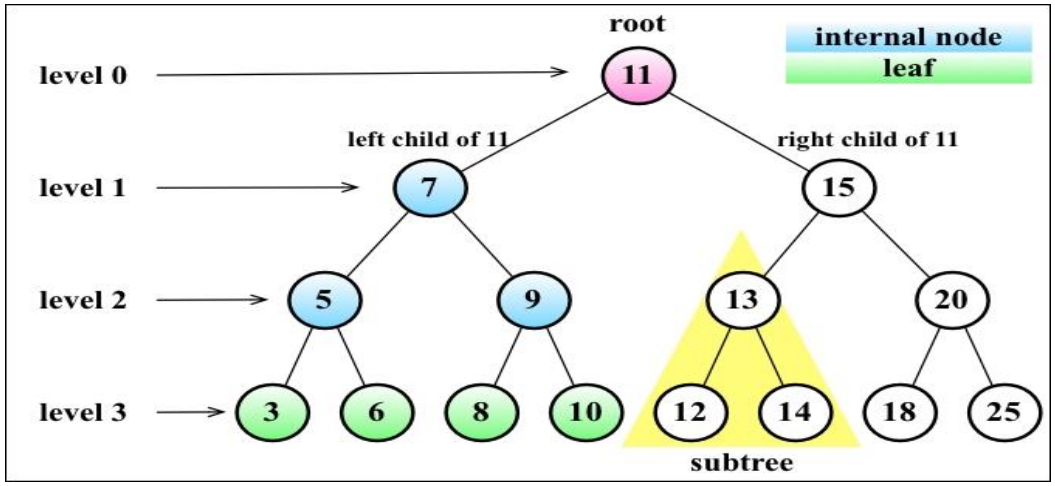
Tree Data Structure 2

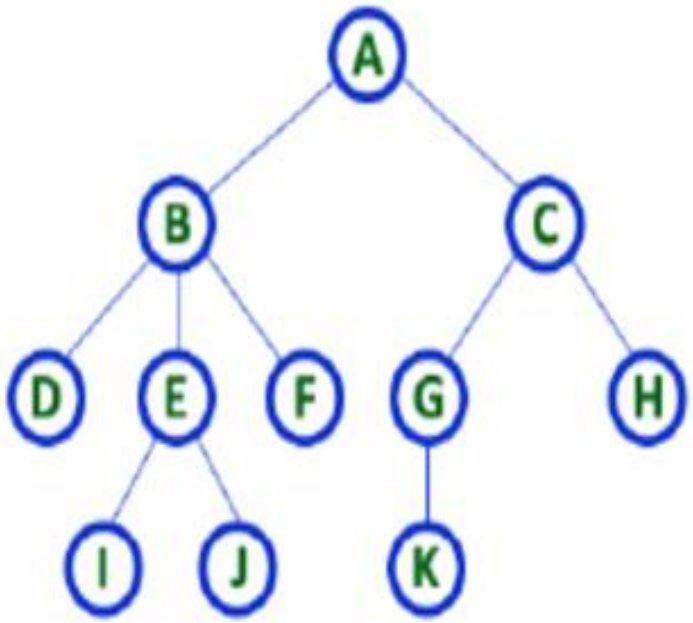


by

Dr. Nadia M. Mohammed

Tree Representation

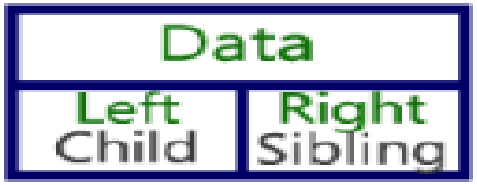
- A tree data structure can be represented as **Left Child - Right Sibling Representation.**
- Consider the following tree:



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'

- Graphical representation of that node is as follows:



- In this representation, we use list with one type of node which consists of three fields namely:

1- **Data field**: stores the actual value of a node.

