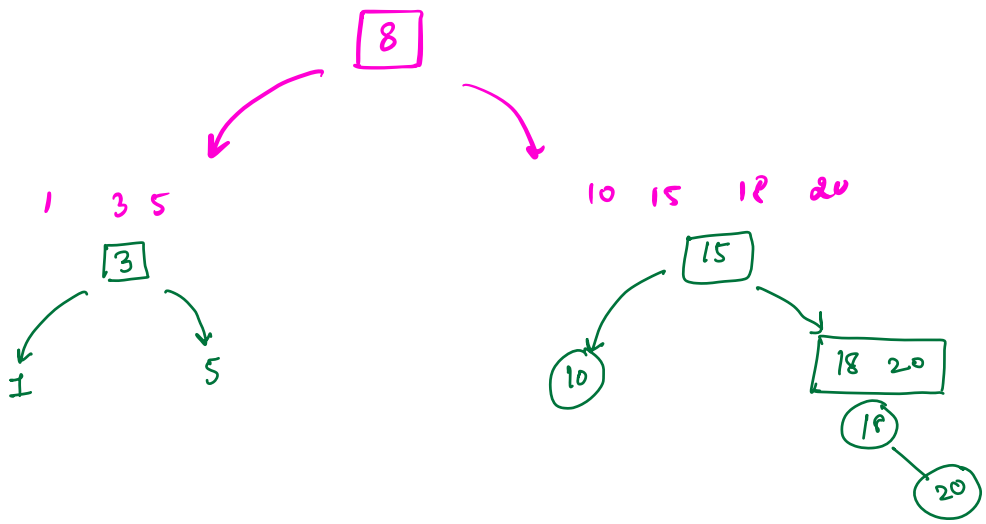
- sorted array ⇒ Balanced BST H = logN

$\forall$ node $\left|\text{height of LST} - \text{height of RST}\right| <= 1$

1 3 5 8 10 15 18 20

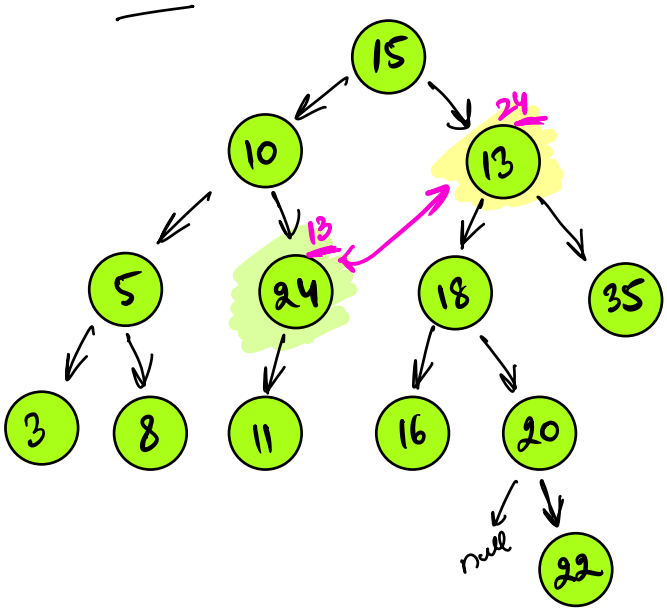



```
Node build ( int arr[], int start, int end)
{
    if ( start > end) return null;
    int mid = (start + end)/2;
    Node temp = new Node( arr[mid]);

    temp.left = build( arr, start, mid-1);
    temp.right = build ( arr, mid+1, end);
    return temp;
}
```

T.C. = O(n)

