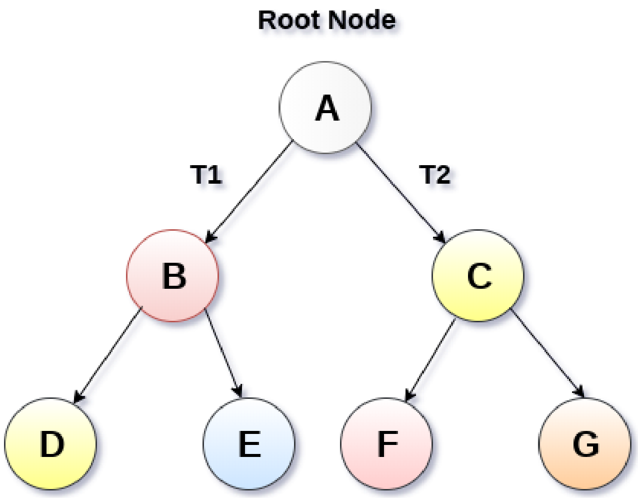
# Binary Tree

Binary Tree is a special type of generic tree in which, each node can have at most two children. Binary tree is generally partitioned into three disjoint subsets.

1. Root of the node
2. left sub-tree which is also a binary tree.
3. Right binary sub-tree

A binary Tree is shown in the following image.



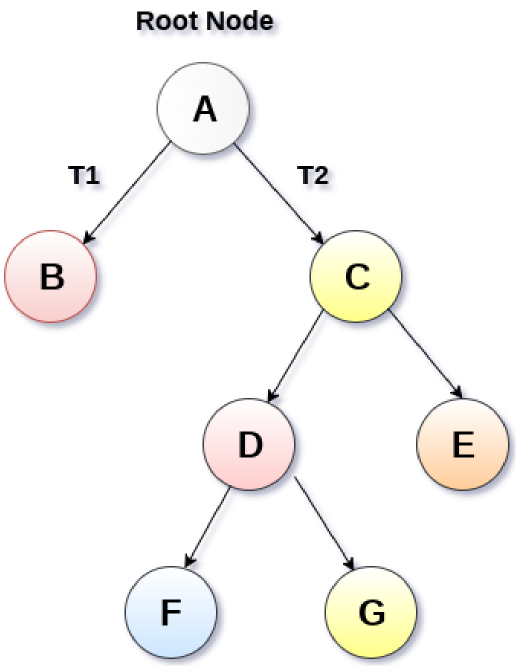
Binary Tree

## Types of Binary Tree

### 1. Strictly Binary Tree

In Strictly Binary Tree, every non-leaf node contain non-empty left and right sub-trees. In other words, the degree of every non-leaf node will always be 2. A strictly binary tree with n leaves, will have (2n - 1) nodes.

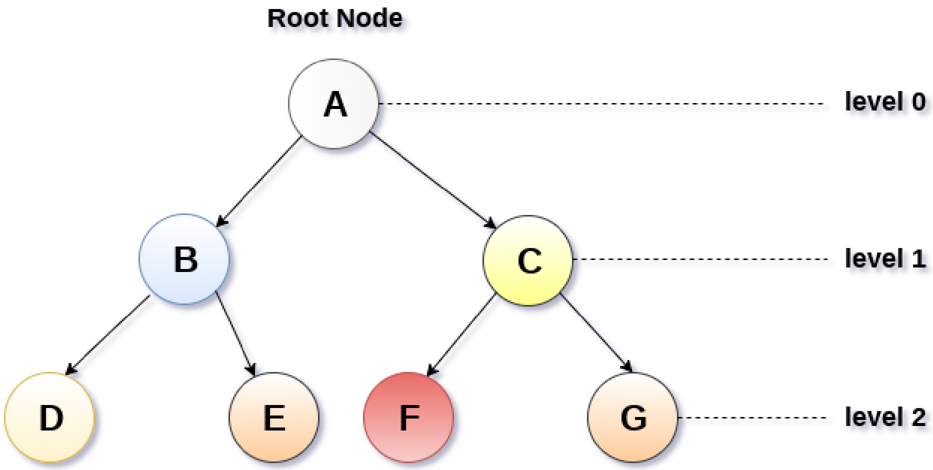
A strictly binary tree is shown in the following figure.



**Strictly Binary Tree**

## 2. Complete Binary Tree

A Binary Tree is said to be a complete binary tree if all of the leaves are located at the same level d. A complete binary tree is a binary tree that contains exactly 2^l nodes at each level between level 0 and d. The total number of nodes in a complete binary tree with depth d is $2^{d+1}$-1 where leaf nodes are $2^{d}$ while non-leaf nodes are $2^{d}$-1.



**Complete Binary Tree**

## Binary Tree Traversal

| SN | Traversal | Description |
|---|---|---|
| 1 | Pre-order Traversal | Traverse the root first then traverse into the left sub-tree and right sub-tree respectively. This procedure will be applied to each sub-tree of the tree recursively. |
| 2 | In-order Traversal | Traverse the left sub-tree first, and then traverse the root and the right sub-tree respectively. This procedure will be applied to each sub-tree of the tree recursively. |
| 3 | Post-order Traversal | Traverse the left sub-tree and then traverse the right sub-tree and root respectively. This procedure will be applied to each sub-tree of the tree recursively. |

