```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

# Average case: when the tree is balanced
# O(n) time | O(h) space - where n is the number of nodes in
# the Binary Tree and h is the height of the Binary Tree
def allKindsOfNodeDepths(root):
    return getTreeInfo(root).sumOfAllDepths


def getTreeInfo(tree):
    if tree is None:
        return TreeInfo(0, 0, 0)

    leftTreeInfo = getTreeInfo(tree.left)
    rightTreeInfo = getTreeInfo(tree.right)

    sumOfLeftDepths = leftTreeInfo.sumOfDepths + leftTreeInfo.numNodesInTree
    sumOfRightDepths = rightTreeInfo.sumOfDepths + rightTreeInfo.numNodesInTree

    numNodesInTree = 1 + leftTreeInfo.numNodesInTree + rightTreeInfo.numNodesInTree
    sumOfDepths = sumOfLeftDepths + sumOfRightDepths
    sumOfAllDepths = sumOfDepths + leftTreeInfo.sumOfAllDepths + rightTreeInfo.sumOfAllDepths

    return TreeInfo(numNodesInTree, sumOfDepths, sumOfAllDepths)


class TreeInfo:
    def __init__(self, numNodesInTree, sumOfDepths, sumOfAllDepths):
        self.numNodesInTree = numNodesInTree
        self.sumOfDepths = sumOfDepths
        self.sumOfAllDepths = sumOfAllDepths


# This is the class of the input binary tree.
class BinaryTree:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

```