Problem # 1305 : All Elements in Two Binary Search Trees

<https://leetcode.com/problems/all-elements-in-two-binary-search-trees/>

Solution:

<https://leetcode.com/problems/all-elements-in-two-binary-search-trees/discuss/830982/Simple-Python-3-Solution-in-O(n)-time>

Simple Python 3 Solution in O(n) time

1. Create the inorder traversal list for each tree. This is done in the inline function traverseInorder. Inorder traversal will give us a sorted list of node values.
2. Merge the two sorted lists using two pointers i and j for list1 and list2 respectively into a result list.
3. One of the lists may be longer than the other and so add the remaining elements in the longer list to the result list. Return the result list.

-------------------------

# 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: TreeNode, root2: TreeNode) -> List[int]:

def traverseInorder(root):

if not root:

return([])

left\_list = traverseInorder(root.left)

right\_list = traverseInorder(root.right)

return(left\_list + [root.val] + right\_list)

list1 = traverseInorder(root1)

list2 = traverseInorder(root2)

# Merge the two lists

result = []

i = 0

j = 0

while (i < len(list1) and j < len(list2)):

if list1[i] <= list2[j]:

result.append(list1[i])

i += 1

else:

result.append(list2[j])

j += 1

if (i < len(list1)):

result += list1[i:]

elif (j <len(list2)):

result += list2[j:]

return(result)

'''