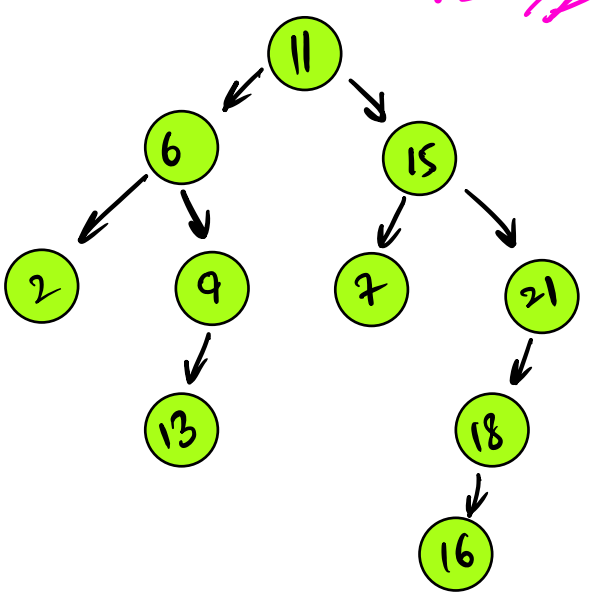
Restore BST

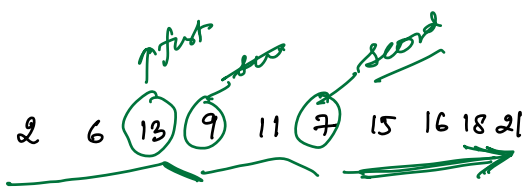


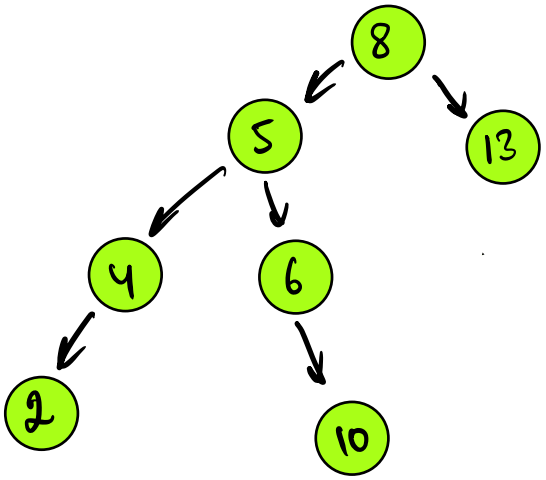
In a BST, two nodes were swapped. Find these 2 nodes if they are swapped again, it will become BST.

