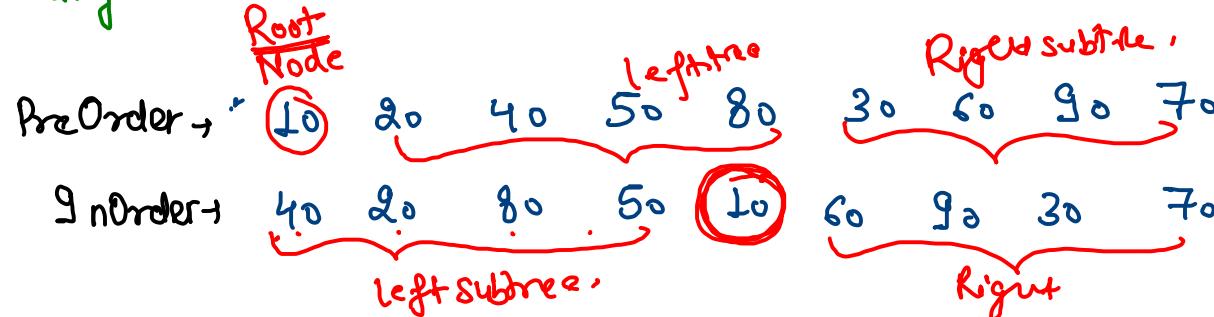
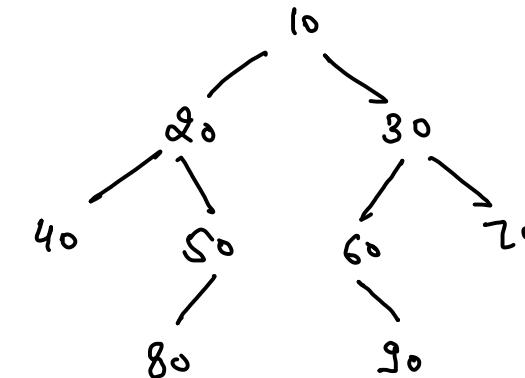


Binary Tree Construction:



PreOrder → Root Node Left Subtree Right Subtree

InOrder → leftSubtree Root Node Right Subtree



PreOrder
InOrder }
PostOrder
InOrder }

single
possible
Tree?

The diagram shows the construction of a binary search tree (BST) from given PreOrder and InOrder traversal results.

PreOrder → 10 20 40 50 80 30 60 90 70

InOrder → 40 20 80 50 10 60 90 20 70

PreOrder → Root Left Right

InOrder → Left Root Right

The tree structure is as follows:

```

    graph TD
        Root[10] --> L1[20]
        Root --> R1[30]
        L1 --> L2[40]
        L1 --> R2[60]
        R1 --> L3[50]
        R1 --> R3[70]
        L2 --> L4[80]
        L2 --> R4[90]
    
```

Annotations indicate the traversal types:
 - The first 5 nodes (10, 20, 40, 50, 80) are grouped by a bracket labeled "L.S." (Left Subtree).
 - The next 3 nodes (30, 60, 90) are grouped by a bracket labeled "R.S." (Right Subtree).
 - The last node (70) is labeled "R.R." (Right Right).
 - The root node (10) is circled in blue.
 - The left child of 10 (20) is circled in blue.
 - The right child of 10 (30) is circled in blue.
 - The left child of 20 (40) is circled in blue.
 - The right child of 20 (60) is circled in blue.
 - The left child of 30 (50) is circled in blue.
 - The right child of 30 (70) is circled in blue.
 - The left child of 40 (80) is circled in blue.
 - The right child of 40 (90) is circled in blue.

