```
1   // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3   using System.Collections.Generic;
4
5   public class Program {
6     // Average case: when the tree is balanced
7     // O(n) time | O(h) space - where n is the number of nodes in
8     // the Binary Tree and h is the height of the Binary Tree
9     public static int NodeDepths(BinaryTree root) {
10      int sumOfDepths = 0;
11      Stack<Level> stack = new Stack<Level>();
12      stack.Push(new Level(root, 0));
13      while (stack.Count > 0) {
14        Level top = stack.Pop();
15
16        BinaryTree node = top.root;
17        int depth = top.depth;
18        if (node == null) continue;
19
20        sumOfDepths += depth;
21        stack.Push(new Level(node.left, depth + 1));
22        stack.Push(new Level(node.right, depth + 1));
23      }
24      return sumOfDepths;
25    }
26
27    public class Level {
28      public BinaryTree root;
29      public int depth;
30
31      public Level(BinaryTree root, int depth) {
32        this.root = root;
33        this.depth = depth;
34      }
35    }
36
37    public class BinaryTree {
38      public int value;
39      public BinaryTree left;
40      public BinaryTree right;
41
42      public BinaryTree(int value) {
43        this.value = value;
44        left = null;
45        right = null;
46      }
47    }
48  }
49
```