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