PreOrder → Root Left Right

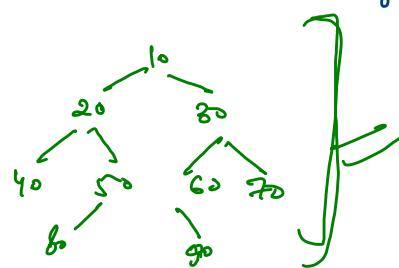
InOrder → Left Root Right

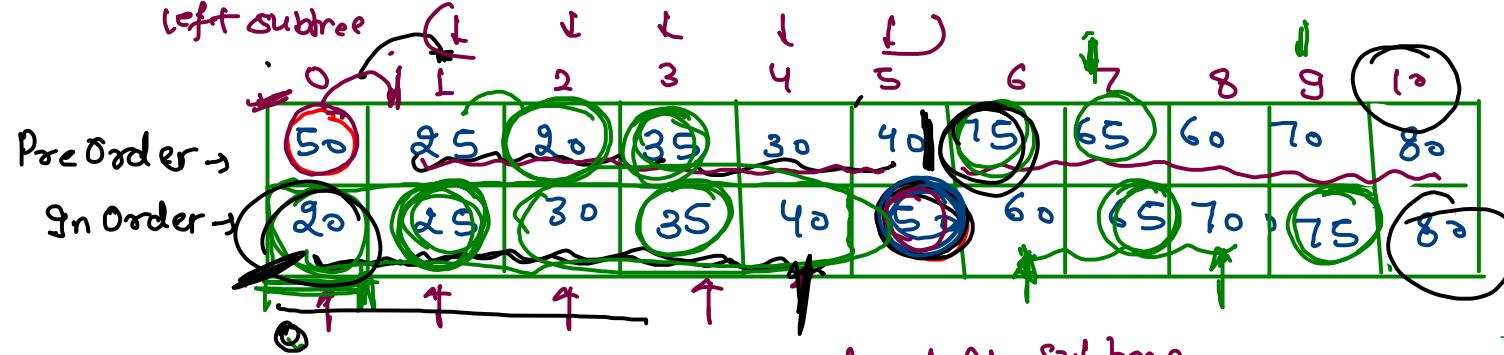
PreOrder -
InOrder -

A hand-drawn diagram illustrating energy flow in a food chain. The diagram shows three levels of producers (green circles) and three levels of consumers (blue circles).

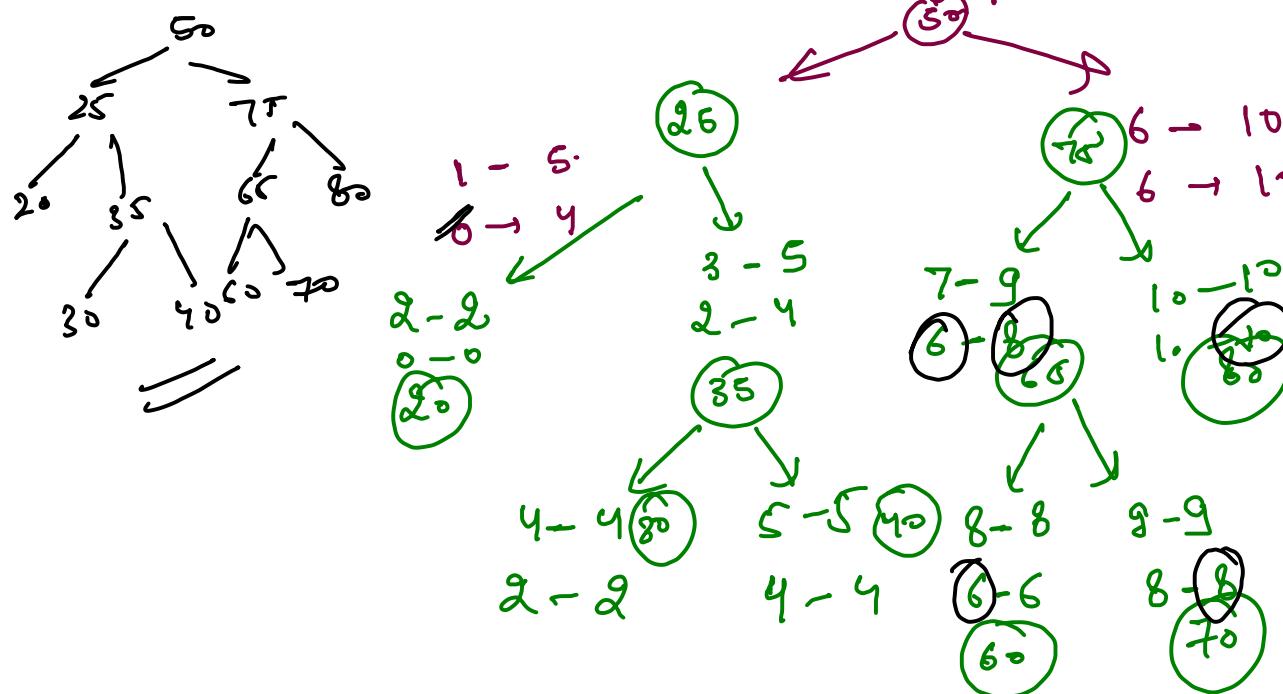
- Producers (Level 1):** Three green circles labeled "40".
- Primary Consumers (Level 2):** Three blue circles labeled "20".
- Secondary Consumers (Level 3):** One blue circle labeled "20".
- Tertiary Consumers (Level 4):** One blue circle labeled "20".
- Quaternary Consumers (Level 5):** One blue circle labeled "20".

