🌳 Introduction to Binary Trees

For beginners and interview preparation

Why Study Trees?

- Trees (and graphs) are the **most common data structure in interviews**.
- Many real-world problems and coding questions revolve around **tree structures**.
- Trees are **recursive** in nature and ideal for learning recursion.

What is a Node?

A **node** is the building block of a tree.
It contains:

1. **Data** – Any kind of value (int, bool, object, etc.)
2. **Pointers** – References to other nodes

What is a Graph?

- A **graph** is a group of **nodes (vertices)** and **edges (connections)**.
- **Linked Lists** and **Trees** are **special types of graphs**.
- But in coding problems, trees and general graphs are treated separately because graphs are broader and harder.

🌲 What is a Tree?

- A **tree is a type of graph**.
- In this course, we focus on **Binary Trees** – each node has **at most two children**.
- Trees start from a special node called the **root**.

🌿 Binary Tree Terminology

Term	**Meaning**
Root	The top-most node of the tree
Parent	A node that has one or more children
Child	A node connected from a parent
Leaf	A node with no children
Depth	Number of edges from the root to a node (root = depth 0)
Sub Tree	Any node and all of its descendants – can be treated as a tree

Why Are Subtrees Important?

- Every node in a binary tree can be viewed as the **root of its own subtree**.
- This is key to solving tree problems using **recursion**.

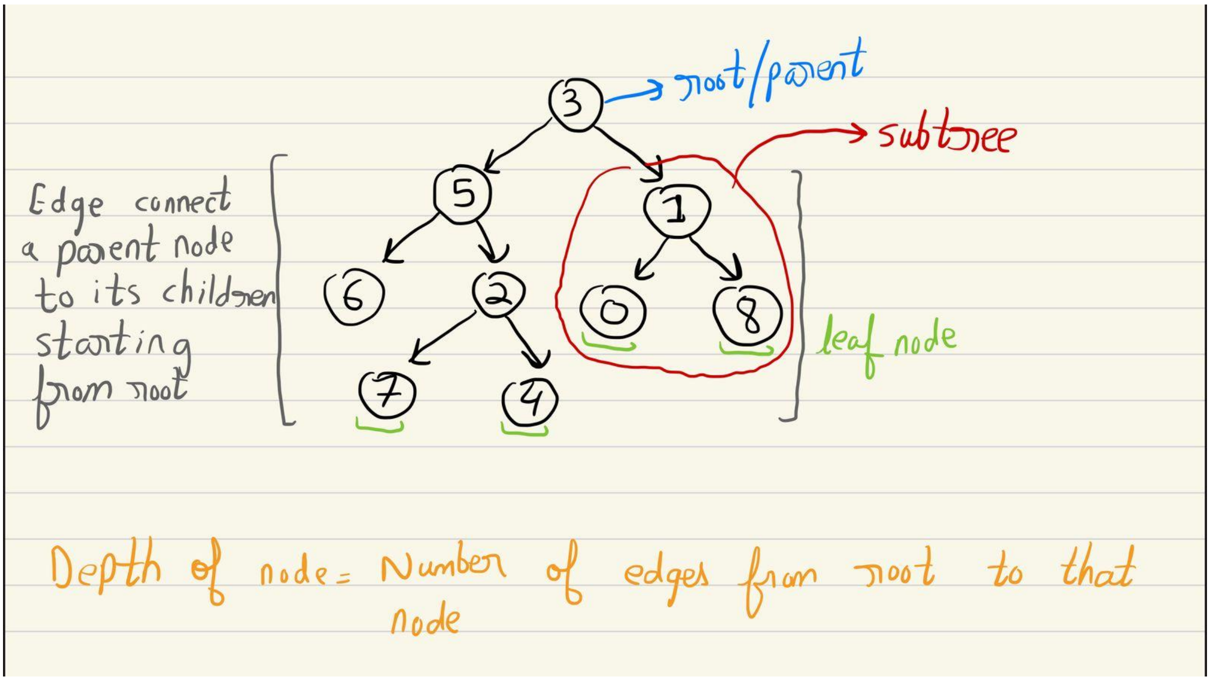
Example:
Think of a company structure:
The **CEO** is the root.
The **SVP of Engineering** is a subtree.
If you only care about engineers, treat the SVP as the root of a new tree (subtree).
This makes it easier to break problems down.

Important Takeaway:

Binary Trees are:

- **Recursive**, so most problems can be solved using recursion.
- **Structured**, with root-child-leaf relationships.
- **Modular**, with every node forming a potential tree of its own.

Visual Representation:


3
root/parent
subtree
5
1
6
2
0
8
7
4
Edge connect a parent node to its children starting from root
leaf node
Depth of node = Number of edges from root to that node

Code representation

Just like with a linked list, binary trees are implemented using objects of a custom class. This is the typical class definition that will be provided to you in algorithm problems:

- **Python3**

```
class TreeNode:
  def __init__(self, val, left, right):
    self.val = val
    self.left = left
    self.right = right
```

- **C++**

```
struct TreeNode {
  int val;
  TreeNode *left;
  TreeNode *right;
  TreeNode(int val) : val(val), left(nullptr), right(nullptr) {}
};
```