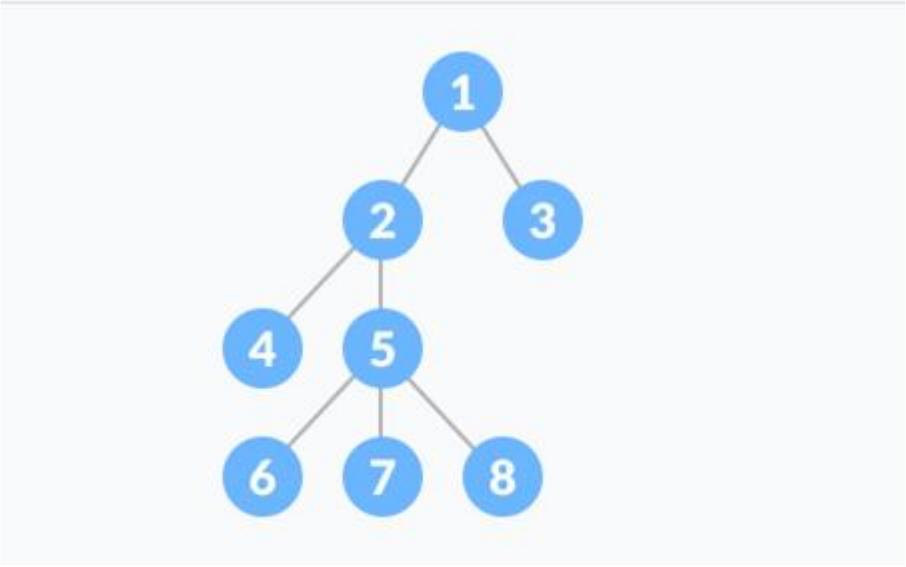
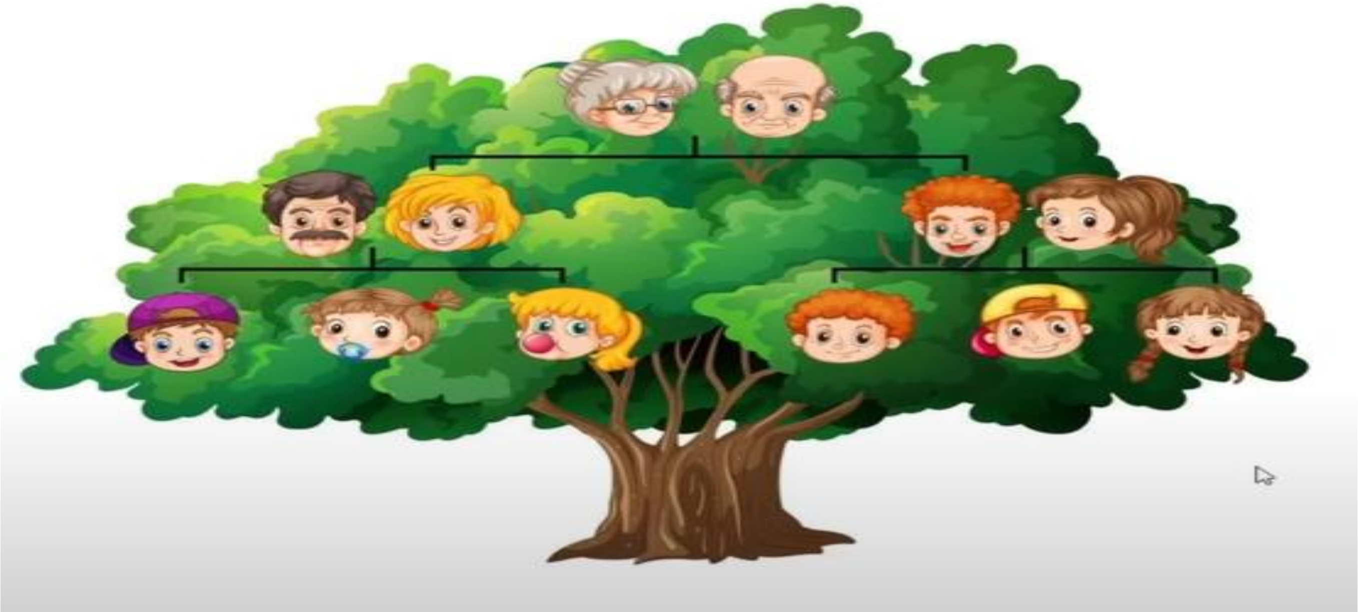
Tree Data Structure

In this tutorial, you will learn about tree data structure. Also, you will learn about different types of trees and the terminologies used in tree.

A tree is a nonlinear hierarchical data structure that consists of nodes connected by edges.



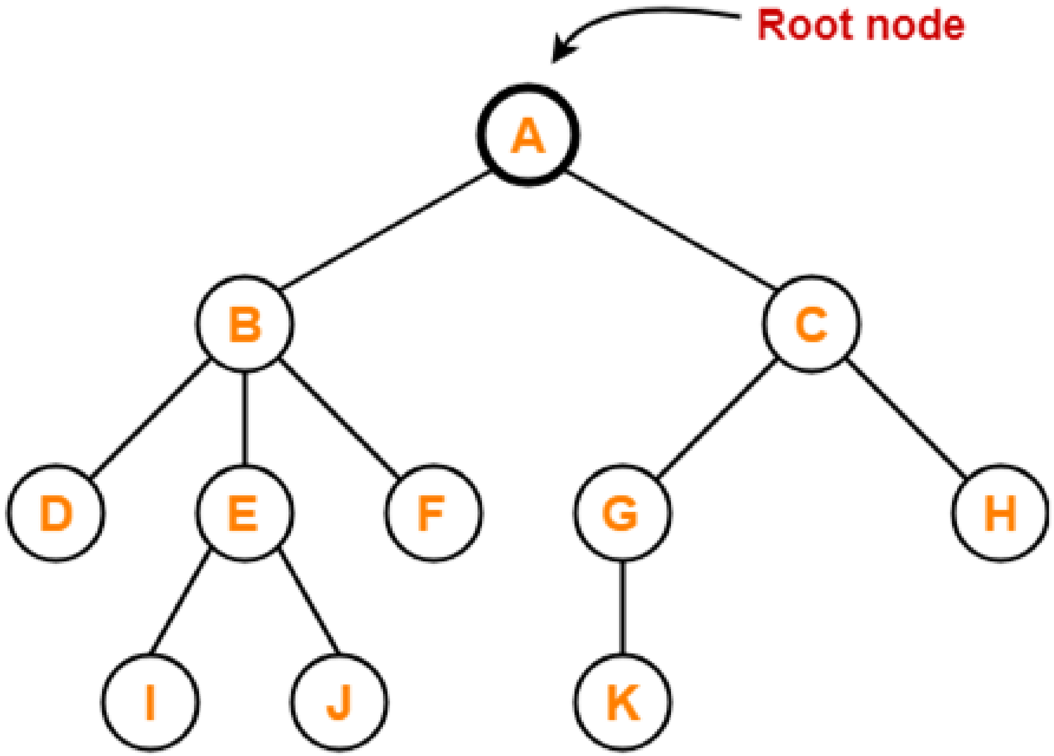


Root

The first node from where the tree originates is called as a **root node**.

In any tree, there must be only one root node.

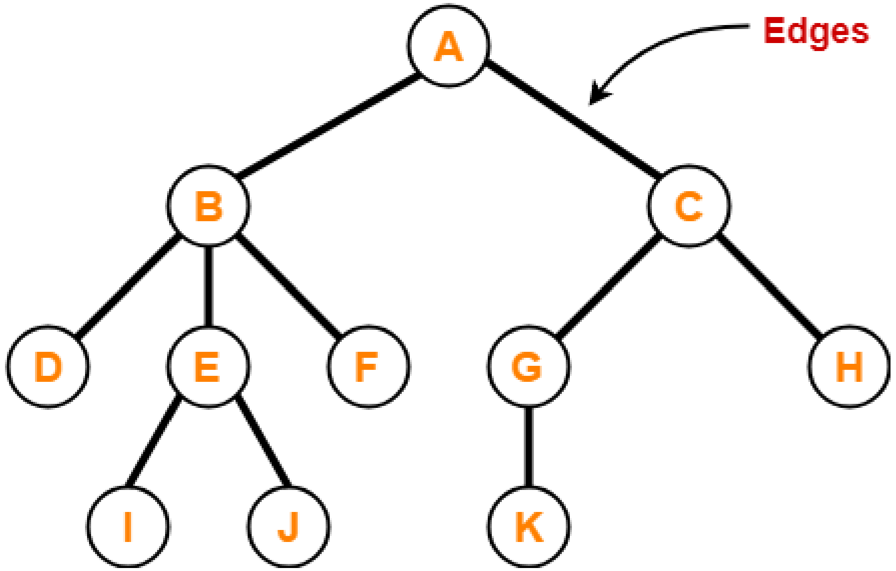
We can never have multiple root nodes in a tree data structure.



