel,post,posi,idx);
    node.right = constructFromPrePost(pre,prsi+tnel+1,prei,post,idx+1,poei-1);

    return node;
```

$\Rightarrow O(1)$

$\Rightarrow O(n)$

B.C. $T(n) = O(n) + 2T(n/2)$

W.C. $T(n) = O(n) + T(n-1)$




```
private static TreeNode construct(int in[],int lo,int hi){
    if(lo > hi){
        return null;
    }
    int mid = (lo+hi)/2;
    TreeNode node = new TreeNode(in[mid]);
    node.left = construct(in,lo,mid-1);
    node.right = construct(in,mid+1,hi);

    return node;
}
public static TreeNode constructFromInOrder(int[] inorder) {
    return construct(inorder,0,inorder.length-1);
}
```

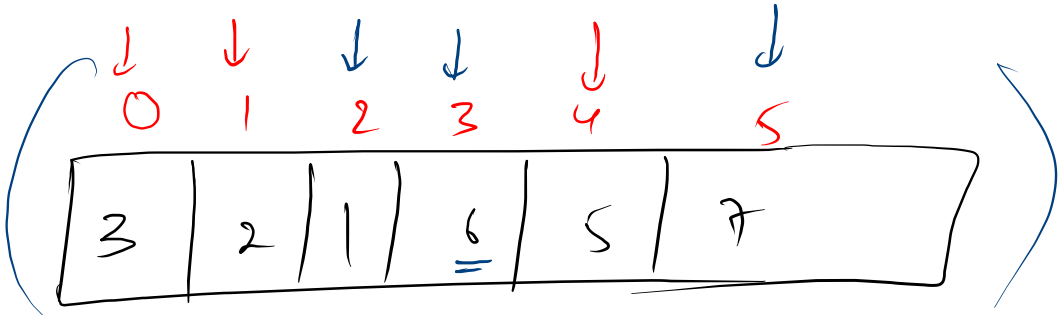
321657

Preorder

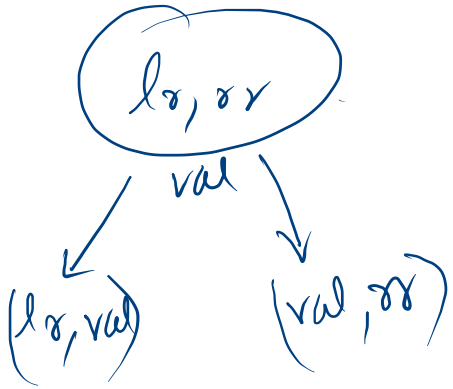
[N, L, R]

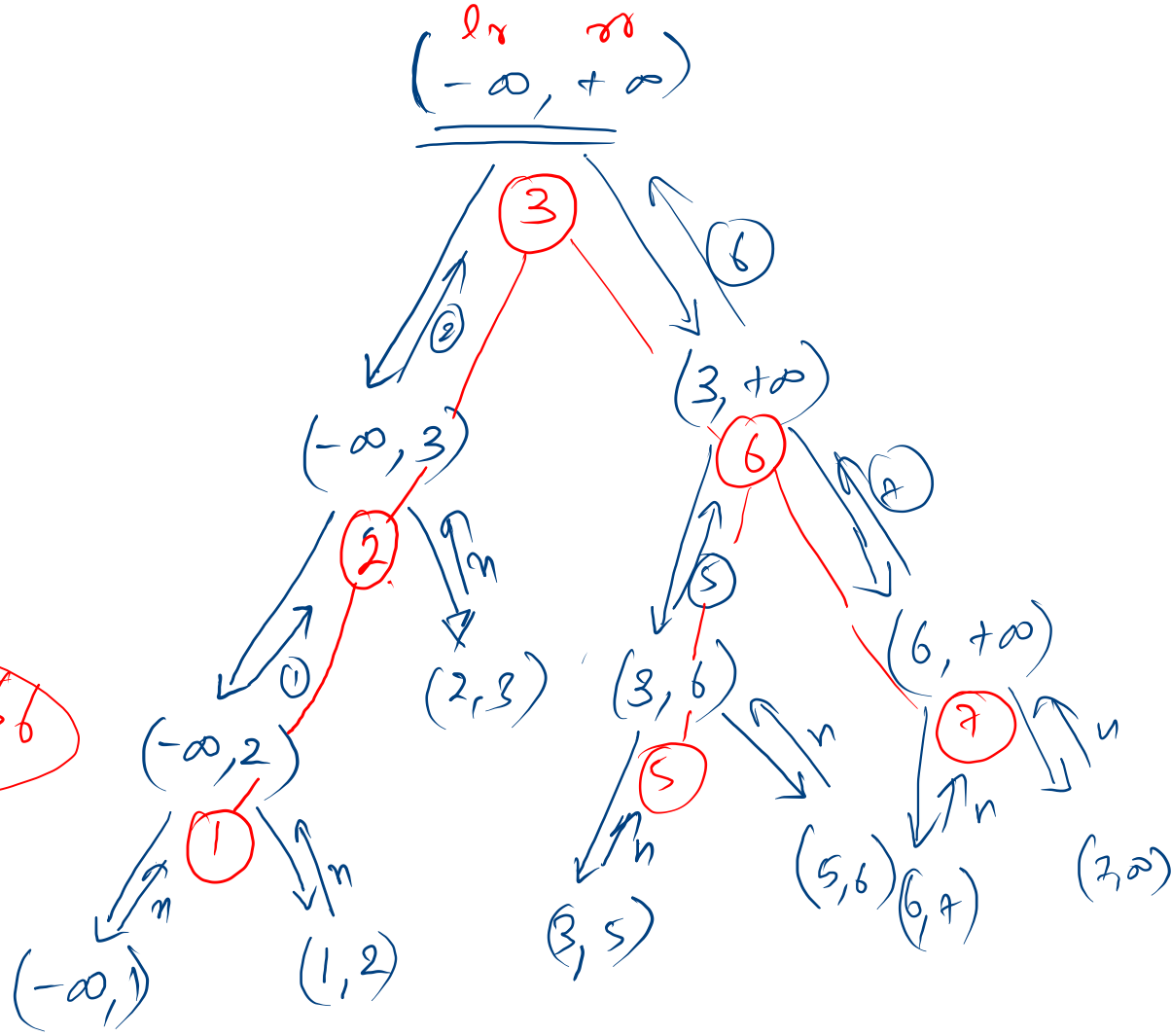
BST

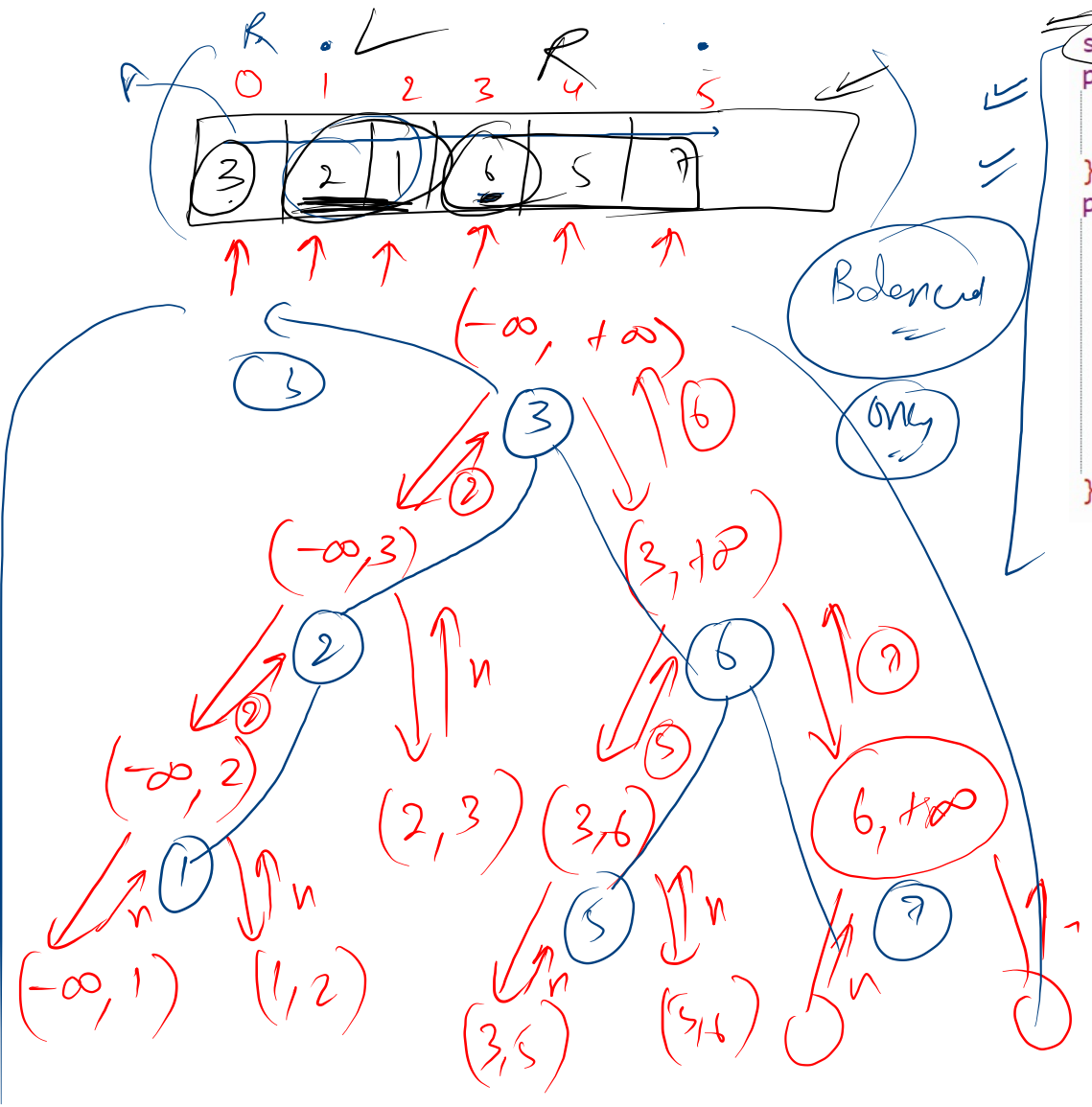
BST Prop



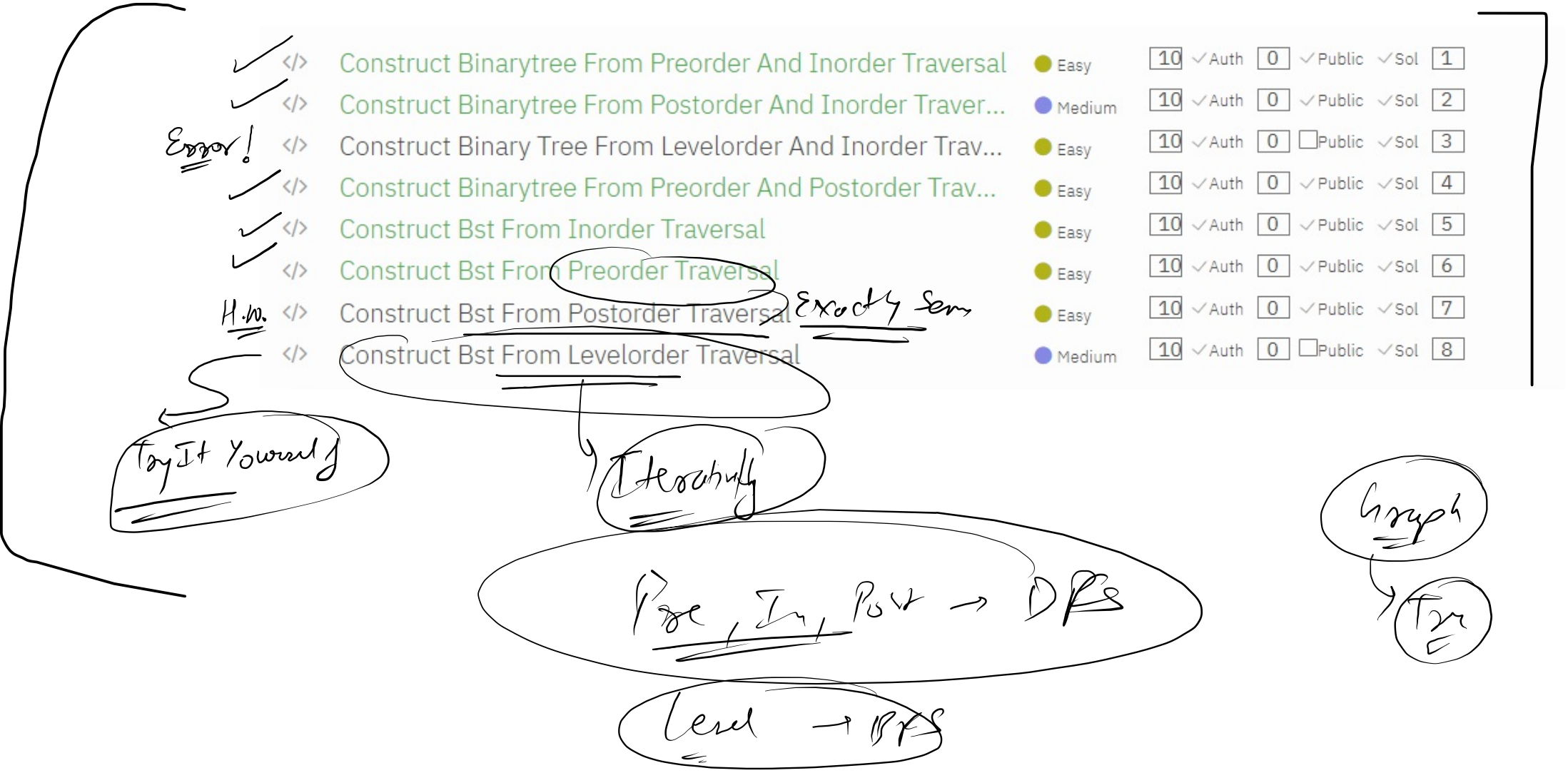
bst idx = ~~0 1 2 3 4 5~~ 6








```java
static int bstIdx;
public static TreeNode bstFromPreorder(int[] preorder) {
    bstIdx = 0;
    return bstFromPreorder(preorder,Integer.MIN_VALUE,Integer.MAX_VALUE);
