9) CONVERTING THE BINARY TREE INTO BINARY SEARCH TREE (BST).

NOTES:
LINK OF EXPLANATION:
https://www.youtube.com/watch?v=8AnntMKIWlQ

TRICK:
The Main crux of the question is to take the array of size N and copy the inorder traversal of the binary tree into that array.
Then sort the array which takes the complexity of (N log N).
Then we again start inorder traversal of the tree and copy the sorted array elements into the tree.

MAIN FUNCTIONS:

```
class Solution{
  public:
    // The given root is the root of the Binary Tree
    // Return the root of the generated BST
    void InOrderTraversal(Node *root,vector<int> &v){
        if(root!=NULL){
            InOrderTraversal(root->left,v);
            v.push_back(root->data);
            InOrderTraversal(root->right,v);
        }
    }
    void BST(Node *root,vector<int> &v,int &index){
        if(root!=NULL){
            BST(root->left,v,index);
            root->data=v[index];
            index++;
            BST(root->right,v,index);
        }
    }
    Node *binaryTreeToBST (Node *root)
    {
        vector<int> v;
        InOrderTraversal(root,v);
        sort(v.begin(),v.end());
        int index=0;
        BST(root,v,index);
        return root;
    }
};
```