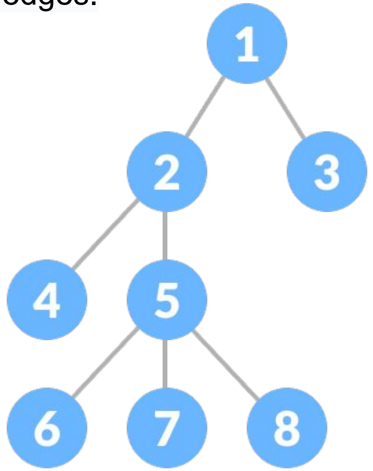
Tree Data Structure

Tree

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



Why Tree Data Structure?

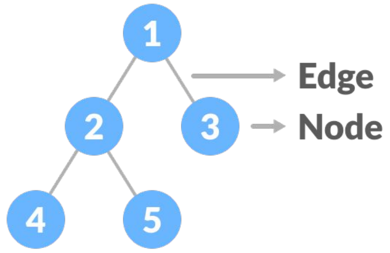
Other data structures such as arrays, linked list, stack, and queue are linear data structures that store data sequentially. In order to perform any operation in a linear data structure, the time complexity increases with the increase in the data size. But, it is not acceptable in today's computational world.

Different tree data structures allow quicker and easier access to the data as it is a non-linear data structure.

Tree Terminologies

Node: A node is an entity that contains a key or value and pointers to its child nodes.

The last nodes of each path are called **leaf nodes** or **external nodes** that do not contain a link/pointer to child nodes. The node having at least a child node is called an **internal node.**



Edge: It is the link between any two nodes.

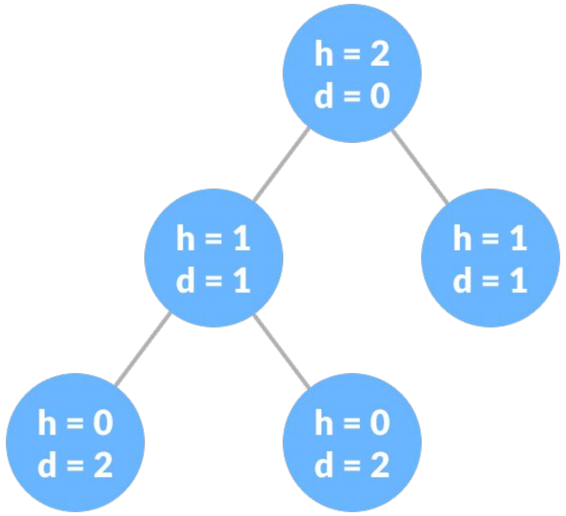
Root: It is the topmost node of a tree.

Height of a Node: The height of a node is the number of edges from the node to the deepest leaf (ie. the longest path from the node to a leaf node).

Depth of a Node: The depth of a node is the number of edges from the root to the node.

Height of a Tree: The height of a Tree is the height of the root node or the depth of the deepest node.