}
public static TreeNode bstFromPreorder(int pre[],int lr,int rr){
    if(bstIdx >= pre.length || pre[bstIdx] < lr || pre[bstIdx] > rr){
        return null;
    }

    TreeNode node = new TreeNode(pre[bstIdx++]);
    node.left = bstFromPreorder(pre,lr,node.val);
    node.right = bstFromPreorder(pre,node.val,rr);
    return node;
}
```


Construct Binarytree From Preorder And Inorder Traversal Easy 10 Auth 0 Public Sol 1
Construct Binarytree From Postorder And Inorder Traver... Medium 10 Auth 0 Public Sol 2
Error!
Construct Binary Tree From Levelorder And Inorder Trav... Easy 10 Auth 0 Public Sol 3
Construct Binarytree From Preorder And Postorder Trav... Easy 10 Auth 0 Public Sol 4
Construct Bst From Inorder Traversal Easy 10 Auth 0 Public Sol 5
Construct Bst From Preorder Traversal Easy 10 Auth 0 Public Sol 6
H.W.
Construct Bst From Postorder Traversal
Exactly Same
Easy 10 Auth 0 Public Sol 7
Construct Bst From Levelorder Traversal Medium 10 Auth 0 Public Sol 8
Try It Yourself
Iterchudy
Pre, In, Post → DFS
Level → BFS
Graph
Tree