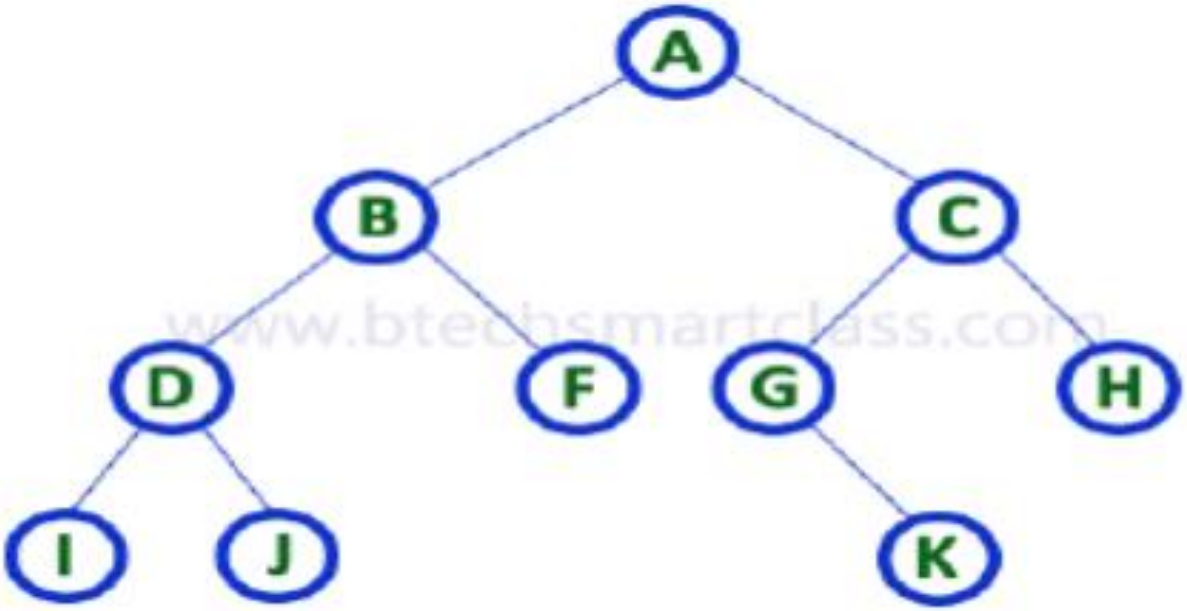
2- **Left child reference field**: left reference field stores the address of the left child.

3- **Right sibling reference field**: stores the address of the right sibling node.

Binary Tree

- In a normal tree, every node can have any number of children.
- **Binary tree :** is a special type of tree data structure in which every node can have a **maximum of 2 children**. One is known as left child and the other is known as right child.
- **In a binary tree, every node can have either zero children or one child or two children but not more than 2 children.**

Example



Binary Tree Types

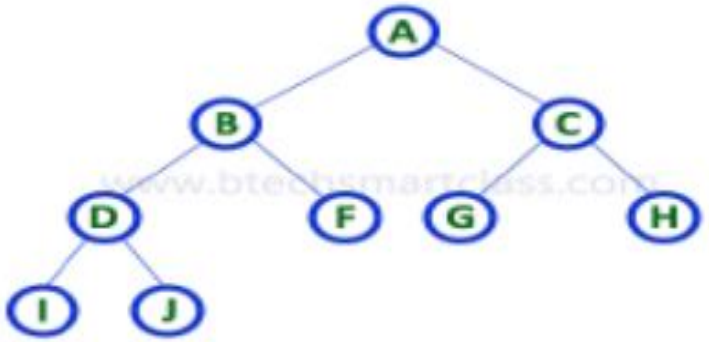
There are different types of binary trees and they are:

1. Strictly Binary Tree.

2. Complete Binary Tree.

1. Strictly Binary Tree

A binary tree in which every node must have exactly two children or zero number of children.

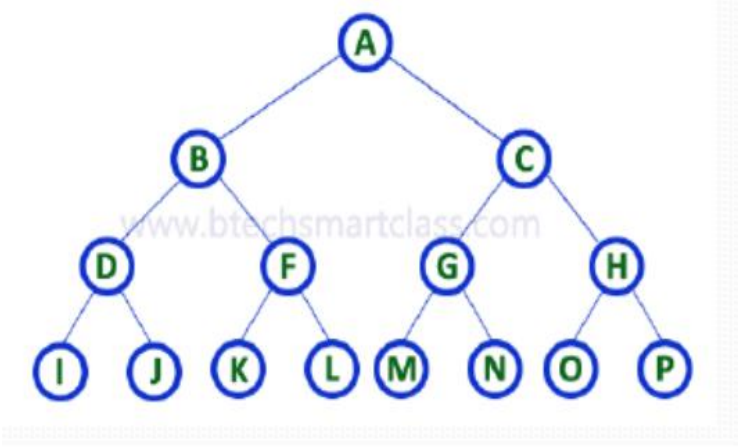


- Strictly binary tree is also called as **Full Binary Tree.**
- Strictly binary tree data structure is used to represent mathematical expressions

Example

2. Complete Binary Tree

A binary tree in which every internal node has exactly two children and all leaf nodes are at same level.



