

Prompt	Scratchpad	Our Solution(s)	Video Explanation	Run Code
--------	------------	-----------------	-------------------	----------

Solution 1	Solution 2	Solution 3	Solution 4
<pre>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  ▾ def allKindsOfNodeDepths(root): 7    ▾     if root is None: 8         return 0 9         return allKindsOfNodeDepths(root.left) + allKindsOfNodeDepths(root.right) + nodeDepths(root) 10 11 12  ▾ def nodeDepths(node, depth=0): 13    ▾     if node is None: 14         return 0 15         return depth + nodeDepths(node.left, depth + 1) + nodeDepths(node.right, depth + 1) 16 17 18  # This is the class of the input binary tree. 19  ▾ class BinaryTree: 20    ▾     def __init__(self, value): 21         self.value = value 22         self.left = None 23         self.right = None 24</pre>			

