All Elements in Two Binary Search Trees (Leetcode 1305)

Inorder Traversal gives sorted lists of tree nodes
Wherever we need to ~~do~~ merge two lists we can use Merge sort
In this question we will use Merge sort of two-sorted lists.

Question:- Return a list containing all the integers from both trees sorted in ascending order.

Approach:-
- In order traversal on both the trees to get sorted lists of nodes for each tree.
- Do merge sort on two-sorted lists which will be our final answer list.

Pseudocode:-

In order traversal

Recursively calling nodes from left to right.

→ if root is null stop it ~~& add it to it~~ right there.
→ if not, then go left now consider this as root node & check if is null or not
→ after done, reaching leftmost point ~~with everythg~~ on ~~left~~ tree add its parent then traverse towards its right child.

Merge-sort of sorted lists

→ two sorted can be of same / diff sizes
→ if same size no-issues their elements will be emptied and stored to final answer list simultaneously.
→ if of diff size. then one will end before the another.
→ ~~How~~ How elements will be stored in the final answer list?
we will run two pointers on each lists and the element which is lesser between two items from two differents lists will get stored to final answer list and pointer of that element ~~will be~~ that is stored will increase. and if the two lists are of diff size the list which remained will be stored as it is final answer list as ~~they are~~ their elements are already sorted.

Code:-

```
private void inorder(TreeNode root, List<Integer> list){
        if (root == null)
            return;
        inorder (root.left, list);
        list.add (root.val);
        inorder(root.right, list);
}
```

Main function

```
List<Integer> ans = new ArrayList<>();
List <Integer > a = new Arraylist<>();
 - " -       b =    - " -
   inorder (root1, a);  inorder(root2, b);
   mergeLists (a, b, ans);
```

```
private void mergelists
      (List<Integer> list1, List<Integer>
                               list2,
                      List<Integer> ans){
  int n = list1.size(),   i=0
  int m = list2 size ( ), j=0;
  while (i<n && j<m) {
      if(list1.get(i) <= list2.get(j))
             ans.add(list1.get(i++));
      else
         ans.add(list2.get(j++));
  }
  while(i<n)
       ans.add(list1.get(i++));
  while (j<m)  ans. add (list2.get(j++));
```