Arrows indicate energy flow from one level to the next. Labels "fr" and "gn" are present near the first two levels of producers.





in-ele = 5 + element - left subtree



Pre-Starting Index

Pre-Ending Index

In Starting Index

In Ending Index

Return → Node,

Step 8'

- ① Make Root Node
- ② Find data Index in InOrder.
- ③ Split calls in left side, right side.

```

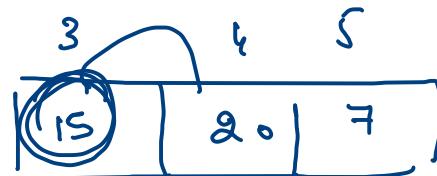
// create root node
Treenode root = new TreeNode(pre[pst]);
root.left = construct(pre, in, pst + 1 ? ist, ?);
root.right = construct(pre, in, ?, pend, ?, iend);

```

Root left right

pend - 3 → (pst + no. of ele)
gnd × 2. di - 1

pst + no. of ele
pst + 2.
gn. start + diff



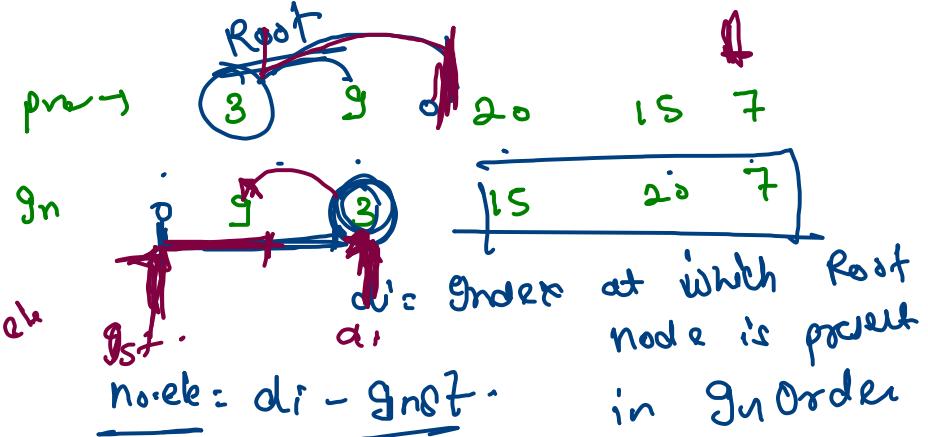
while (gnorder[di] != data) {

di = gnst

di = 3 4

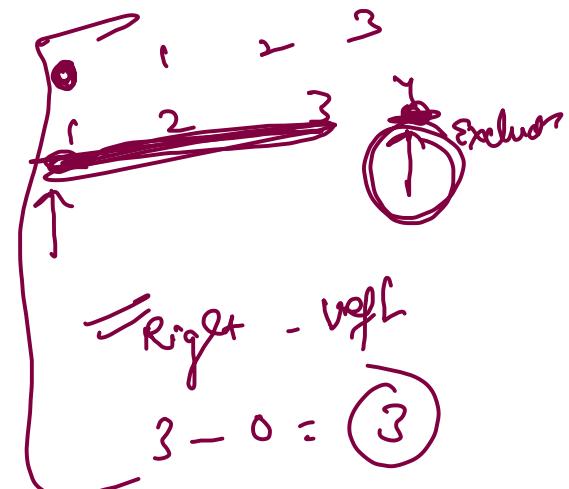
diff;