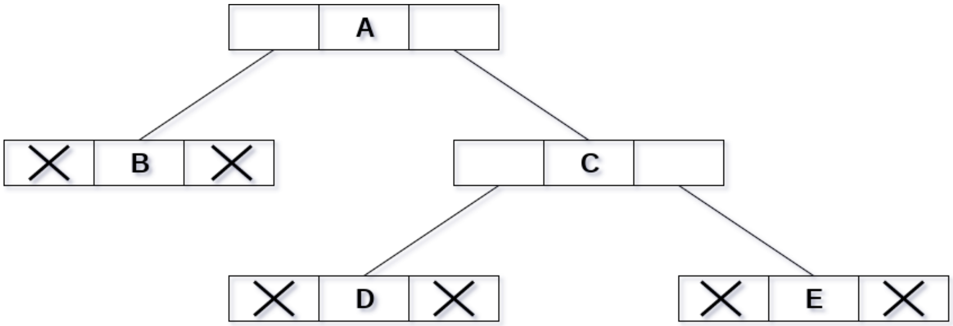
## Binary Tree representation

There are two types of representation of a binary tree:

### 1. Linked Representation

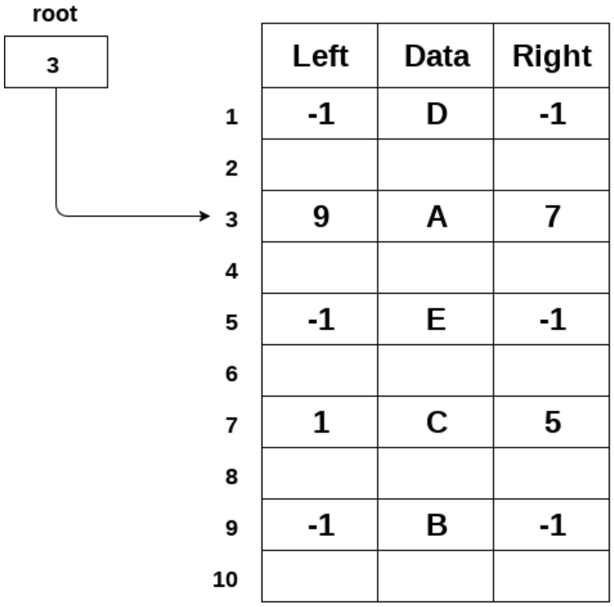
In this representation, the binary tree is stored in the memory, in the form of a linked list where the number of nodes are stored at non-contiguous memory locations and linked together by inheriting parent child relationship like a tree. every node contains three parts : pointer to the left node, data element and pointer to the right node. Each binary tree has a root pointer which points to the root node of the binary tree. In an empty binary tree, the root pointer will point to null.

Consider the binary tree given in the figure below.



In the above figure, a tree is seen as the collection of nodes where each node contains three parts : left pointer, data element and right pointer. Left pointer stores the address of the left child while the right pointer stores the address of the right child. The leaf node contains **null** in its left and right pointers.

The following image shows about how the memory will be allocated for the binary tree by using linked representation. There is a special pointer maintained in the memory which points to the root node of the tree. Every node in the tree contains the address of its left and right child. Leaf node contains null in its left and right pointers.

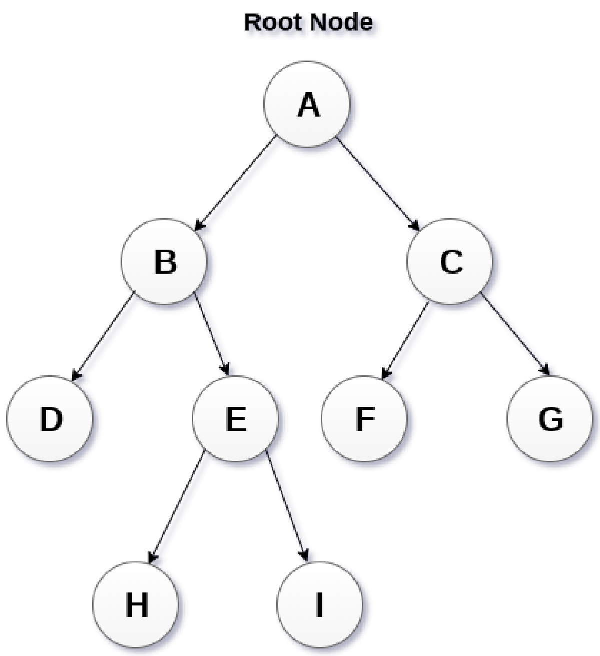


| | Left | Data | Right |
|---|---|---|---|
| 1 | -1 | D | -1 |
| 2 | | | |
| 3 | 9 | A | 7 |
| 4 | | | |
| 5 | -1 | E | -1 |
| 6 | | | |
| 7 | 1 | C | 5 |
| 8 | | | |
| 9 | -1 | B | -1 |
| 10 | | | |

**Memory Allocation of Binary Tree using linked Representation**

## 2. Sequential Representation

This is the simplest memory allocation technique to store the tree elements but it is an inefficient technique since it requires a lot of space to store the tree elements. A binary tree is shown in the following figure along with its memory allocation.



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | | | H | I |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

Sequential Representation of Binary Tree

In this representation, an array is used to store the tree elements. Size of the array will be equal to the number of nodes present in the tree. The root node of the tree will be present at the 1st index of the array. If a node is stored at ith index then its left and right children will be stored at 2i and 2i+1 location. If the 1st index of the array i.e. tree[1] is 0, it means that the tree is empty.

← prev next →

Please Share

## Learn Latest Tutorials

- C. Graphics
- Automata
- Testing
- NumPy
- Verbal A.
- AWS

## Preparation

- Aptitude
- Reasoning
- Verbal A.
- Interview

## B.Tech / MCA

- DBMS
- DS
- DAA
- OS
- C. Network
- Compiler D.
- COA
- D. Math.
- Web Tech.
- Cyber Sec.
- C
- C++
- Java
- .Net
- Python
- Programs
- Control S.