```
package Love_Babbar;

public class BST {
    class BinarySearchTree {
        // Find a value in a BST
        // Deletion of a node in a BST
        // Find min and max value in a BST
        // Find inorder successor and inorder predecessor in a BST
        // Check if a tree is a BST or not
        // Populate Inorder successor of all nodes
        // Find LCA of 2 nodes in a BST
        // Construct BST from preorder traversal
        // Convert Binary tree into BST
        // Convert a normal BST into a Balanced BST
        // Merge two BST [ V.V.V>IMP ]
        // Find Kth largest element in a BST
        // Find Kth smallest element in a BST
        // Count pairs from 2 BST whose sum is equal to given value “X”
        // Find the median of BST in O(n) time and O(1) space
        // Count BST nodes that lie in a given range
        // Replace every element with the least greater element on its right
        // Given “n” appointments, find the conflicting appointments
        // Check preorder is valid or not
        // Check whether BST contains Dead end
        // Largest BST in a Binary Tree [ V.V.V.V.V IMP ]
        // Flatten BST to sorted list
    }

}
```