① Maximum height / depth of binary tree

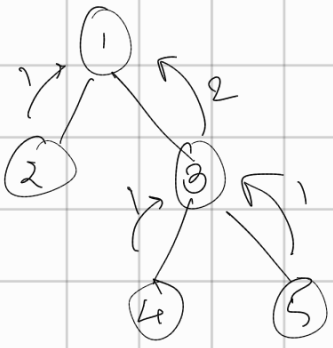
Recursive solution.

CODE

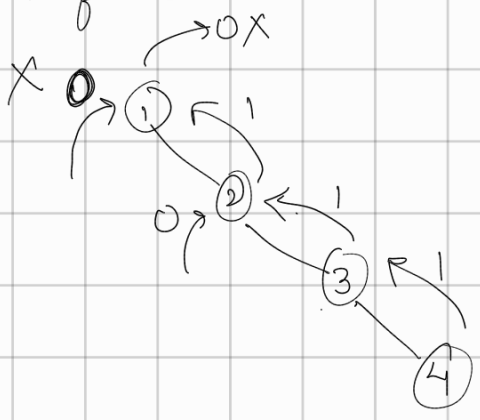
```
if(root == null)  return 0;
return  1 + Math.max( maxDepth(root.left),
                      maxDepth(root.right));
```

For maxm path sum, replace 1 with root.val

③ Minimum distance from root to leaf node



but



So we cannot just return min because that gives WA in node with single child so to handle that-

CODE

```
if (root == null) return 0;
if(root.left == null || root.right == null)
        return 1+Math.max(minDepth(root.left), minDepth(root.right));
return  1 + Math.min(minDepth(root.left), minDepth(root.right));
```

⑥ is Balanced

Balanced ⇒ $|lh - rh| <= 1$

Also, for a tree to be balanced, all nodes of the tree have to be balanced also

∴ CODE:

```
if(root == null) return true;
int h1 = height(root.left), h2 = height(root.right);
if(Math.abs(h1 - h2)<=1)
    return isBalanced(root.left) && isBalanced(root.right);
return false;
```

④ Diameter: longest distance b/w any node to any node
↳ leaf to leaf

at any node, check if [height(left) + height(right) + 1] is max & return max among them

```
int ans;

public int diameter(TreeNode root){
    if(root == null) return 0;
    int h1 = diameter(root.left), h2 = diameter(root.right);
    ans = Math.max(ans, h1 + h2 + 1);
    return Math.max(h1, h2) + 1;
}

public int diameterOfBTree(TreeNode root){
    ans = 0
```

```
    diameter(root);
    return ans > 0 ? ans : 0;
}
```

⑤ Diameter of N-ary tree

find the largest distance between any 2 nodes.

A[i] —edge— i

Max dia for node = (max height + second max height + 1)