/*Given a binary tree, determine if it is height-balanced.

For this problem, a height-balanced binary tree is defined as:

a binary tree in which the depth of the two subtrees of every node never differ by more than 1.

Example 1:

Given the following tree [3,9,20,null,null,15,7]:

3 / \ 9 20 / \ 15 7

Return true.

Example 2:

Given the following tree [1,2,2,3,3,null,null,4,4]:

1 / \ 2 2 / \ 3 3 / \ 4 4

Return false.*/

-
- 思想：
- (1) 递归思想，添加辅助求节点高度的函数，判断当前节点的两个儿子是否高度值相差小于1

```
public boolean isBalanced(TreeNode root) {

    if (root == null) {
        return true;
    }

    return isBalancedHelper(root);
}

private boolean isBalancedHelper(TreeNode root) {

    if (root.left == null && root.right == null) {
        return true;
    } else if (root.left == null && root.right != null) {
        return treeHeight(root.right) == 1;
    } else if (root.left !=null && root.right == null) {
        return treeHeight(root.left) == 1;
    } else {
        int leftHeight = treeHeight(root.left);
        int rightHeight = treeHeight(root.right);

        if (Math.abs(leftHeight - rightHeight) <= 1) {

            return isBalancedHelper(root.left) && isBalancedHelper(root.right);
```

```
        } else {
            return false;
        }
    }
}

private int treeHeight(TreeNode root) {

    if (root == null) {
        return 0;
    }

    return 1 + Math.max(treeHeight(root.left), treeHeight(root.right));
}
```