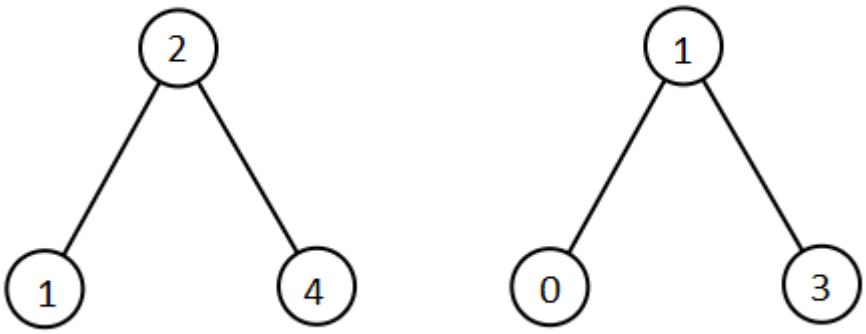
1305. All Elements in Two Binary Search Trees

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 = [0,-10,10], root2 = [5,1,7,0,2]
Output: [-10,0,0,1,2,5,7,10]

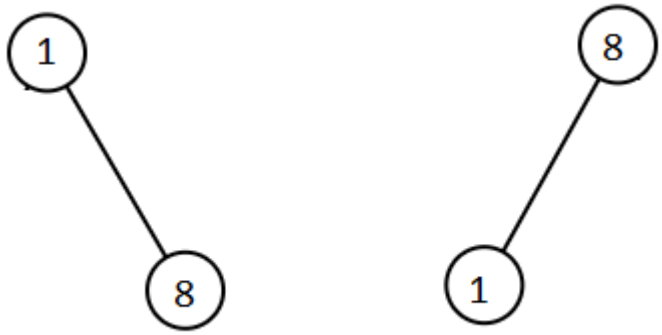
Example 3:

Input: root1 = [], root2 = [5,1,7,0,2]
Output: [0,1,2,5,7]

Example 4:

Input: root1 = [0,-10,10], root2 = []
Output: [-10,0,10]

Example 5:



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

```
class Solution:
    def getAllElements(self, root1: TreeNode, root2: TreeNode) ->
List[int]:
        tree1 = self.helper(root1,[])
        tree2 = self.helper(root2,[])
        if tree1 is None:
            n = 0
        else:
            n = len(tree1)
        if tree2 is None:
            m = 0
        else:
            m = len(tree2)
        # m = len(tree2)
        res = [0]*(n+m)
        i = 0
        j = 0
        k = 0
        while k<len(res) and i<n and j<m:
            if tree1[i]<=tree2[j]:
                res[k] = tree1[i]
                i = i+1
            elif tree1[i]>tree2[j]:
                res[k] = tree2[j]
                j = j+1
            k = k+1
        while i<n:
            res[k] = tree1[i]
            k = k+1
            i = i+1
        while j<m:
```

```
            res[k] = tree2[j]
            k = k+1
            j = j+1
        return res



    def helper(self,root,elements):
        if root is None:
            return
        self.helper(root.left,elements)
        elements.append(root.val)
        self.helper(root.right,elements)
        return elements
```