Here, node A is the only root node.

Edge

The connecting link between any two nodes is called as an **edge**.



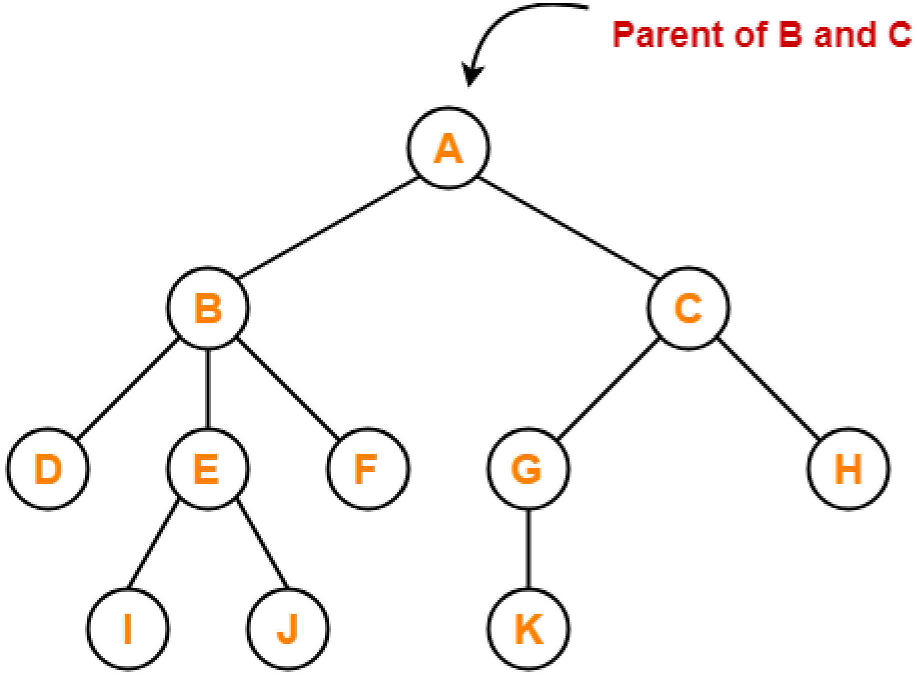
Parent

The node which has a branch from it to any other node is called as a **parent node**.

In other words, the node which has one or more children is called as a parent node.

In a tree, a parent node can have any number of child nodes.

- Node A is the parent of nodes B and C
- Node B is the parent of nodes D, E and F
- Node C is the parent of nodes G and H
- Node E is the parent of nodes I and J
- Node G is the parent of node K

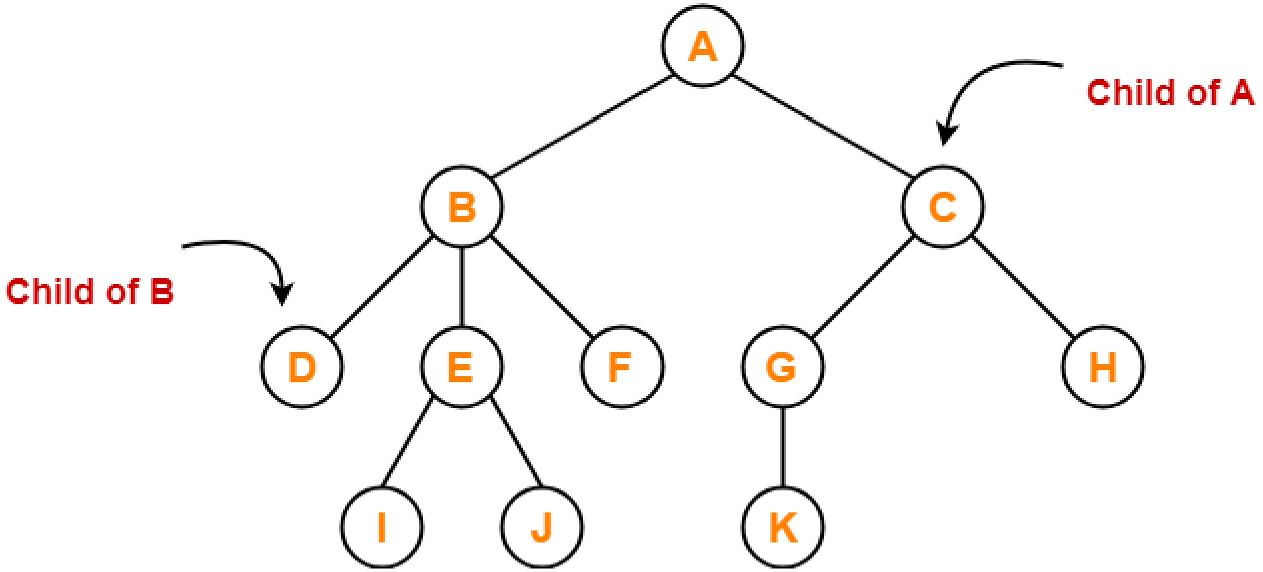


Child

The node which is a descendant of some node is called as a **child node**.

All the nodes except root node are child nodes.

- Nodes B and C are the children of node A
- Nodes D, E and F are the children of node B
- Nodes G and H are the children of node C
- Nodes I and J are the children of node E
- Node K is the child of node G

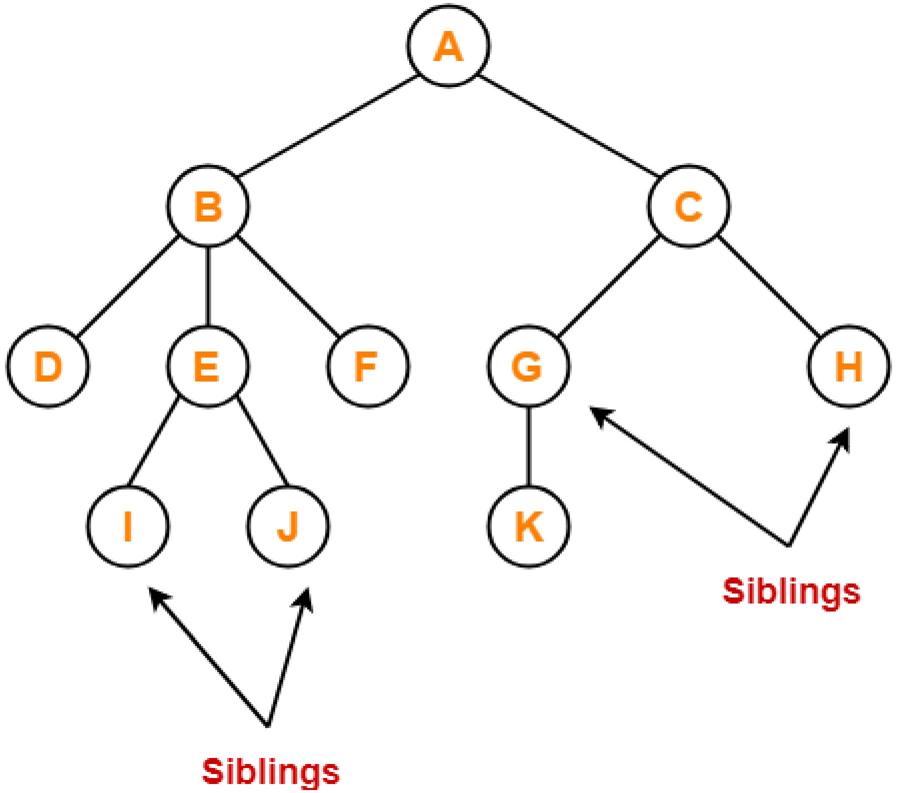


Siblings

Nodes which belong to the same parent are called as **siblings**.

In other words, nodes with the same parent are sibling nodes.