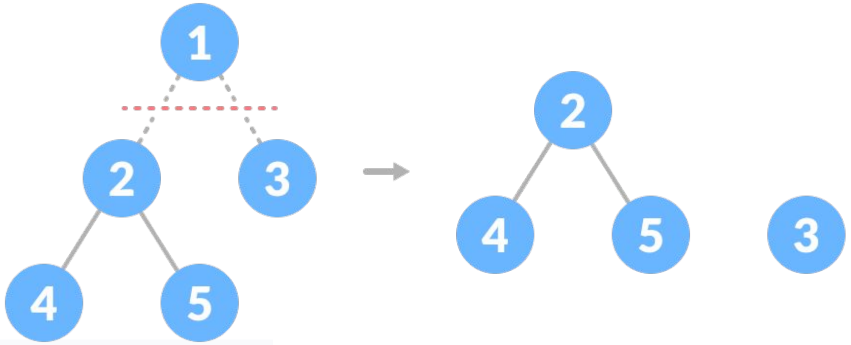
Height and depth of each node in a tree



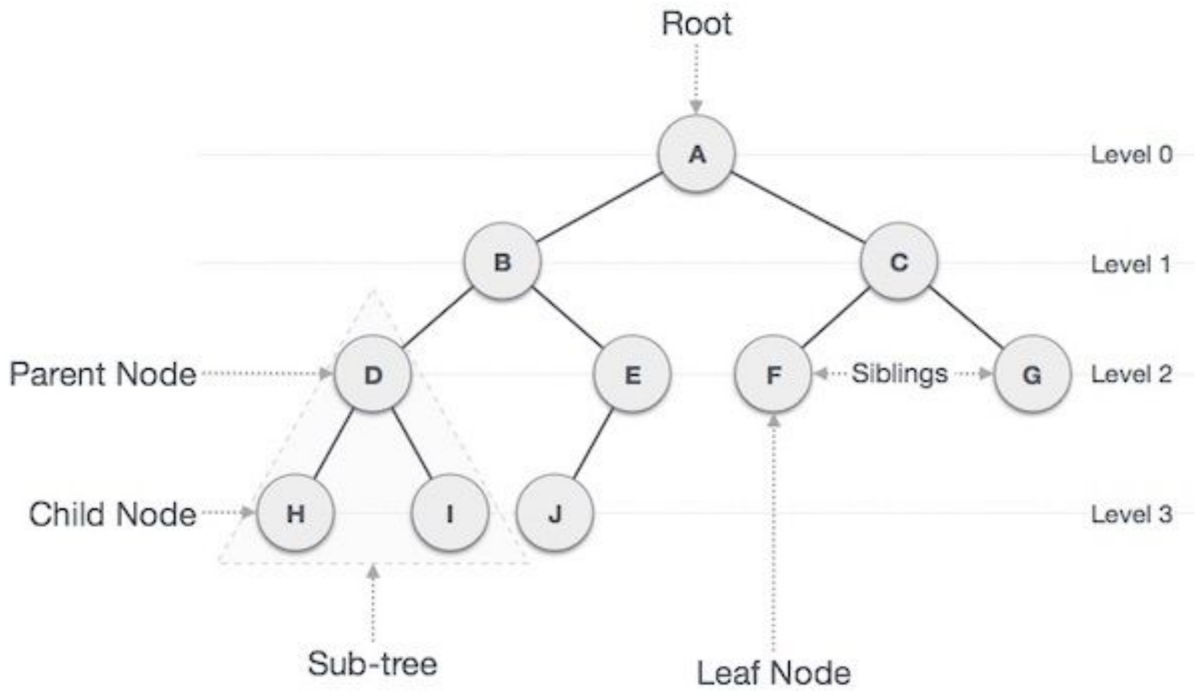
Degree of a Node: The degree of a node is the total number of branches of that node.

Forest: A collection of disjoint trees is called a forest.

Creating forest from a tree



You can create a forest by cutting the root of a tree.


Root
A
B
C
D
E
F
G
H
I
J
Level 0
Level 1
Level 2
Level 3
Parent Node
Siblings
Child Node
Sub-tree
Leaf Node

Important Terms

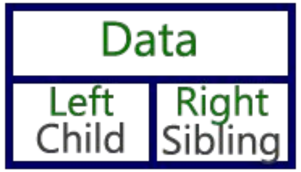
Following are the important terms with respect to tree.

- Path − Path refers to the sequence of nodes along the edges of a tree.
- Root − The node at the top of the tree is called root. There is only one root per tree and one path from the root node to any node.
- Parent − Any node except the root node has one edge upward to a node called parent.
- Child − The node below a given node connected by its edge downward is called its child node.
- Leaf − The node which does not have any child node is called the leaf node.
- Subtree − Subtree represents the descendants of a node.
- Visiting − Visiting refers to checking the value of a node when control is on the node.
- Traversing − Traversing means passing through nodes in a specific order.
- Levels − Level of a node represents the generation of a node. If the root node is at level 0, then its next child node is at level 1, its grandchild is at level 2, and so on.
- keys − Key represents a value of a node based on which a search operation is to be carried out for a node.

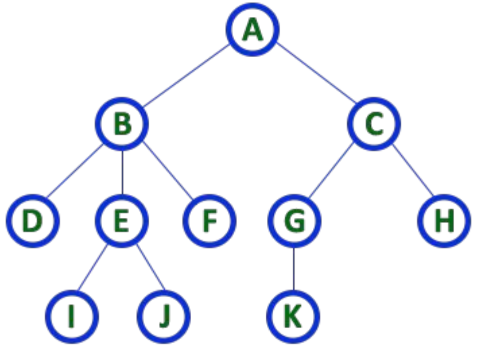
Tree Representation

Left Child - Right Sibling Representation

In this representation, we use a list with one type of node which consists of three fields namely Data field, Left child reference field and Right sibling reference field. Data field stores the actual value of a node, left reference field stores the address of the left child and right reference field stores the address of the right sibling node. Graphical representation of that node is as follows…

