BALANCED BINARY TREE

Problem asks us to check if a given binary tree is balanced or not.

A balanced binary tree is one where absolute difference between height of left subtree and right subtree is less than 1 at all points

Brute force approach is to find left height and right height at each node, check if it is balanced or not and return a boolean.
Time Complexity is $O(N^2)$
Optimal approach involves modifying the getHeight function, where we return -1 as height if it is not balanced.

Pseudocode :

```
int maxDepth(Node * root) {
    if (root == NULL) {
        return 0;
    }
    int r = maxDepth(root->right);
    int l = maxDepth(root->left);
    if (r == -1 || l == -1) {
        return -1;
    }
```

```
    if (abs(l - r) > 1)  return -1;
    return max(l, r) + 1;
}

bool isBalanced(Node * root) {
    return maxDepth(root) != -1;
}
```