- Nodes B and C are siblings
- Nodes D, E and F are siblings
- Nodes G and H are siblings
- Nodes I and J are siblings

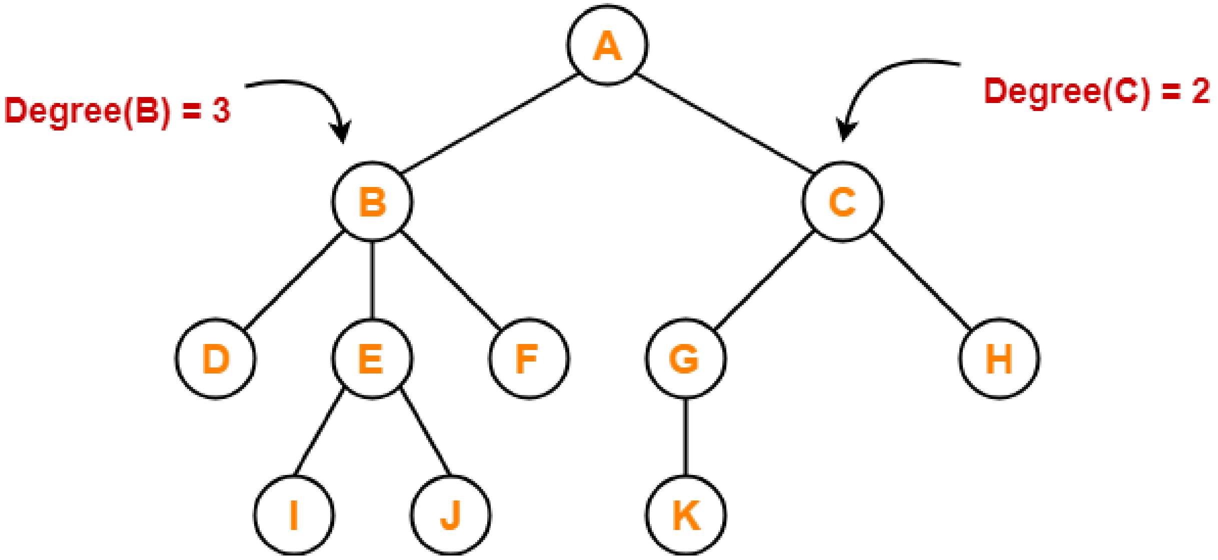


Degree

Degree of a node is the total number of children of that node.

Degree of a tree is the highest degree of a node among all the nodes in the tree.

- Degree of node A = 2
- Degree of node B = 3
- Degree of node C = 2
- Degree of node D = 0
- Degree of node E = 2
- Degree of node F = 0
- Degree of node G = 1
- Degree of node H = 0
- Degree of node I = 0
- Degree of node J = 0
- Degree of node K = 0

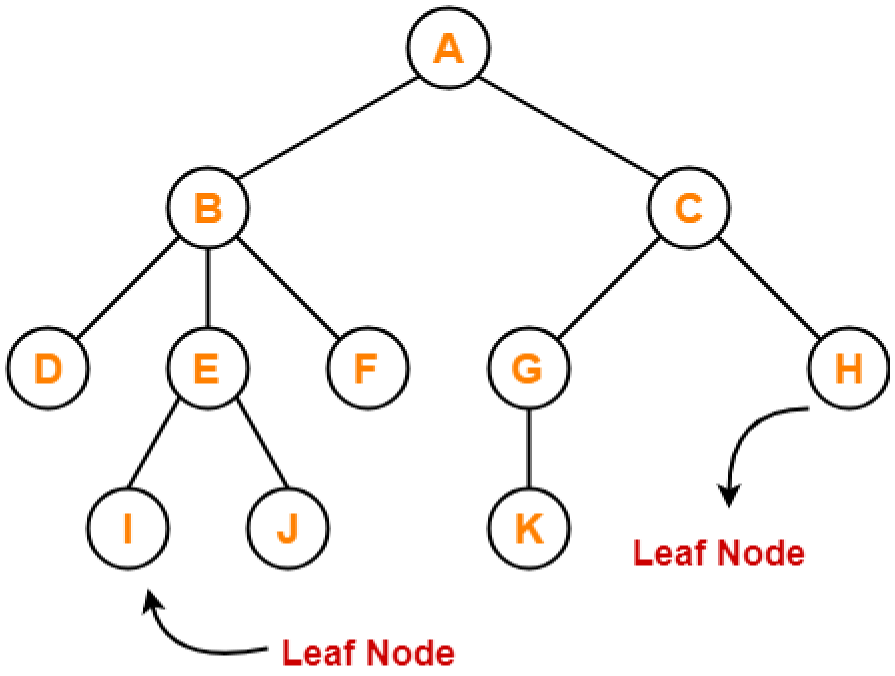


Leaf Node

The node which does not have any child is called as a **leaf node**.

Leaf nodes are also called as **external nodes** or **terminal nodes**.

Here, nodes D, I, J, F, K and H are leaf nodes.

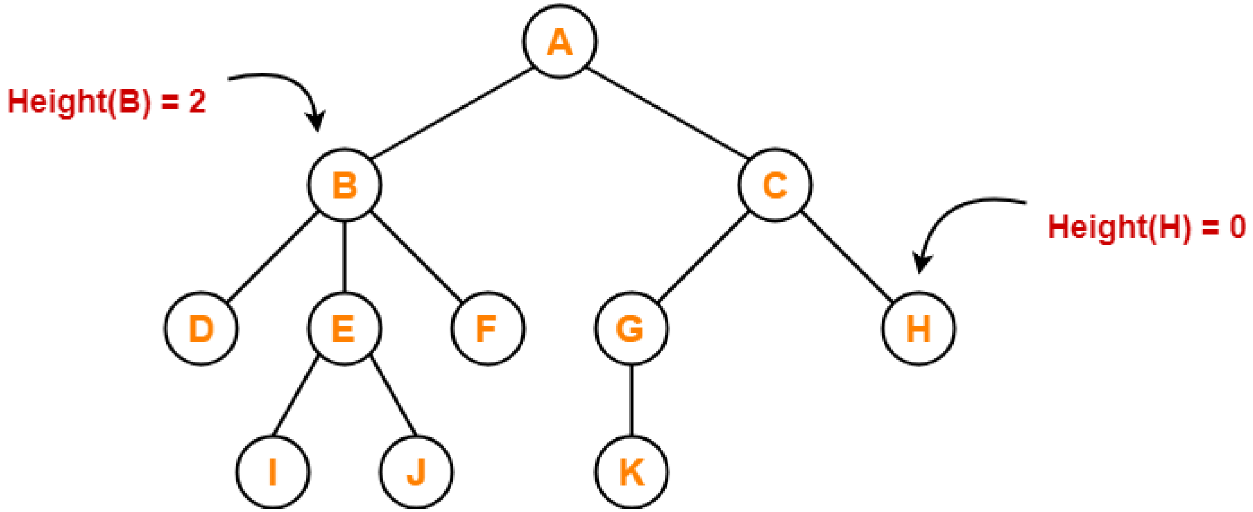


Height

Total number of edges that lies on the longest path from any leaf node to a particular node is called as **height of that node**.

Height of a tree is the height of root node.

- **Height of all leaf nodes = 0**
- **Height of node A = 3**
- **Height of node B = 2**
- **Height of node C = 2**
- **Height of node D = 0**
- **Height of node E = 1**
- **Height of node F = 0**
- **Height of node G = 1**
- **Height of node H = 0**
- **Height of node I = 0**
- **Height of node J = 0**
- **Height of node K = 0**

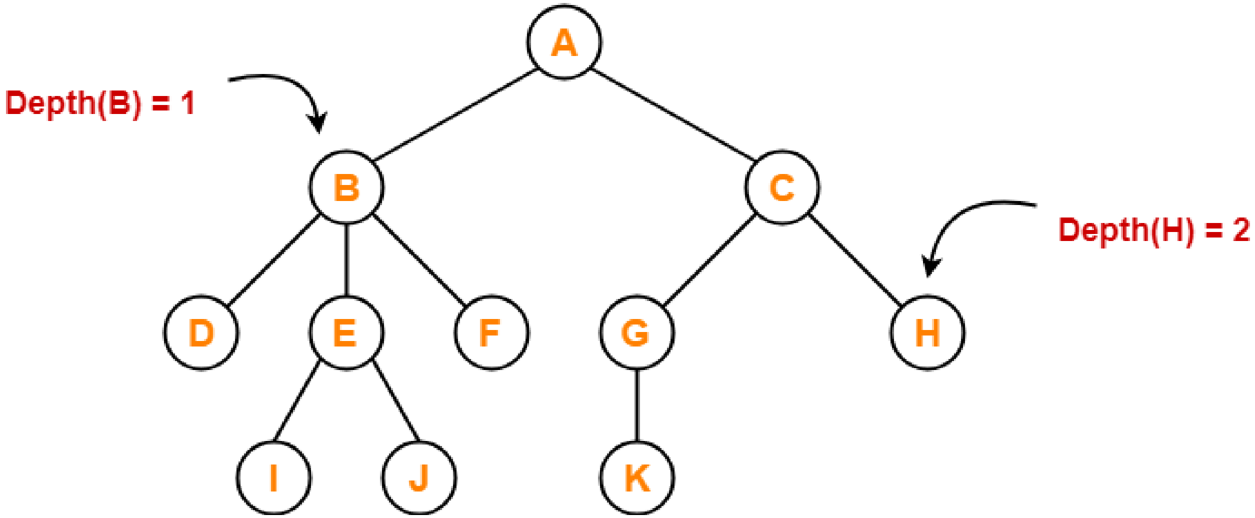


Depth

Total number of edges from root node to a particular node is called as **depth of that node**.

