

Binary Tree to BST



Easy Accuracy: 50.0% Submissions: 31715 Points: 2

Given a Binary Tree, convert it to Binary Search Tree in such a way that keeps the original structure of Binary Tree intact.

Example 1:

Input:

```
    1
   / \
  2   3
Output: 1 2 3
```

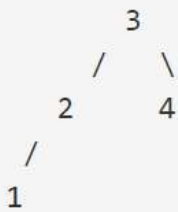
Example 2:

Input:

```
      1
     / \
    2   3
   /
  4
Output: 1 2 3 4
```

Explanation:

The converted BST will be



Your Task:

You don't need to read input or print anything. Your task is to complete the function **binaryTreeToBST()** which takes the root of the Binary tree as input and returns the root of the BST. The driver code will print **inorder** traversal of the converted BST.

Expected Time Complexity: $O(N \log N)$.

Expected Auxiliary Space: $O(N)$.

Constraints:

$1 \leq \text{Number of nodes} \leq 1000$

[View Bookmarked Problems](#)

Company Tags

☐ Adobe ☐ Amazon

Topic Tags

```
package BST;

import java.util.Arrays;

public class BinaryTreeToBST {

    class Node {

        int data;

        Node left, right;

        Node(int item) {

            data = item;

            left = right = null;

        }

    }

    int index;

    Node binaryTreeToBST(Node root) {

        // Your code here
        int n = getLength(root);
```

```

    int arr[] = new int[n];

    index = 0;
    getInorder(root, arr);

    index = 0;

    Arrays.sort(arr);

    BT_to_BST(root, arr);

    return root;
}
void BT_to_BST(Node root, int arr[]) {

    if (root == null) {

        return;
    }
    BT_to_BST(root.left, arr);

    root.data = arr[index++];

    BT_to_BST(root.right, arr);
}
void getInorder(Node root, int[] arr) {

    if (root == null) {

        return;
    }
    getInorder(root.left, arr);

    arr[index++] = root.data;

    getInorder(root.right, arr);
}

int getLength(Node root) {

    if (root == null){

        return 0;
    }
    return 1 + getLength(root.left) + getLength(root.right);
}
}

```