```cpp
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

using namespace std;

class BinaryTree {
public:
  int value;
  BinaryTree *left;
  BinaryTree *right;

  BinaryTree(int value) {
    this->value = value;
    left = NULL;
    right = NULL;
  }
};

int nodeDepths(BinaryTree *node, int depth = 0);

// Average case: when the tree is balanced
// O(nlog(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
int allKindsOfNodeDepths(BinaryTree *root) {
  int sumOfAllDepths = 0;
  vector<BinaryTree *> stack = {root};
  while (stack.size() > 0) {
    BinaryTree *node = stack.back();
    stack.pop_back();
    if (node == NULL)
      continue;
    sumOfAllDepths += nodeDepths(node);
    stack.push_back(node->left);
    stack.push_back(node->right);
  }
  return sumOfAllDepths;
}

int nodeDepths(BinaryTree *node, int depth) {
  if (node == NULL)
    return 0;
  return depth + nodeDepths(node->left, depth + 1) +
         nodeDepths(node->right, depth + 1);
}
```