Depth of the root node = 0

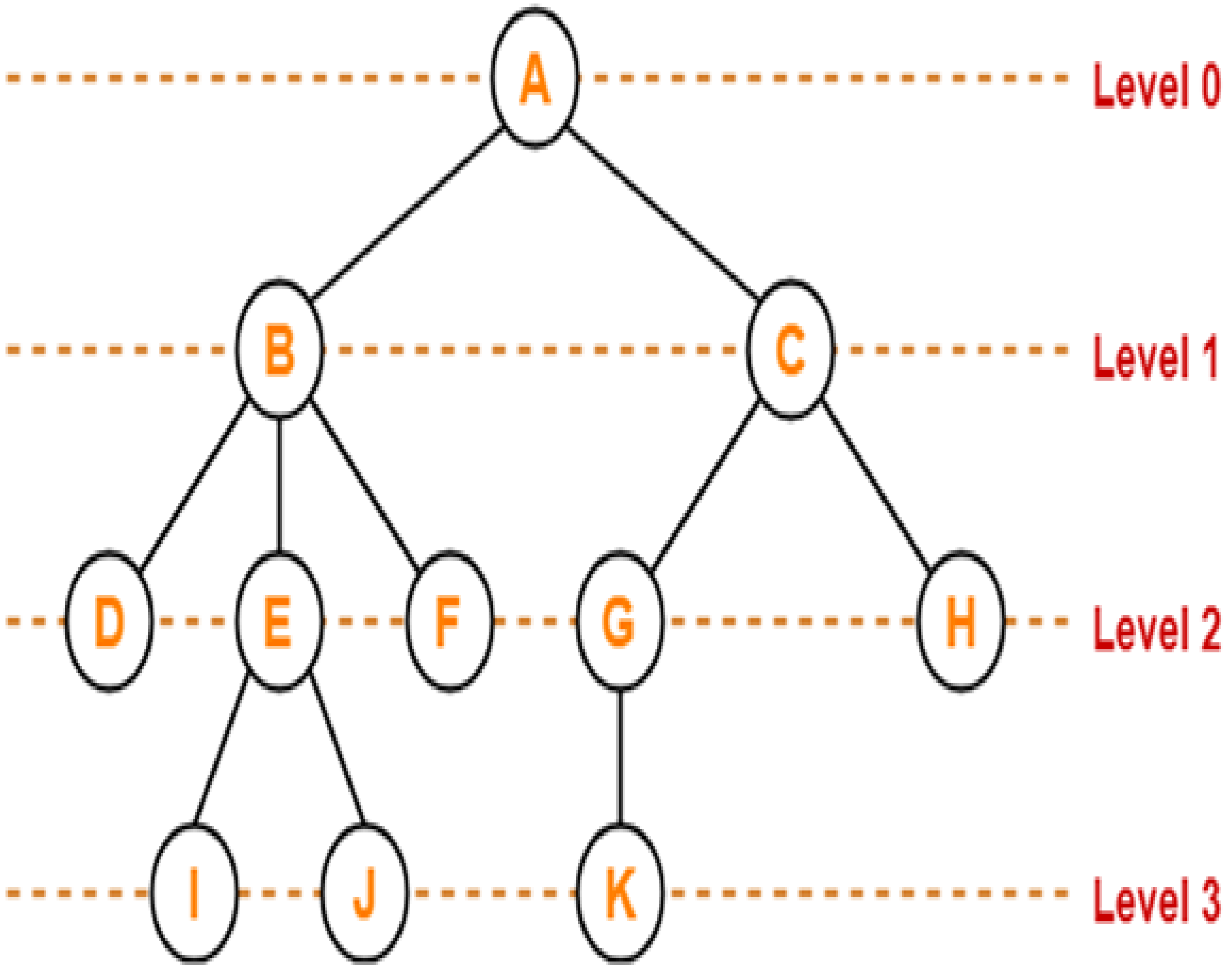
- **Depth of node A = 0**
- **Depth of node B = 1**
- **Depth of node C = 1**
- **Depth of node D = 2**
- **Depth of node E = 2**
- **Depth of node F = 2**
- **Depth of node G = 2**
- **Depth of node H = 2**
- **Depth of node I = 3**
- **Depth of node J = 3**
- **Depth of node K = 3**



Level

In a tree, each step from top to bottom is called as **level of a tree**.

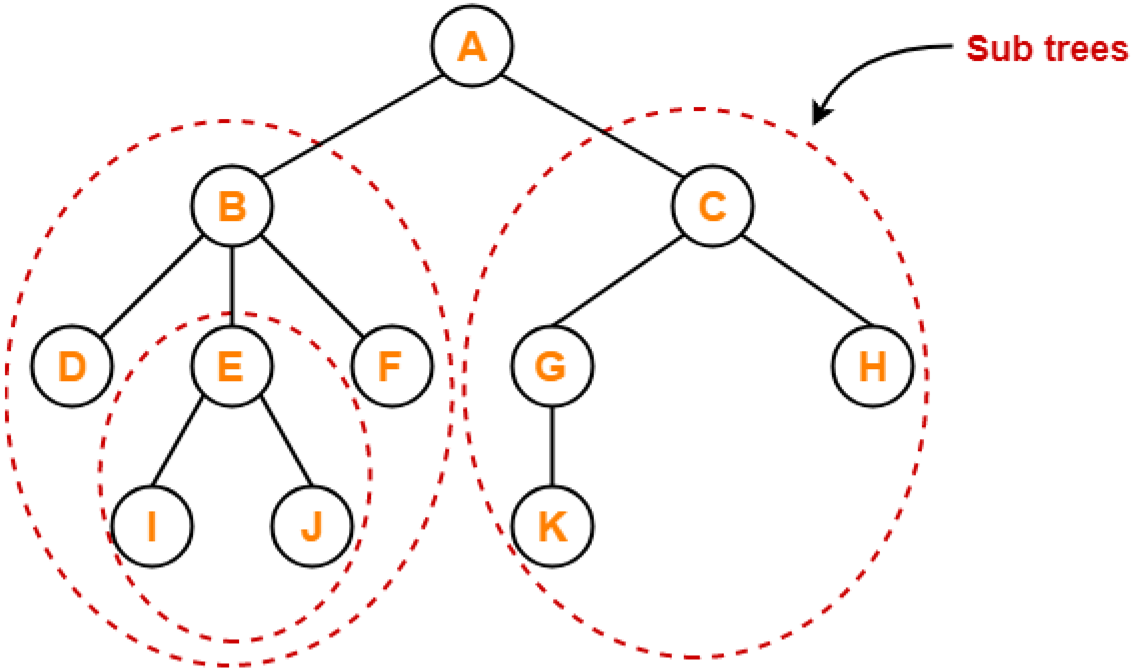
The level count starts with 0 and increments by 1 at each level or step



Subtree

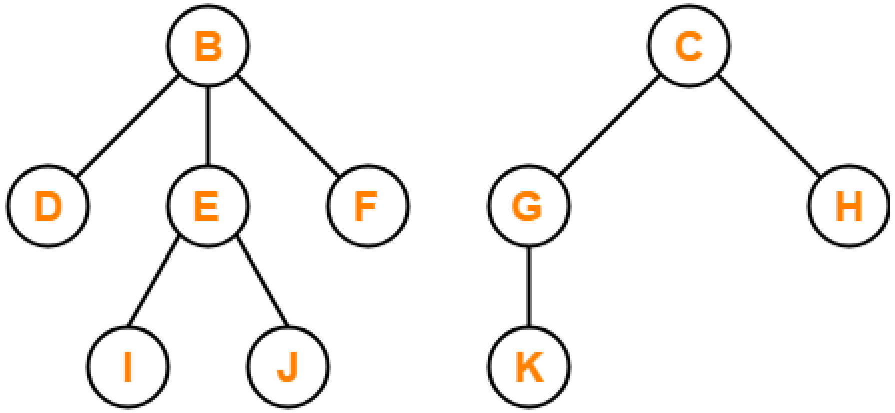
In a tree, each child from a node forms a **subtree** recursively.

