**[Question 1] Transform a Binary Tree to a Binary Search Tree**

You are to convert an arbitrary binary tree to a binary search tree with the same structure and the same set of keys. Let the binary tree be T, your task is to construct a binary search tree T' such that for any pair of node n and n' at the same position (same path from the root) in T and T' respectively, the number of nodes in the left subtree of n is the same as that in the left subtree of n', and the number of nodes in the right subtree of n is also the same as that in the right subtree of n'. In addition, binary tree T' satisfies the property of binary search trees.

Implement a function to transform an arbitrary binary tree to a binary search tree. Test inputs begin with the number of lines below. Each line describes a binary tree by a list of tree nodes with key at itself, its left child and its right child, separated by ‘:’. Letter x represents a null node, e.g., a tree node with two x is a leaf node. The sequence of tree nodes is arbitrary. For example, ‘0:x:x -1:1:-2 -2:0:x 1:x:x’ represents a 4-node binary tree rooted at -1, the left child is 1 and the right child is -2, where node 1 is a leaf node. For each binary tree, output the binary search tree in pre-order traversal separated by space, ‘-1 -2 1 0’ for the above example.

Sample code is provided in the file A2Q1.py. You need to modify the function to\_BST. The maximum of tree height is 100 and the maximum number of nodes in a binary tree is 1,000,000. Keys are integers between -231 and 231-1, and they are distinct.

**[Question 1] Thoughts and Approach**