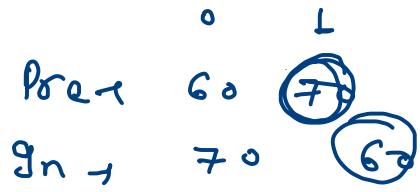
di' = 4 =



count =

di - gnst





~~$\text{fst} = 0$~~ ~~$\text{gnst} = 0$~~
 $\text{pend} = 1$ $\text{gnchd} = 1$

~~$\text{fst} > \text{gend} \parallel \text{gnst} > \text{gend}$~~
~~not for null~~

Root =
 $\text{di} = 1$
 $\text{nEle} = 1$
 $\text{fst} = 1$ $\text{gnst} = 0$
 $\text{pend} = 1$ $\text{gEnd} = 0$
 7°

$\text{fst} = 2$ $\text{gnst} = 2$
 $\text{pend} = 1$ $\text{gend} = 1$

PostOrder →	2	8	3	10	4	9	7	5
inOrder →	2	3	8	5	10	4	7	9

0 1 2 3 4 5 6 7

PostOrder → Left Right Root
 inOrder → left Node right

```

if(pst > pend || ist > iend)
    return null;

if(pst == pend || ist == iend) {
    return new TreeNode(post[pst]);
}

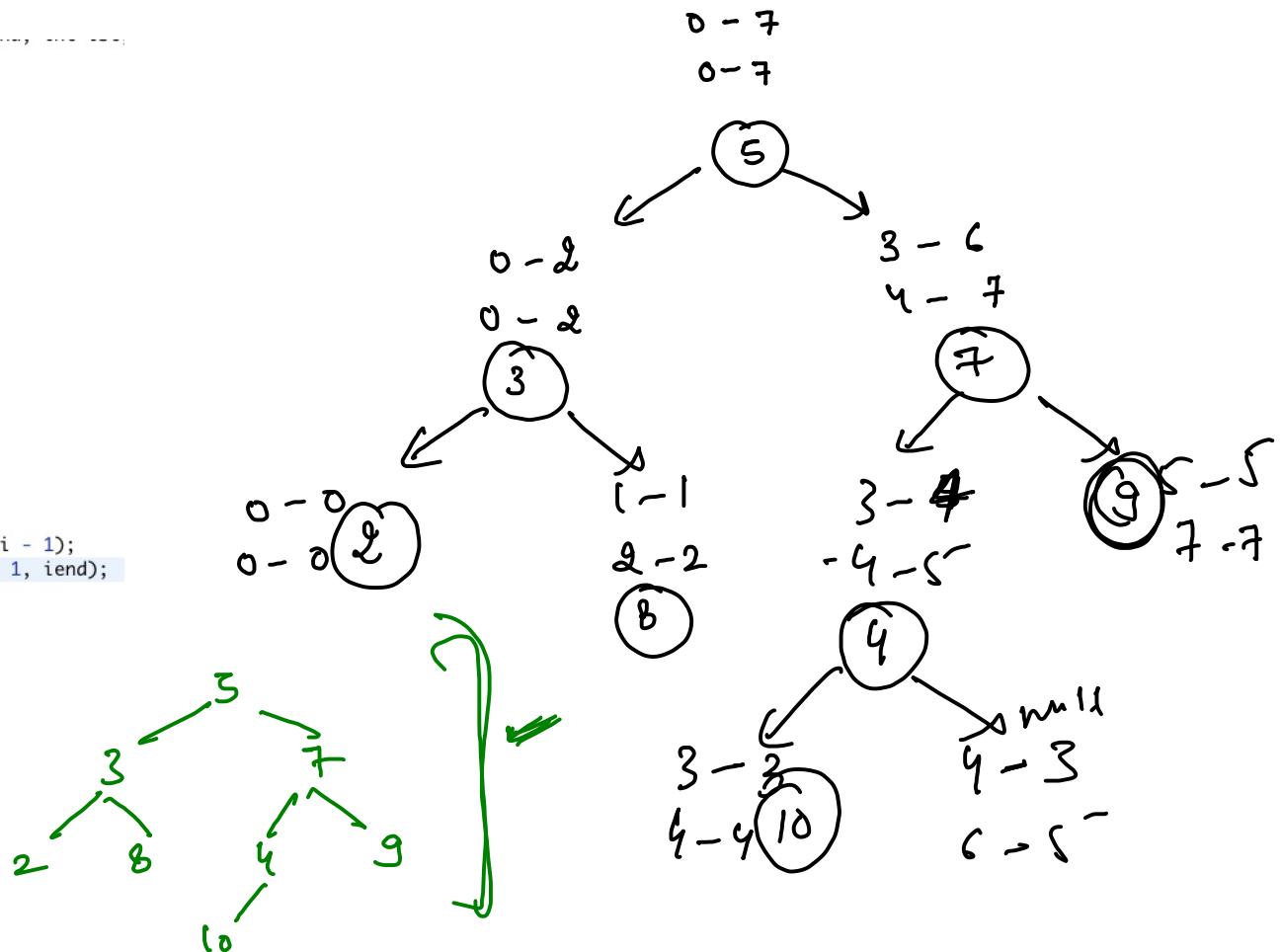
// create root node
TreeNode root = new TreeNode(post[pend]);

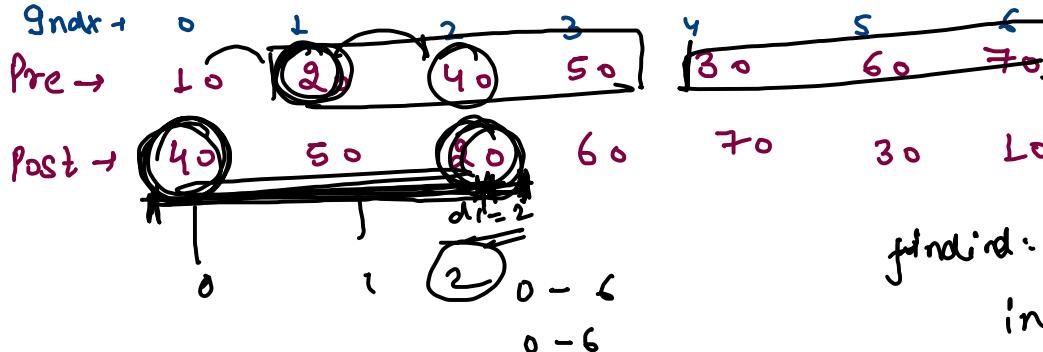
// di?
int di = ist;

while(in[di] != post[pend]) {
    di++;
}

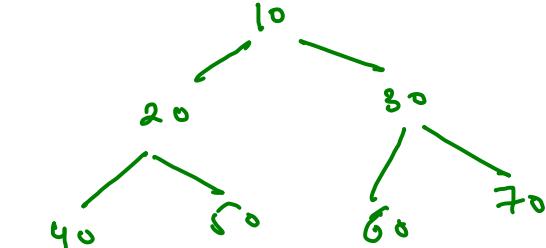
// no. of ele?
int nEle = di - ist;

root.left = construct(post, in, pst, pst + nEle - 1, ist, di - 1);
root.right = construct(post, in, pst + nEle, pend - 1, di + 1, iend);
return root;
    
```