Every child node forms a subtree on its parent node.



Forest

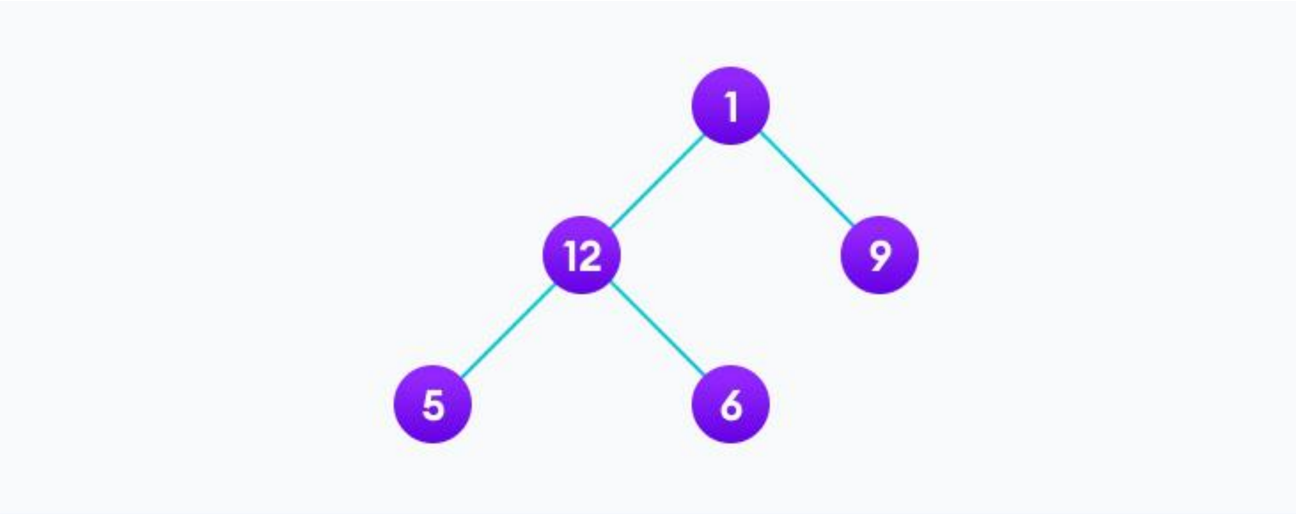
A forest is a set of disjoint trees.



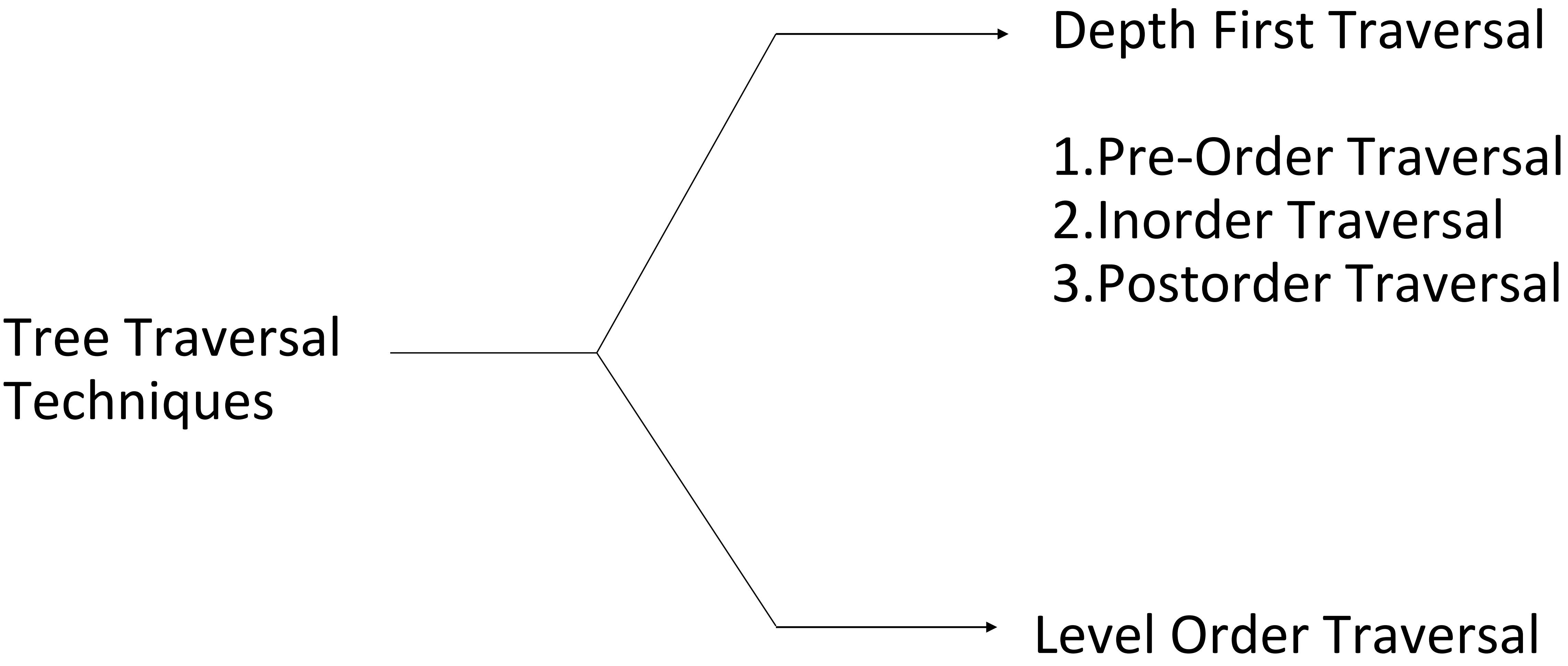
Forest

Tree Traversal

Tree traversal refers to the process of visiting each node in a tree data structure.



