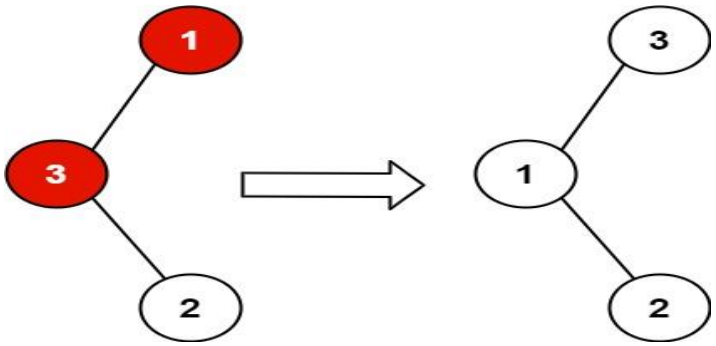
Binary Search Tree problems

1. Recover Binary Search Tree

You are given the root of a binary search tree (BST), where the values of **exactly** two nodes of the tree were swapped by mistake. *Recover the tree without changing its structure*.

Example 1:

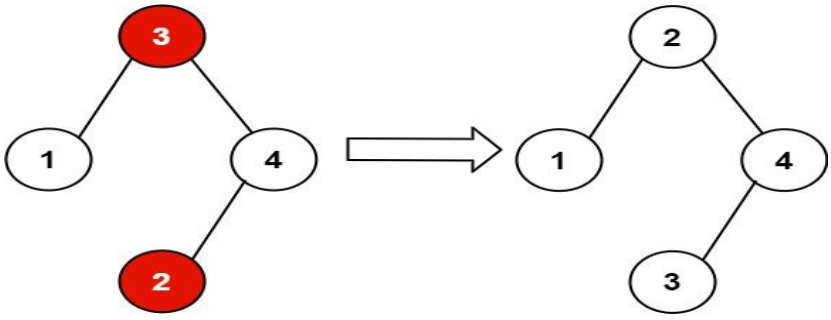


Input: root = [1,3,null,null,2]

Output: [3,1,null,null,2]

Explanation: 3 cannot be a left child of 1 because 3 > 1. Swapping 1 and 3 makes the BST valid.

Example 2:



Input: root = [3,1,4,null,null,2]

Output: [2,1,4,null,null,3]

Explanation: 2 cannot be in the right subtree of 3 because 2 < 3. Swapping 2 and 3 makes the BST valid.

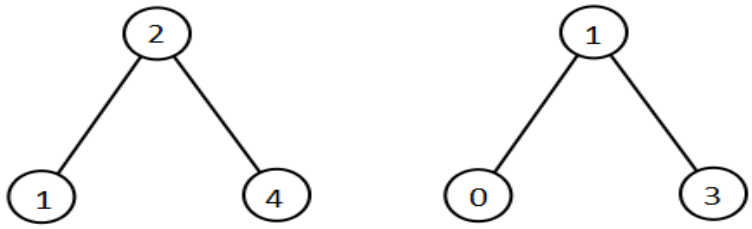
Constraints:

- The number of nodes in the tree is in the range [2, 1000].
- $-2^{31} <= Node.val <= 2^{31} - 1$

2. **All Elements in Two Binary Search Trees**

Given two binary search trees `root1` and `root2`, return *a list containing all the integers from both trees sorted in* ***ascending*** *order*.

Example 1:



Input: root1 = [2,1,4], root2 = [1,0,3]

Output: [0,1,1,2,3,4]

Example 2:

Input: root1 = [1,null,8], root2 = [8,1]

Output: [1,1,8,8]

Constraints:

- The number of nodes in each tree is in the range [0, 5000].
- -10^5 <= Node.val <= 10^5

3. **Find Leftmost and Rightmost nodes for a given node:**

Given a preorder sequence of the binary search tree of **N** nodes. The task is to find its leftmost and rightmost nodes.

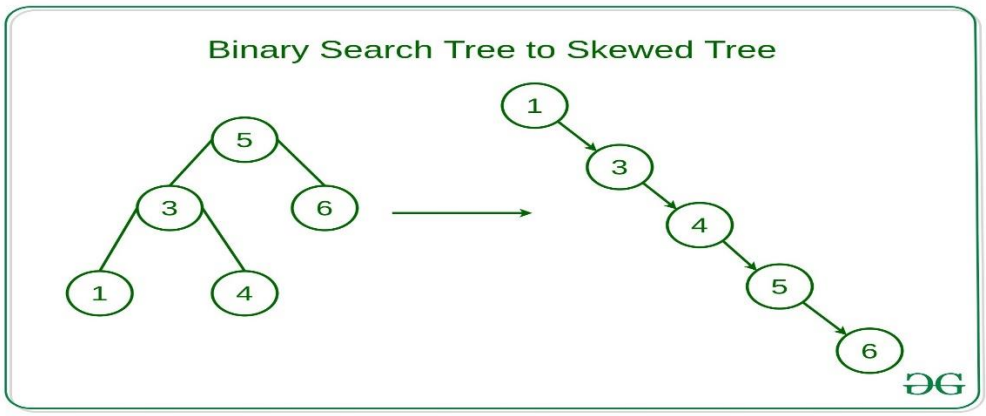
Examples:
Input : N = 5, preorder[]={ 3, 2, 1, 5, 4 }
Output : Leftmost = 1, Rightmost = 5
The BST constructed from this
preorder sequence would be:

```
    3
   / \
  2   5
 /   /
1   4
```
Leftmost Node of this tree is equal to 1
Rightmost Node of this tree is equal to 5

Input : N = 3 preorder[]={ 2, 1, 3}
Output : Leftmost = 1, Rightmost = 3

4. Convert BST into Skewed Tree:

Given a Binary Search Tree and a binary integer **K**, the task is to convert Binary search tree into a Skewed Tree in increasing order if **K = 0** or in decreasing order if **K = 1**.



Examples:

Input: K = 0,
```
    5
   / \
  3   6
```
Output:
```
  3
   \
    5
     \
      6
```

Input: K = 1,
```
    2
   / \
  1   3
```
Output:
```
  3
   \
    2
     \
      1
```

Reference links:

1. https://leetcode.com/problems/recover-binary-search-tree/
2. https://leetcode.com/problems/all-elements-in-two-binary-search-trees/
3. https://www.geeksforgeeks.org/find-leftmost-and-rightmost-node-of-bst-from-its-given-preorder-traversal/
4. https://www.geeksforgeeks.org/convert-a-binary-search-tree-into-a-skewed-tree-in-increasing-or-decreasing-order/