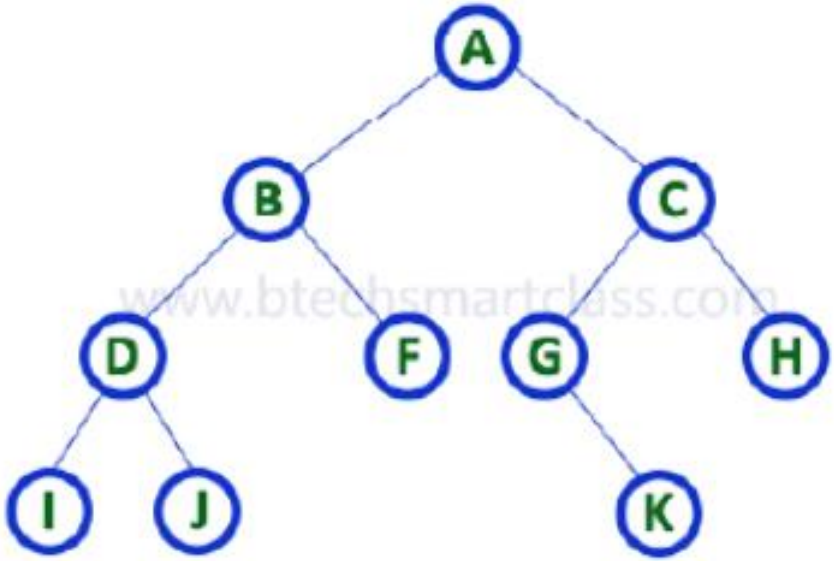
- In complete binary tree there must be 2^{level} number of nodes. For example at level 2 there must be $2^2 = 4$ nodes and at level 3 there must be $2^3 = 8$ nodes.

Binary Tree Representations

A binary tree data structure is represented using two methods. Those methods are as follows:

1. **Array** Representation.
2. **Linked List** Representation.

Consider the following binary tree:



1. Array Representation

- In array representation of binary tree, we use a one dimensional array (1-D Array) to represent a binary tree.
- Consider the above example of binary tree and it is represented as follows:

A	B	C	D	F	G	H	I	J	-	-	-	K	-	-

- To represent a binary tree of depth **'n'** using array representation, we need one dimensional array with a maximum size of $\mathbf{2^{n+1} - 1}$.

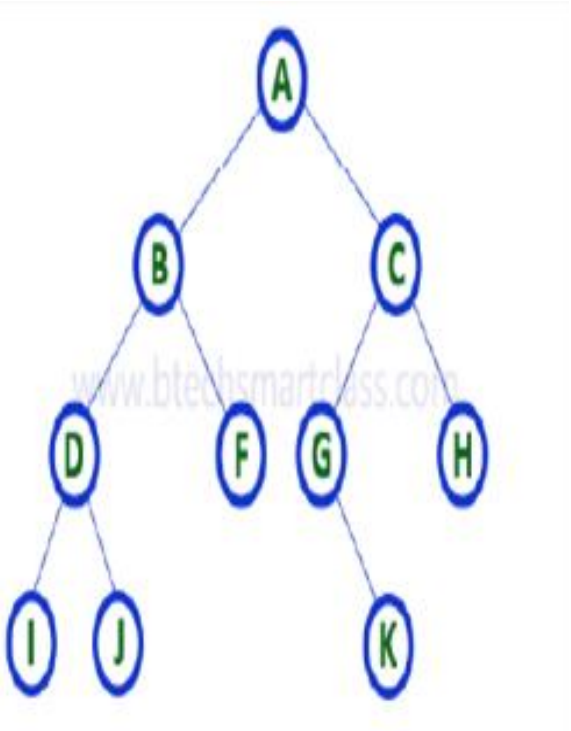
2. Linked List Representation

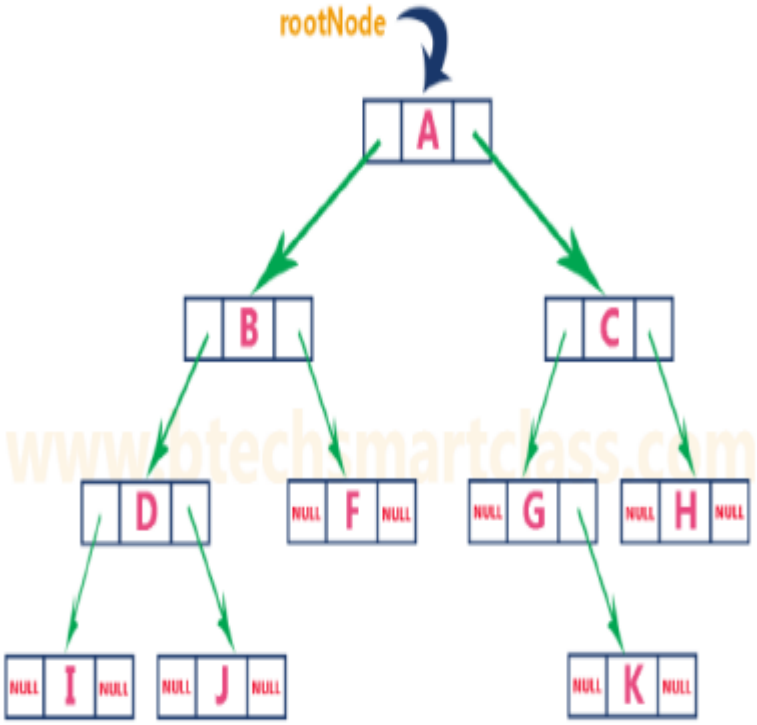
We use **double linked list** to represent a binary tree. In a double linked list, every node consists of three fields:

- First field for storing left child address.
- Second field for storing actual data.
- Third field for storing right child address.