Tree traversal



Tree Traversal Techniques

Depth First Traversal

1. Pre-Order Traversal
2. Inorder Traversal
3. Postorder Traversal

Level Order Traversal

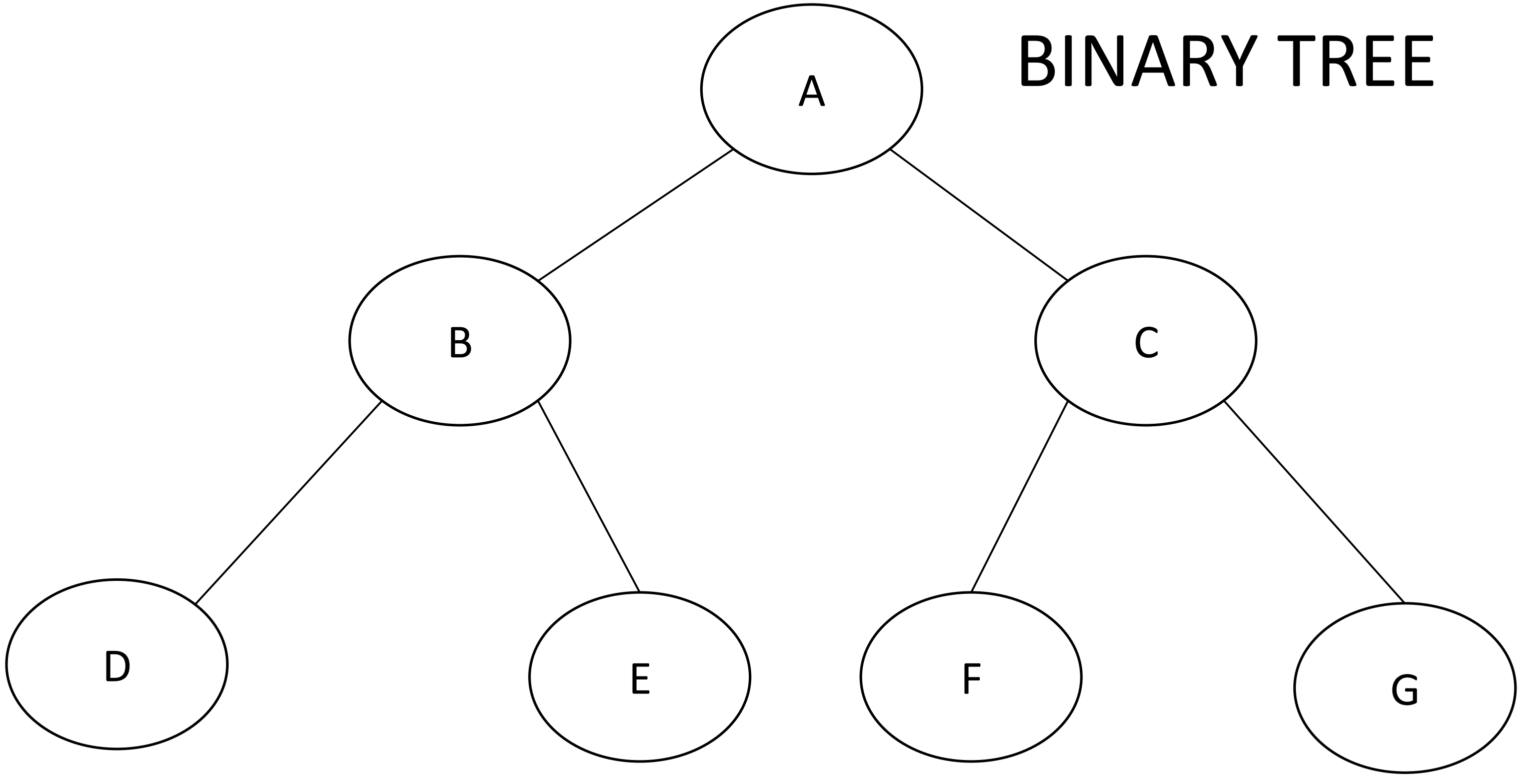
Pre-Order Traversal

- Visit The root
- Traverse the left subtree
- Traverse the right subtree

REMEMBER:

ROOT-LEFT-RIGHT

BINARY TREE



Pre-Order Traversal: A B D E C F G

Shortcut For Pre-Order Traversal

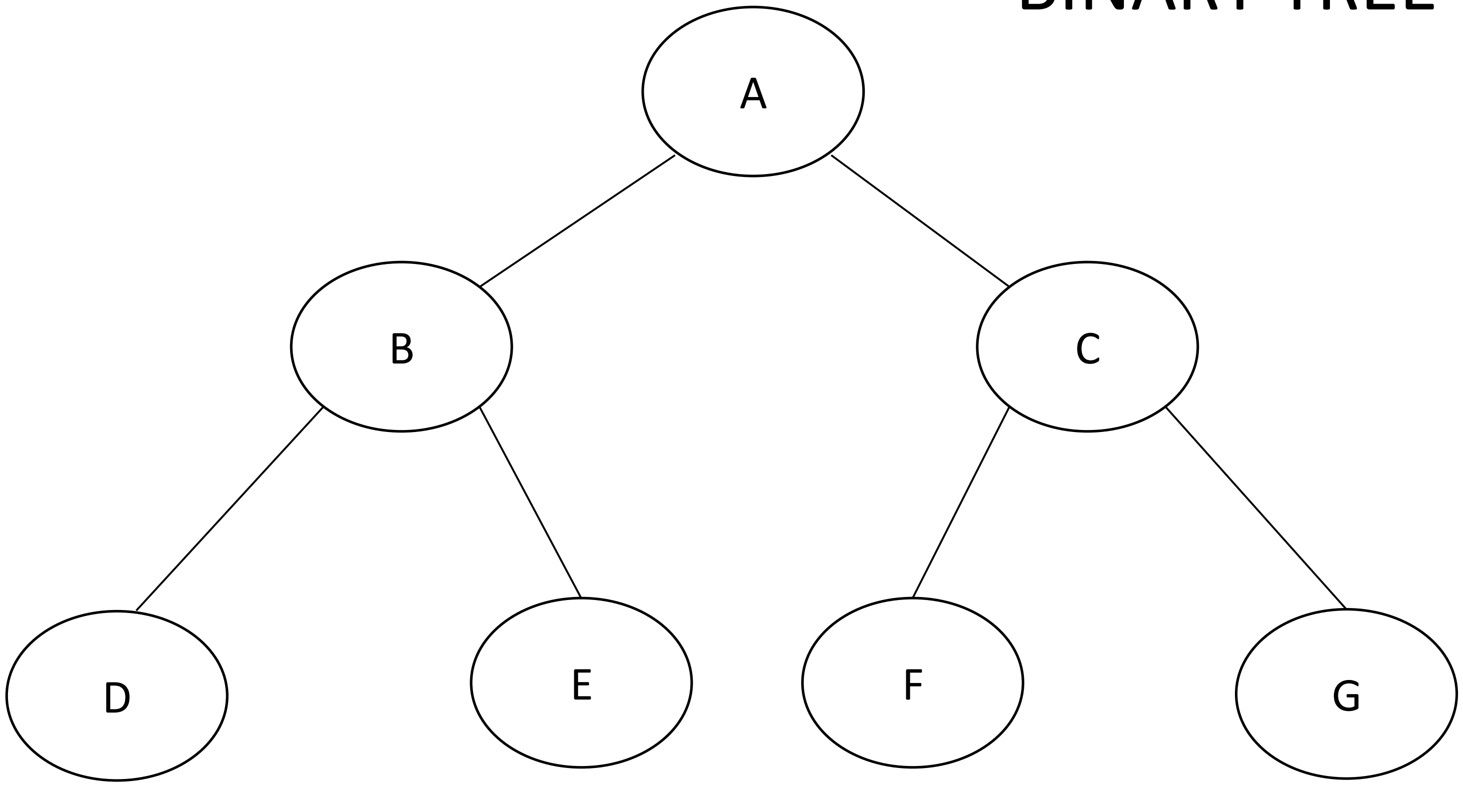
Pre-Order Traversal

Just traverse the entire tree starting from the root node keeping yourself to the left

REMEMBER:

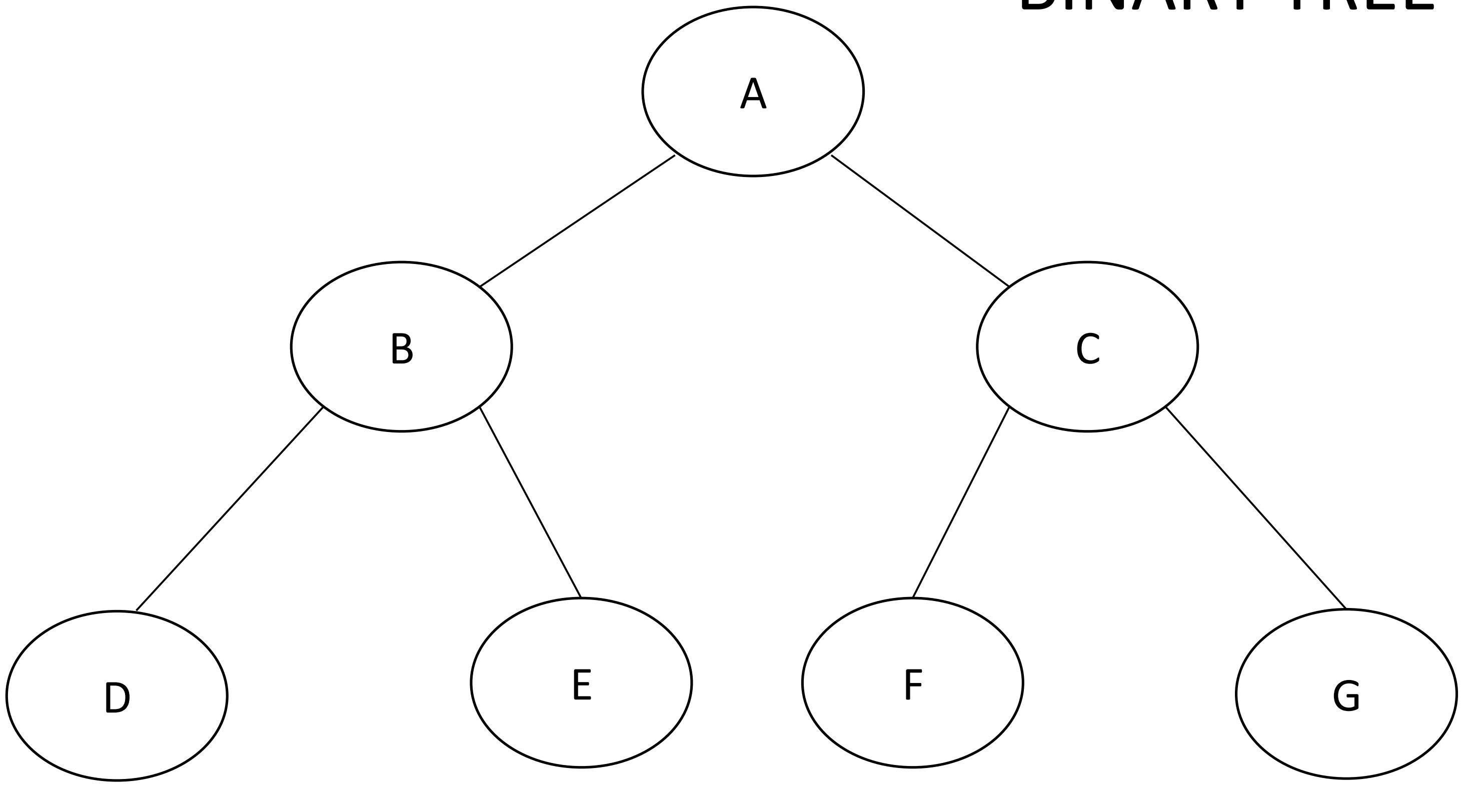
ROOT-LEFT-RIGHT

BINARY TREE



Pre-Order Traversal: A B D E C F G

BINARY TREE



In-Order Traversal

- Traverse the left subtree
- Visit the root
- Traverse the right subtree

REMEMBER:

