Problem 1305: All Elements in Two Binary Search Trees

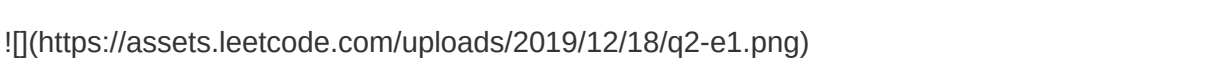
Problem Information

Difficulty: Medium
Acceptance Rate: 80.17%
Paid Only: No
Tags: Tree, Depth-First Search, Binary Search Tree, Sorting, Binary Tree

Problem Description

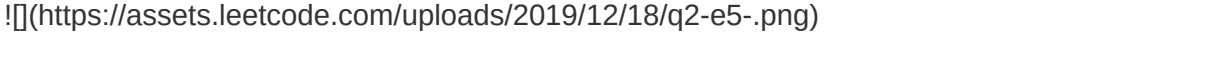
Given two binary search trees `root1` and `root2`, return _a list containing all the integers from both trees sorted in**ascending** order_.

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]`. * `-105 <= Node.val <= 105`

Code Snippets

C++:

```
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode() : val(0), left(nullptr), right(nullptr) {}
* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
* TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left),
right(right) {}
* };
*/
class Solution {
public:
vector<int> getAllElements(TreeNode* root1, TreeNode* root2) {

}
};
```

Java:

```
/**
* Definition for a binary tree node.
* public class TreeNode {
* int val;
* TreeNode left;
* TreeNode right;
* TreeNode() {}
* TreeNode(int val) { this.val = val; }
* TreeNode(int val, TreeNode left, TreeNode right) {
* this.val = val;
* this.left = left;
* this.right = right;
* }
* }
*/
class Solution {
public List<Integer> getAllElements(TreeNode root1, TreeNode root2) {

}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def getAllElements(self, root1: Optional[TreeNode], root2:
Optional[TreeNode]) -> List[int]:
```