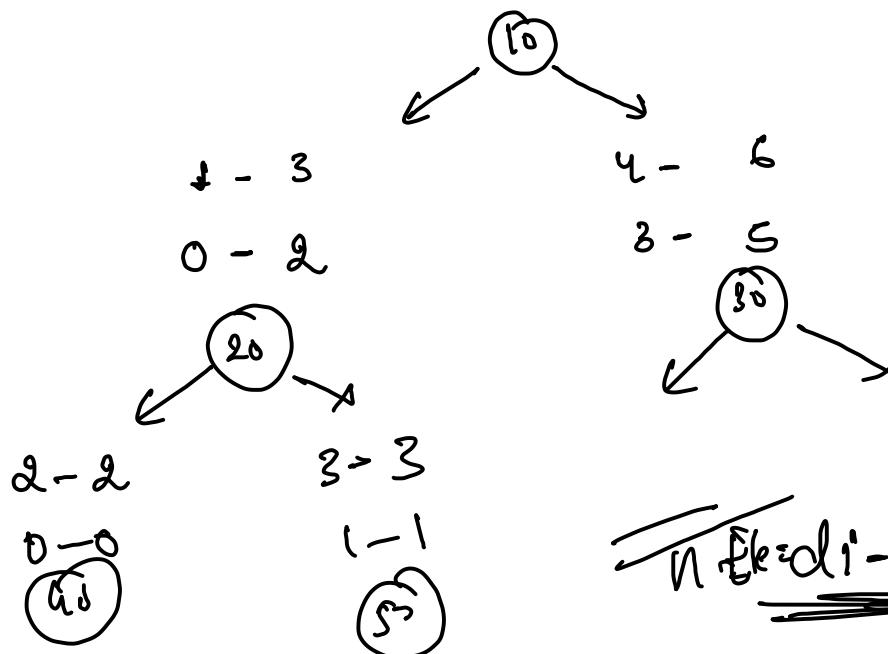


findind: $Pre[st+1]$
in post order.