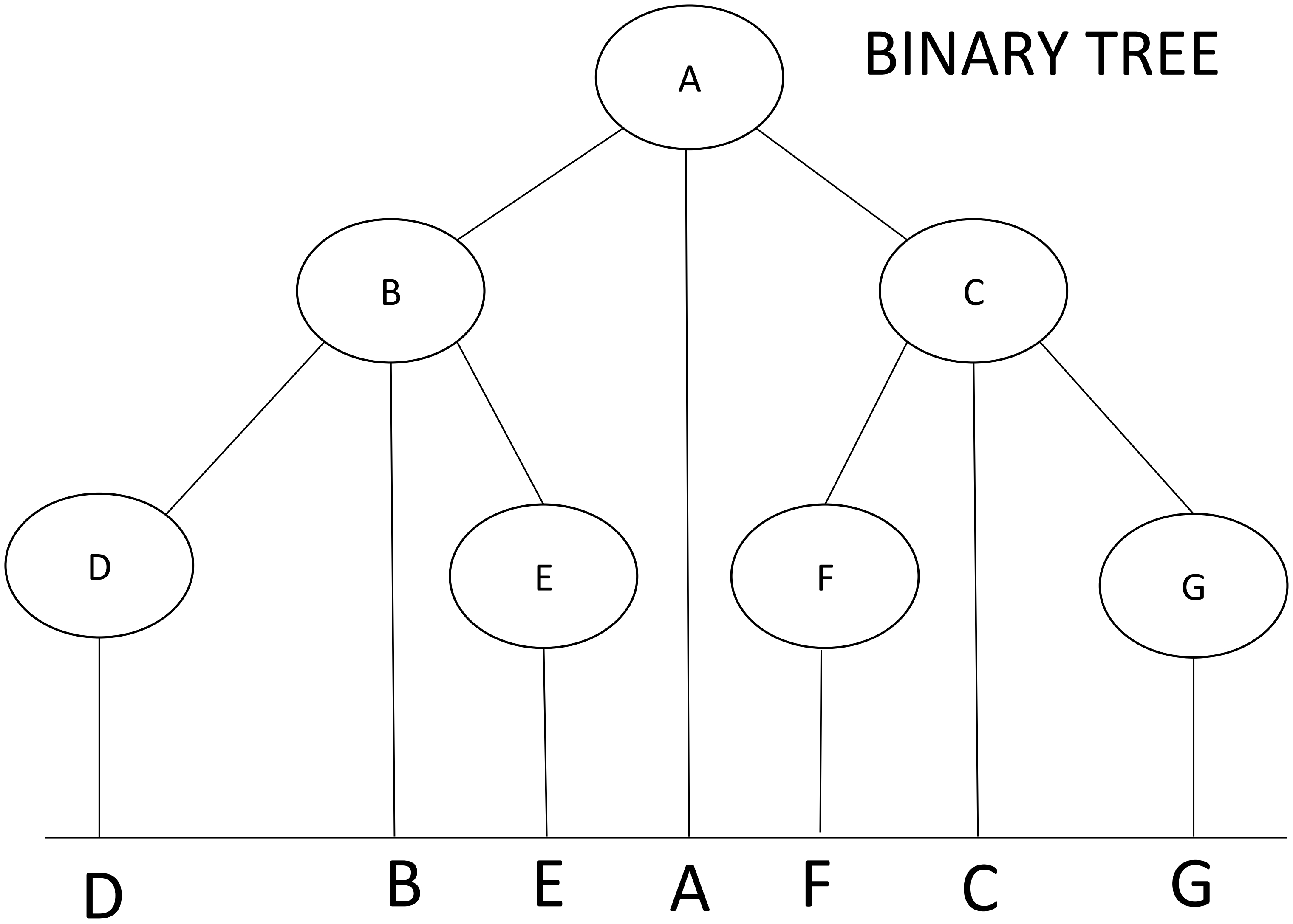
LEFT-ROOT RIGHT

Pre-Order Traversal: D B E AF C G

Shortcut For In Order Traversal

In Order Traversal

Just keep a plane mirror horizontally at the bottom of the tree and take the projection of all nodes.



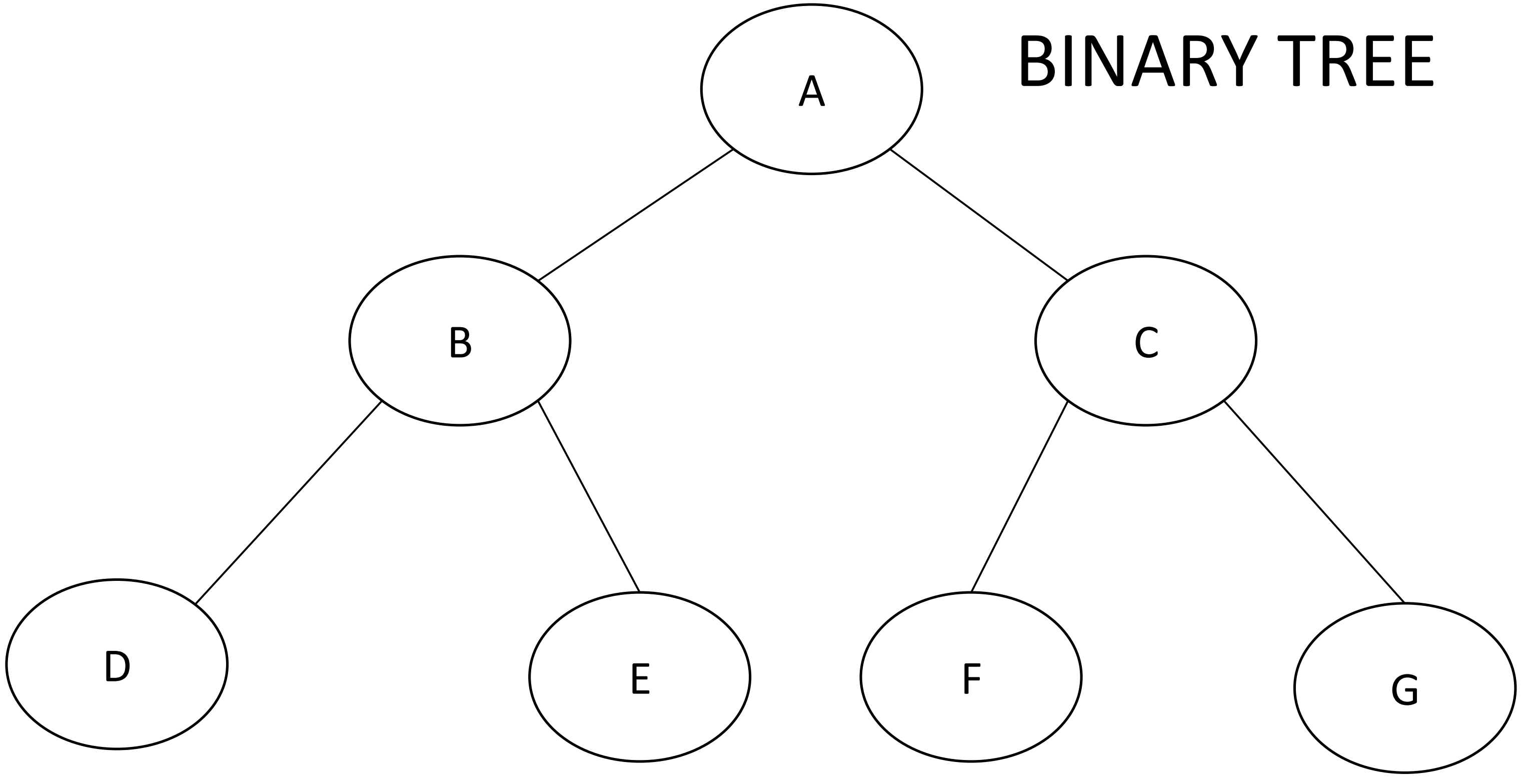
In Order Traversal: D B E AF C G

Post-Order Traversal

- Traverse the left subtree
- Traverse the right subtree
- Visit the root

REMEMBER:

