LeetCode 104 — Maximum Depth of Binary Tree

🔗 https://leetcode.com/problems/maximum-depth-of-binary-tree/

1. Problem Title & Link

Title: LeetCode 104: Maximum Depth of Binary Tree
Link: https://leetcode.com/problems/maximum-depth-of-binary-tree/

2. Problem Statement (Short Summary)

Given the root of a binary tree, return the **maximum depth** of the tree.

Depth = number of nodes on the longest path from root → leaf.

Example:

Depth 1: root

Depth 2: child

Depth 3: grandchild

3. Examples (Input → Output)

Example 1

Input tree:

```
   3
  / \
 9  20
   /  \
  15   7
```

Output: 3

Example 2

Input: root = []
Output: 0

Example 3

Input:

```
 1
  \
   2
```

Output: 2

4. Constraints

- $0 \le$ number of nodes $\le 10^4$
- Node values not important for depth
- Height of tree can be skewed (like linked list)

5. Core Concept (Pattern / Topic)

⭐ **DFS (Depth First Search)**

Also relates to: Recursion, Tree Height, BFS level traversal

6. Thought Process (Step-by-Step Explanation)

Brute Force / Simple Idea

Depth = 1 + max depth of left subtree + right subtree

Use recursion.

Optimized Idea (DFS)

Define a function:

depth(node):

  if node is null → return 0

  return 1 + max(depth(node.left), depth(node.right))

Alternate Approach: BFS (Level Order)

Count levels using a queue:

- Each level increases depth
- Stop when queue empty

Used when tree is huge and recursion might overflow.

7. Visual / Intuition Diagram (ASCII Diagram)

Tree:

```
   3
  / \
 9  20
    / \
   15  7
```

Depth calculation:

left depth = 1 (node 9)

right depth = 1 + max(1,1) = 2

total depth = 1 + max(1,2) = 3

8. Pseudocode

```
function maxDepth(root):
    if root is null:
        return 0

    left = maxDepth(root.left)
    right = maxDepth(root.right)

    return 1 + max(left, right)
```

9. Code Implementation

✅ **Python**

```python
class Solution:
    def maxDepth(self, root):
        if not root:
            return 0
        return 1 + max(self.maxDepth(root.left), self.maxDepth(root.right))
```

✅ **Java**

```java
class Solution {
    public int maxDepth(TreeNode root) {
        if (root == null) return 0;
        return 1 + Math.max(maxDepth(root.left), maxDepth(root.right));
    }
}
```

10. Time & Space Complexity

DFS Recursive:

- **Time:** O(n) → each node visited once
- **Space:**
  - Best: O(log n) (balanced tree)
  - Worst: O(n) (skewed tree → recursion depth)

BFS Iterative:

- **Time:** O(n)
- **Space:** O(width of tree)

11. Common Mistakes / Edge Cases

⚠ Using depth = left + right (wrong)

⚠ Returning 0 incorrectly for non-null

⚠ Forgetting recursion return value

⚠ Stack overflow on skewed trees (use BFS)

Edge cases:
✔ empty tree → depth = 0
✔ single node → depth = 1
✔ only left chain
✔ only right chain

12. Detailed Dry Run (Step-by-Step Table)

Tree:

```
 1
  \
   2
```

Node	Left Depth	Right Depth	Computed Depth
null	0	0	-
2	0	0	1
1	0	1	2

Final answer = **2**

13. Common Use Cases (Real-Life / Interview)

- Directory structure depth
- HTML DOM depth
- Organization tree analysis
- Recursion-based hierarchical algorithms

14. Common Traps

⚠ Forgetting the +1 for the current node

⚠ Returning depth of left only

⚠ Infinite recursion due to incorrect base case

⚠ Mixing height vs diameter logic

15. Builds To (Related LeetCode Problems)

- LC 110 — Balanced Binary Tree

- LC 543 — Diameter of Binary Tree
- LC 572 — Subtree of Another Tree
- LC 101 — Symmetric Tree

16. Alternate Approaches + Comparison

Approach	Time	Space	Notes
DFS Recursion	O(n)	O(h)	Most common & simplest
BFS Level Order	O(n)	O(w)	Avoid recursion overflow
DFS Iterative	O(n)	O(h)	Uses stack

17. Why This Solution Works (Short Intuition)

The maximum depth of a tree is simply:

1 + deeper subtree among left & right.

DFS naturally explores down to leaves and calculates this height.

18. Variations / Follow-Up Questions

- Find minimum depth instead of maximum
- Count number of nodes at deepest level
- Return the deepest leaf
- Compute depth without recursion (iterative)
- Compute max depth in O(1) space (not possible without threading)