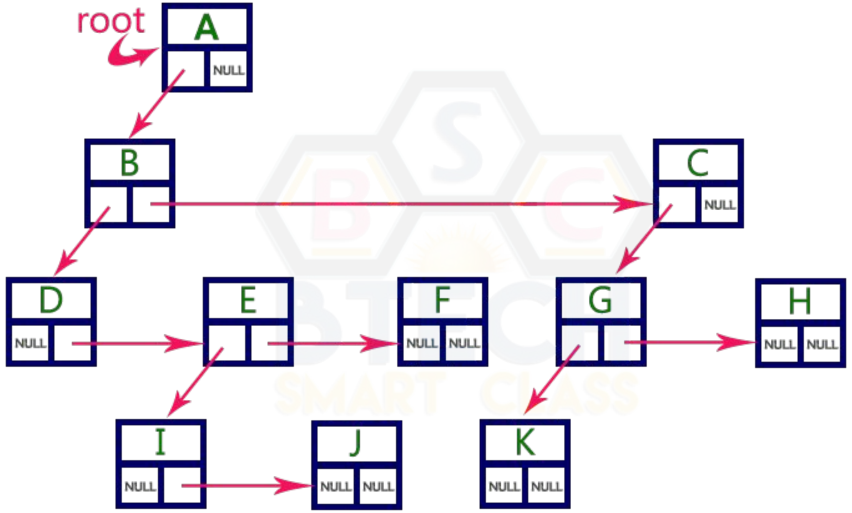


In this representation, every node's data field stores the actual value of that node. If that node has left a child, then left reference field stores the address of that left child node otherwise stores NULL. If that node has the right sibling, then right reference field stores the address of right sibling node otherwise stores NULL.



TREE with 11 nodes and 10 edges

- In any tree with 'N' nodes there will be maximum of 'N-1' edges
- In a tree every individual element is called as 'NODE'

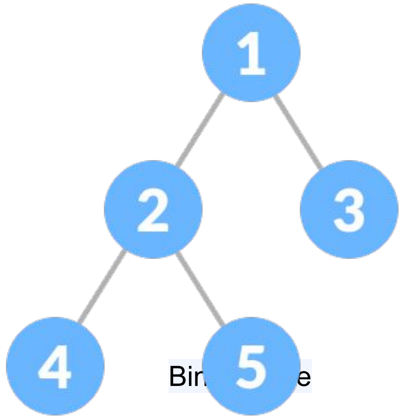


Types of Tree

1. Binary Tree
2. Binary Search Tree
3. AVL Tree
4. B Tree
5. B+ Tree
6. Red Black Tree
7. Trie
8. Heap Tree ... etc...

Binary Tree

A binary tree is a tree data structure in which each parent node can have at most two children.

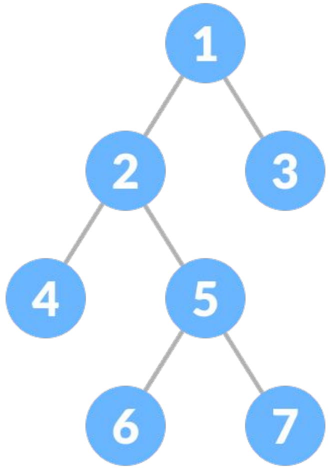


Bi e

Types of Binary Tree

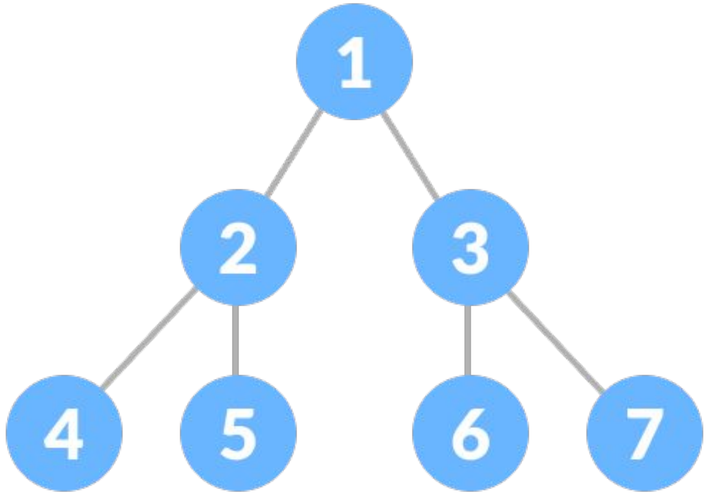
Full Binary Tree

A full Binary tree is a special type of binary tree in which every parent node/internal node has either two or no children. A full binary tree is also called **strictly binary tree** or **proper binary tree** or **2-tree**



Perfect Binary Tree

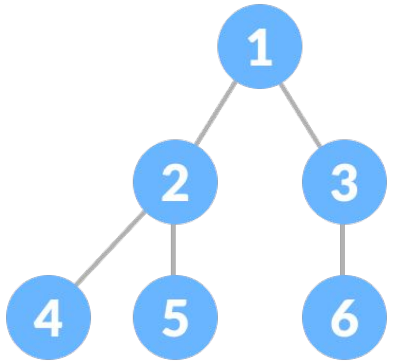
A perfect binary tree is a type of binary tree in which every internal node has exactly two child nodes and all the leaf nodes are at the same level.



