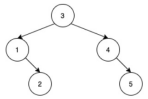
BST SEQUENCES -> we are given the tree



Input: nums = [3,4,5,1,2]
Output: 5
Explanation: The following 5 arrays will yield the same BST:
[3,1,2,4,5]
[3,1,4,2,5]
[3,1,4,5,2]
[3,4,1,2,5]
[3,4,1,5,2]

← Just for reference

Obviously the root will be at first.

-> Just think -> you get all the possible ans from left and then Right, how are we going to merge them.

In this case there will be single array from both sides.

left -> [{1,2}], Right = [{4,5}]
order should be same — order should be same

{1,2} {4,5} { }

{2} {4,5} {1} {1,2} {5} {4}

{ } {4,5} {1,2} {2} {5} {1,4} {2} {5} {4,5}

Similarly there will be another recursive function which will just start building from bottom.

```
public static List<List<Integer>> allSequence(TreeNode node) {
    List<List<Integer>> result = new ArrayList<>();

    if (node == null) {
        result.add(new LinkedList<Integer>());

        return result;
    }

    List<Integer> prefix = new LinkedList<>();

    prefix.add(node.data);

    List<List<Integer>> leftSeq = allSequence(node.left);
    List<List<Integer>> rightSeq = allSequence(node.right);

    for (List<Integer> left : leftSeq) {
        for (List<Integer> right : rightSeq) {
            List<List<Integer>> merged = new ArrayList<>();
            merge(left, right, merged, prefix);
            result.addAll(merged);
        }
    }

    return result;
}
```