, call →

left = $pre \rightarrow$
 $pre[st + 1] \leftarrow post + nEle$

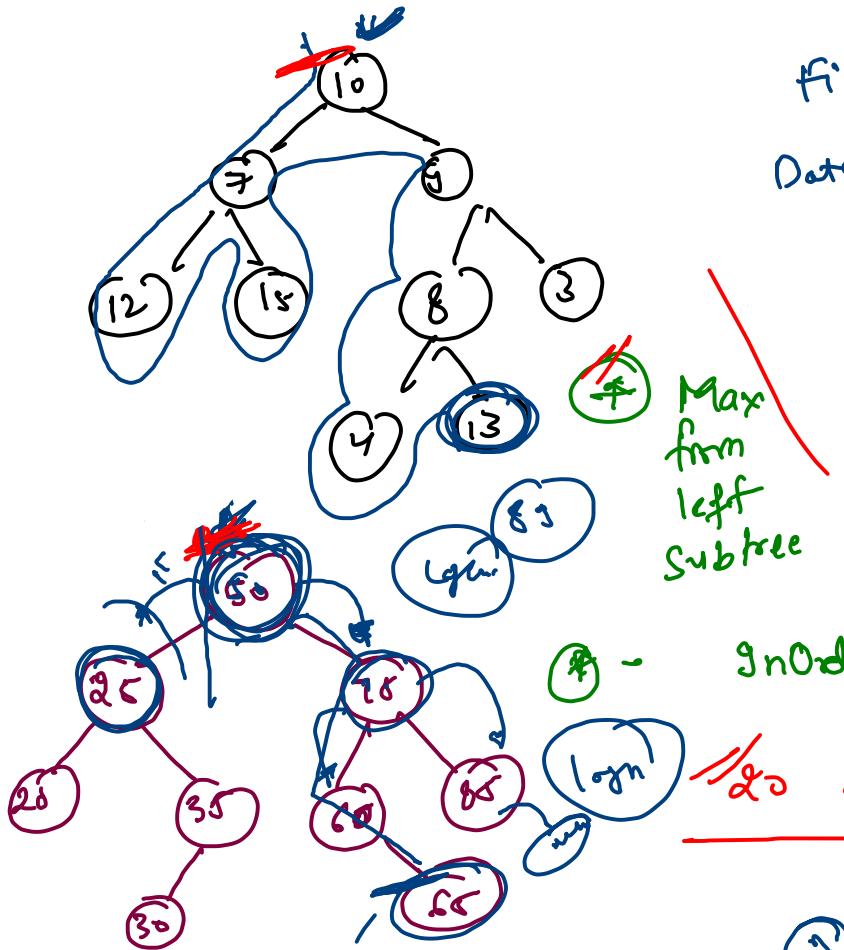


~~$nEle = dr - post + 1$~~

Right Pre →
 $post + nEle \leftarrow postEnd$

~~$post + 1$~~
 $dr + 1 \leftarrow postEnd - 1$

Binary Search Tree.



→ Data = 13

find in B-Tree = $O(n)$

Data → Set & Get

Month, Name
Arrange in first
Print it in Order

Set of Rules for Binary Search Tree.

↑ Root
↓ Node

Min from
Right Subtree



Feb.

inOrder of BST → Sorted Order.

20 25 30 35 40 50 55 60 65 75 80 85

Sorted Order =

```

public static BSTPair isBST2(Node node) {
    if(node == null) {
        return new BSTPair();
    }

    BSTPair lp = isBST2(node.left);
    if(lp.isBST == false) return lp;

    BSTPair rp = isBST2(node.right);
    if(rp.isBST == false) return rp;

    BSTPair mp = new BSTPair();
    mp.isBST = lp.max < node.data && rp.min > node.data;
    mp.min = Math.min(node.data, Math.min(lp.min, rp.min));
    mp.max = Math.max(node.data, Math.max(lp.max, rp.max));

    return mp;
}

```