Complete Binary Tree

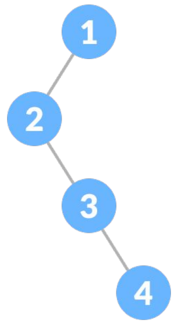
A complete binary tree is just like a full binary tree, but with two major differences

1. Every level must be completely filled
2. All the leaf elements must lean towards the left.
3. The last leaf element might not have a right sibling i.e. a complete binary tree doesn't have to be a full binary tree.



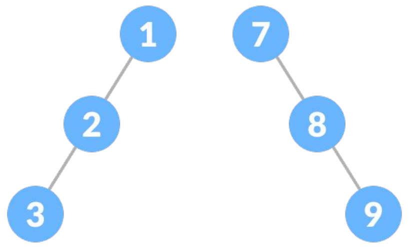
Degenerate or Pathological Tree

A degenerate or pathological tree is the tree having a single child either left or right.



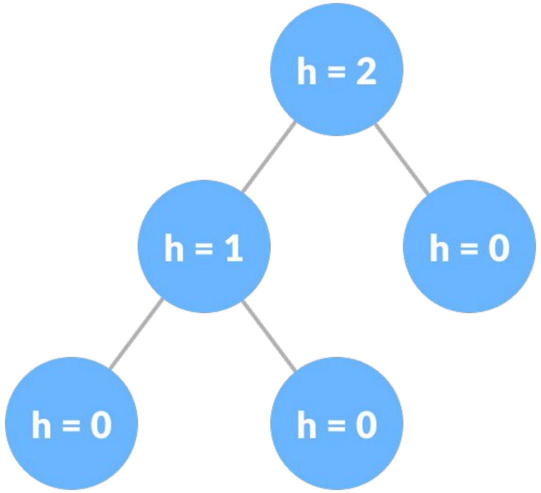
Skewed Binary Tree

A skewed binary tree is a pathological/degenerate tree in which the tree is either dominated by the left nodes or the right nodes. Thus, there are two types of skewed binary tree: left-skewed binary tree and right-skewed binary tree.



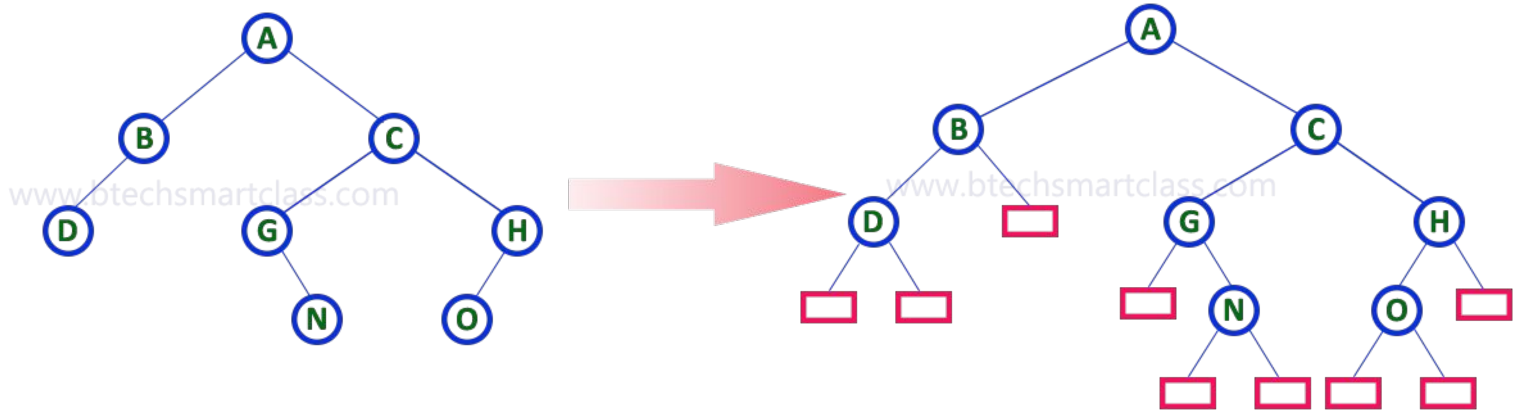
Balanced Binary Tree

It is a type of binary tree in which the height difference between the left and the right subtree for each node is either 0 or 1.



Extended Binary Tree

A binary tree can be converted into Full Binary tree by adding dummy nodes to existing nodes wherever required. The full binary tree obtained by adding dummy nodes to a binary tree is called as Extended Binary Tree.


A
B
C
D
G
H
N
O
www.btechsmartclass.com
A
B
C
D
G
H
N
O
www.btechsmartclass.com