LEFT-RIGHT-ROOT



BINARY TREE

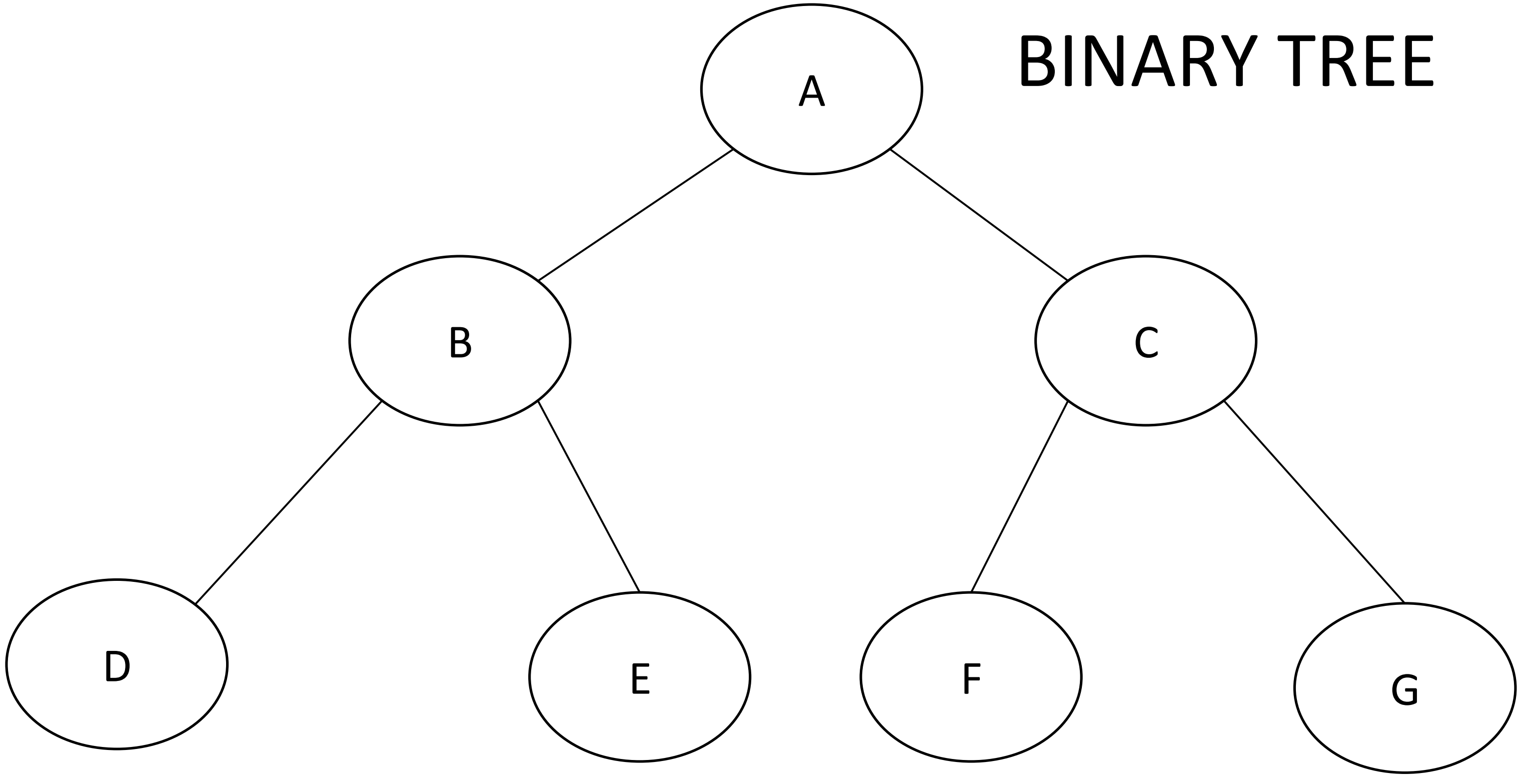
Post-Order Traversal: D E B F G C A

Shortcut For In Order Traversal

Post-Order Traversal

Just pluck the leftmost leaf nodes

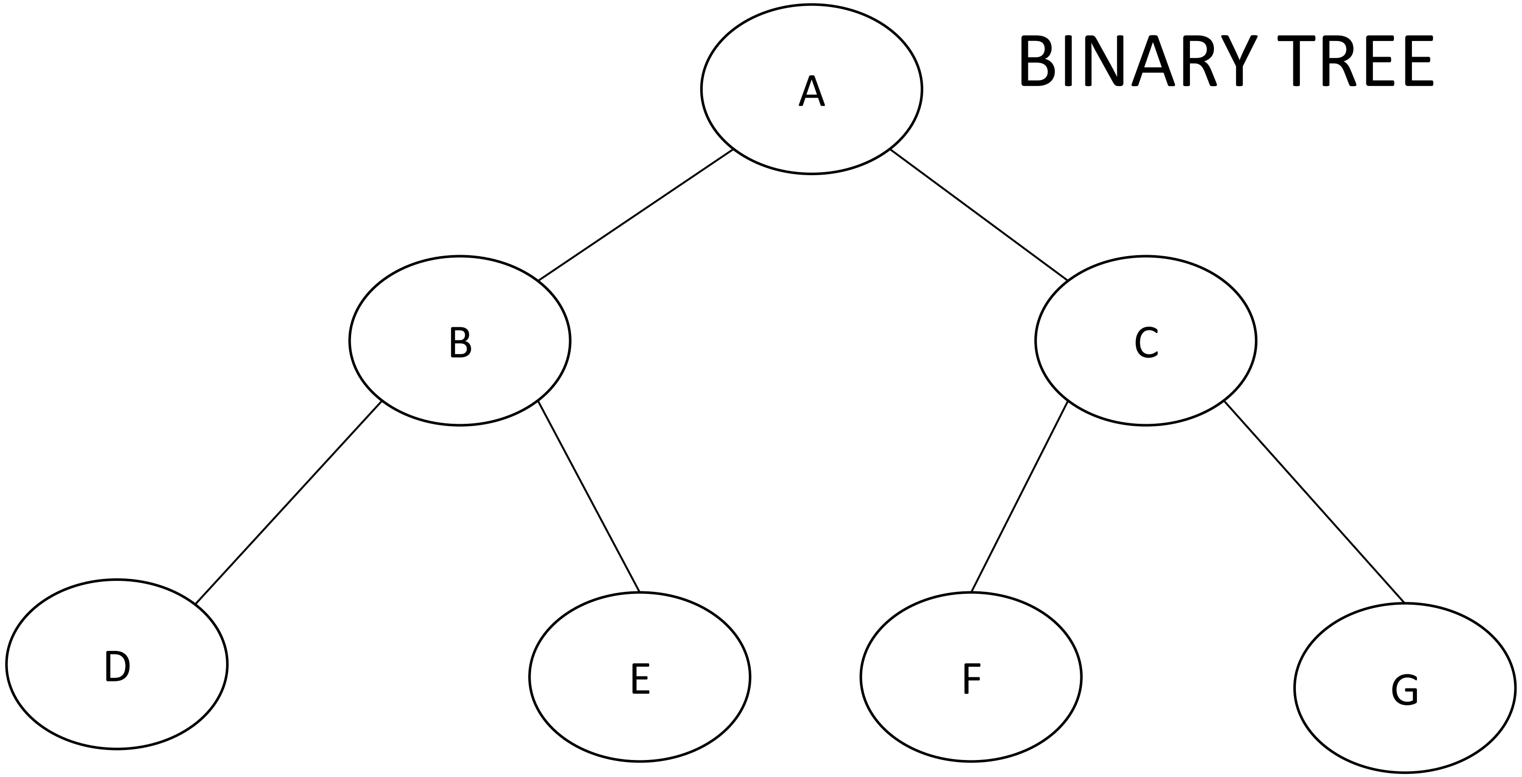
One by one.



Post-Order Traversal: D E B F G C A

Level Order Traversal

Level Order traversal of a tree is the breadth first traversal of a tree which points all the nodes of a tree level by level



BINARY TREE

Level Order Traversal :
A B C D E F G