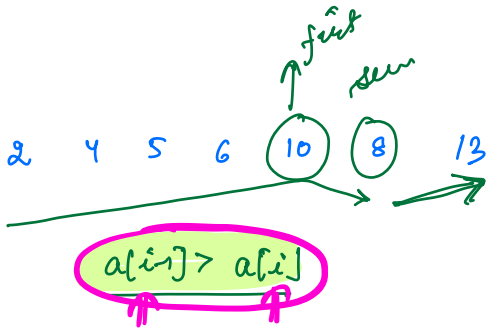
3 5 8 10 11 24 15 16 18 20 22 13 35

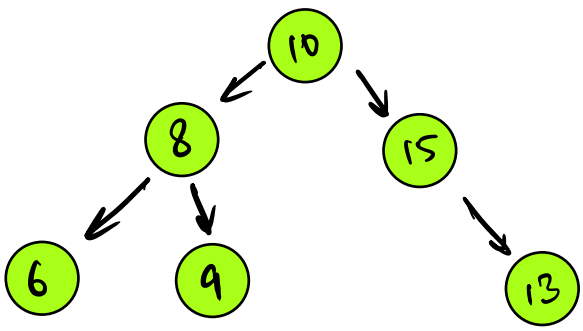
first second

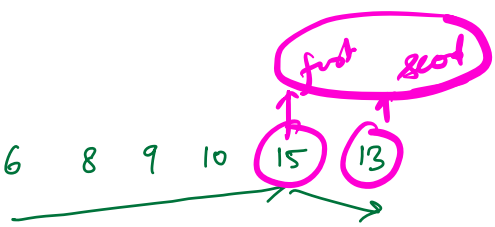






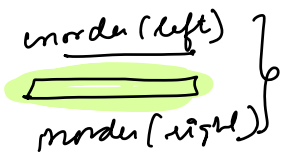




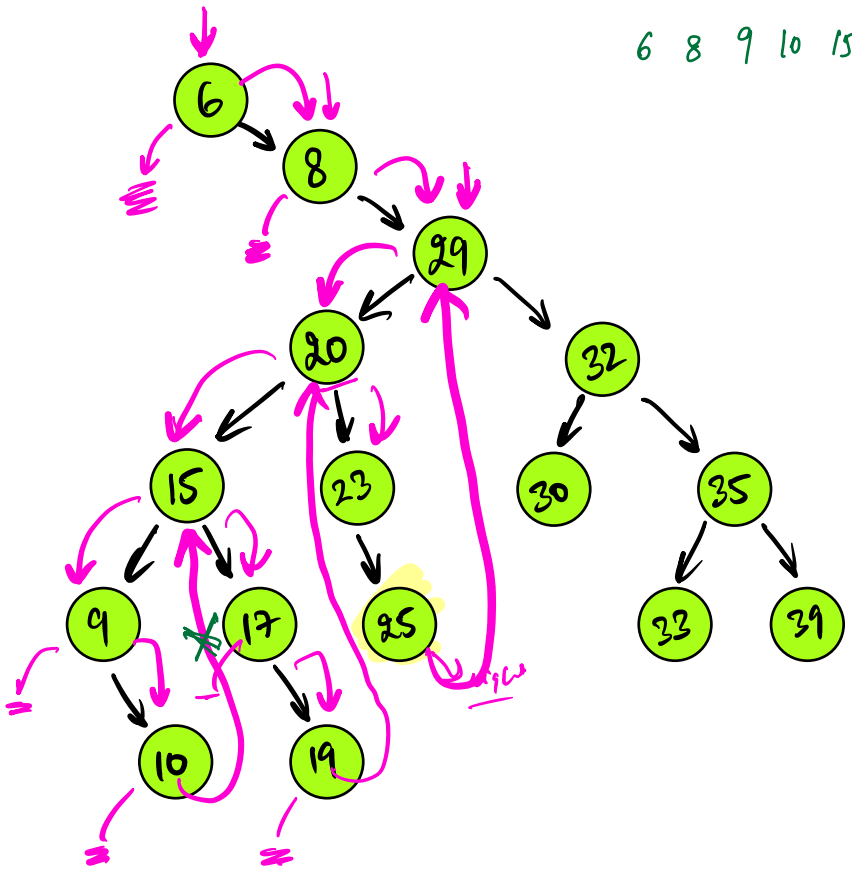


first = null, second = null, prev = null

```
void findswapped( Node root )
{
    if(root == null) return;
    findswapped( root.left);
    if( prev != null && prev.data > root.data)
    {   if( first == null) first = prev;
        second = root;
    } prev = root;
    findswapped ( root.right);
}
```



T.C: O(n)
S.C: O(H)
recursive stack

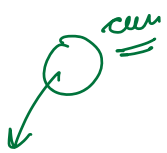


6 8 9 10 15 17 19 20

S.C : O(1)