In this linked list representation, a node has the following structure:

The above example of binary tree represented using Linked list representation is shown as follows:





Binary Tree Traversals

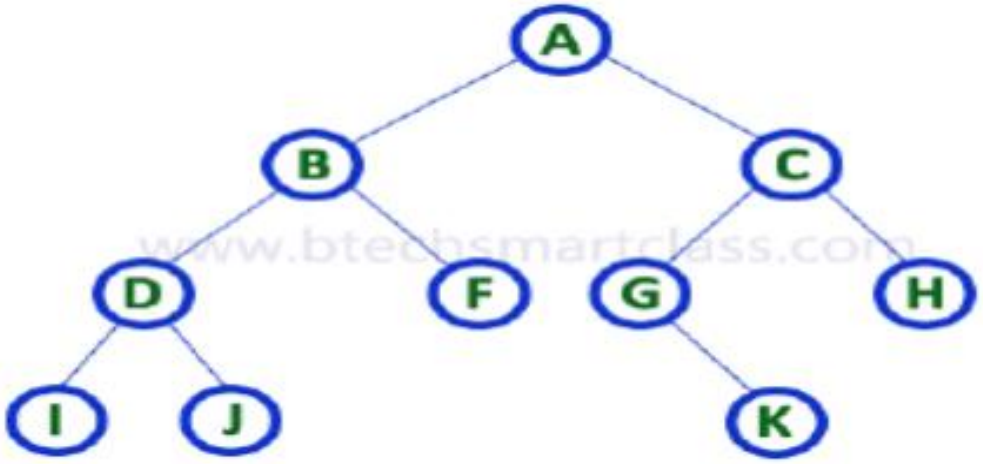
Binary Tree Traversal: Displaying (or) visiting order of nodes in a binary tree is called as Binary Tree Traversal. There are three types of binary tree traversals.

1. **In - Order Traversal.**
2. **Pre - Order Traversal.**
3. **Post - Order Traversal.**

1. In-Order Traversal (left Child-root–right Child)

- In In-Order traversal, the root node is visited between left child and right child. In this traversal, the left child node is visited first, then the root node is visited and later we go for visiting right child node.
- In-Order Traversal for example of binary tree is:

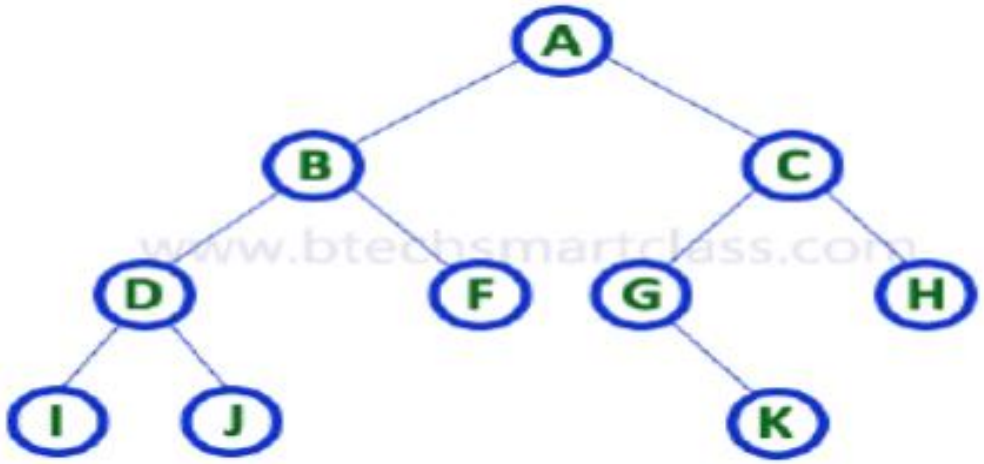
I - D - J - B - F - A - G - K - C - H



2. Pre-Order Traversal (root–left Child–right Child)

- In Pre-Order traversal, the root node is visited first, then its left child and later its right child.
- Pre-Order Traversal for example binary tree is:

A - B - D - I - J - F - C - G - K - H



3. Post-Order Traversal (left Child–right Child- root)

- In this traversal, left child node is visited first, then its right child and then its root node.
- Post-Order Traversal for above example binary tree is:

I - J - D - F - B - K - G - H - C – A