```
1   // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3
4   public class Program {
5     // Average case: when the tree is balanced
6     // O(nlog(n)) time | O(h) space - where n is the number of nodes in
7     // the Binary Tree and h is the height of the Binary Tree
8     public static int AllKindsOfNodeDepths(BinaryTree root) {
9       if (root == null) return 0;
10      return AllKindsOfNodeDepths(root.left) + AllKindsOfNodeDepths(root.right) +
11             nodeDepths(root, 0);
12    }
13
14    public static int nodeDepths(BinaryTree node, int depth) {
15      if (node == null) return 0;
16      return depth + nodeDepths(node.left, depth + 1) + nodeDepths(node.right, depth + 1);
17    }
18
19    public class BinaryTree {
20      public int value;
21      public BinaryTree left;
22      public BinaryTree right;
23
24      public BinaryTree(int value) {
25        this.value = value;
26        left = null;
27        right = null;
28      }
29    }
30  }
31
```