Recursion ↑ X
iterative

25 is inorder predecessor of 29



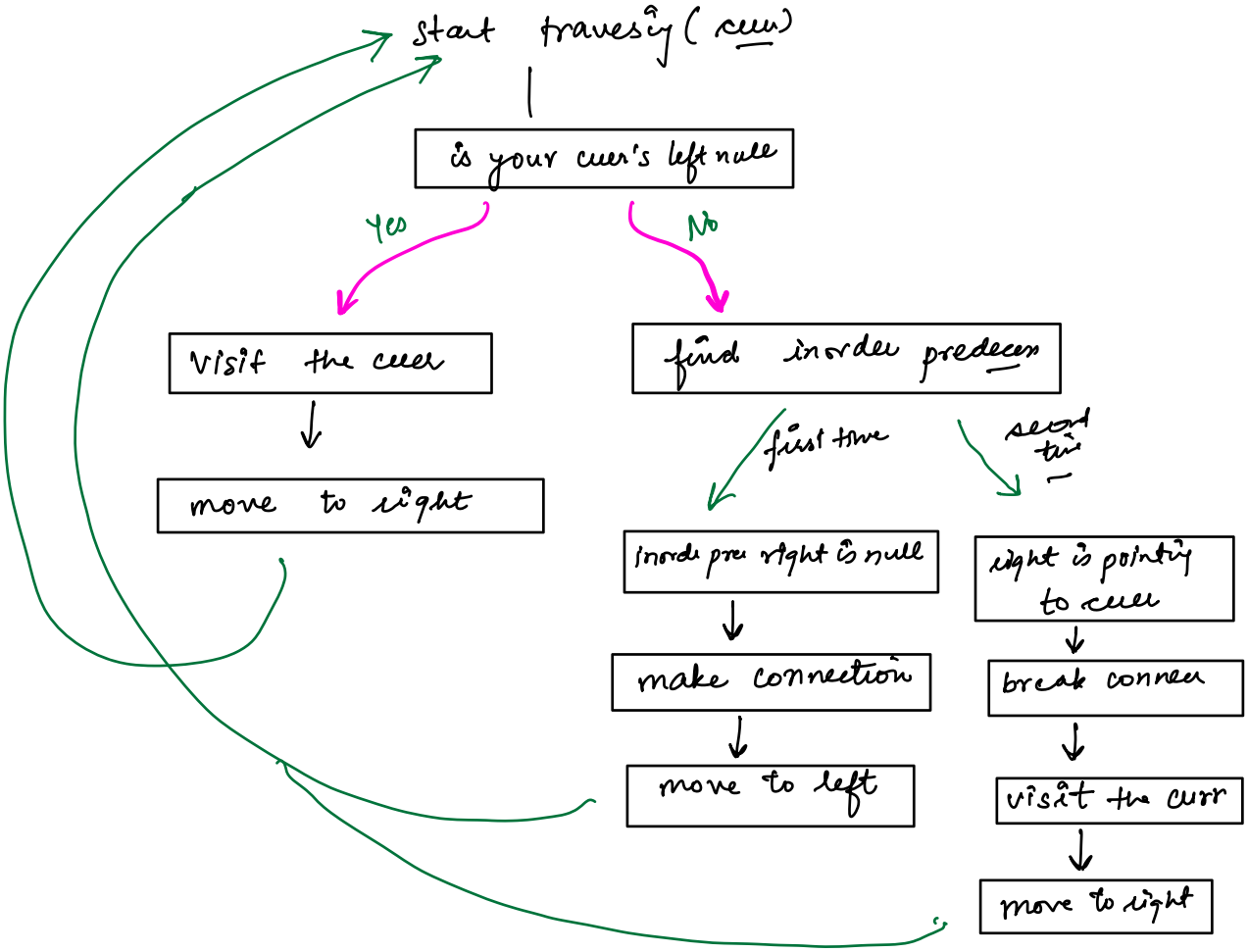

```
temp = curr.left;
while( temp.right != null && temp.right != curr)
{
    temp = temp.right;
}
```

Morris inorder Traversal

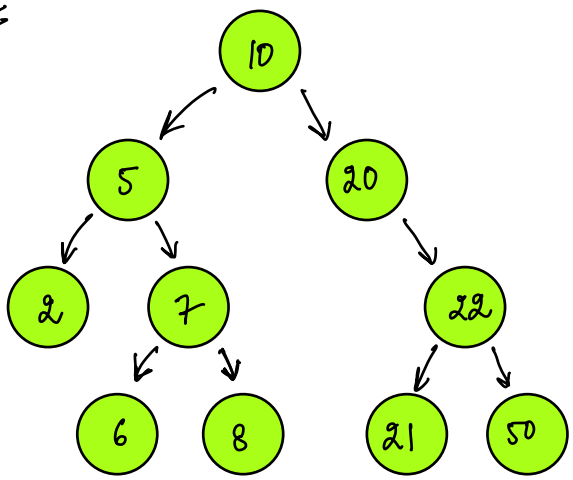
S.C: O(1)

T.C: O(3N) ≈ O(N)

```
curr = root;
while ( curr != null )
{
    if( curr.left == null)
    {
        print( curr.data);
        curr=curr.right;
    }
    else
    {
        temp = curr.left;                                   ①
        while ( temp.right != null && temp.right!=curr)     ②
        {
            temp= temp.right;
        }
        if( temp.right == null)
        {
            temp.right = curr;
            curr = curr.left;
        }
        else{
            temp.right =null;
            print( curr.data);
            curr = curr.right;
        }
    }
}
```


start traversing (curr)
is your curr's left null
Yes
No
visit the curr
move to right
find inorder predecessor
first time
second time
inorder pre right is null
make connection
move to left
right is pointing to curr
break connection
visit the curr
move to right

- BST



K^{th} smallest in BST

K	
3	6
5	8
10	50

inorder

0	1	2	3	4	5	6	7	8	9
2	5	6	7	8	10	20	21	22	50

inorder[k-1]

q queries → O(1)

queries

for only a single tree

↓

Moms