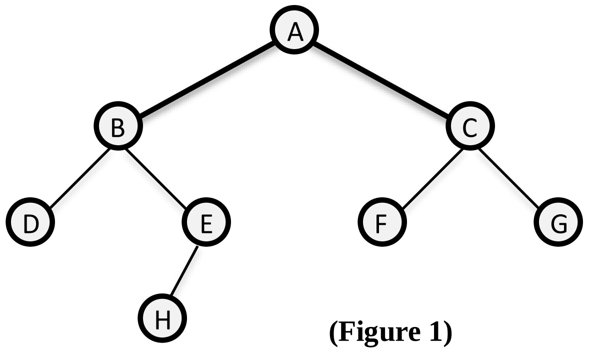
Tree

A tree can be defined recursively as a finite set of nodes such that

i. A single node, called root, itself is a tree.
ii. If there are k trees as T_1, T_2,………,T_k with their roots r_1,r_2,……r_k, and if r is a node then a new tree can be created by making r as a root node and parent of r_1,r_2,……r_k. In this tree T_1, T_2,………,T_k are subtrees and r_1,r_2,……r_k are the children of node r.

A tree is represented pictorially in the form of circles and lines (arcs) as seen below:



(Figure 1)

In the above figure 1, the nodes are shown in circles and these nodes are connected through lines or arcs. Node A is the **root** of a tree. Node A is called the **parent** or **father** of node B and node B is the **son** or **child** of node A. Nodes B and C are children of node A. Similarly, node C is father of nodes F and G and the nodes F and G are children of node C. Nodes B and C are siblings as both the nodes have a common parent.
Nodes B, C and E are **internal nodes** as they have a child or children. Nodes D, H, F and G are **external nodes** also called **terminal nodes** as they have no successor.

The **level** of a node is defined such as:

i. The root of a tree is at level 0.
ii. Level of any node is one more than the level of its parent.

In the figure 1, the following observations are made:
The root A is at level 0.
The nodes B and C are at level 1.
The nodes D, E, F and G are at level 2.
The node H is at level 3.

The depth or height of a tree is one more than the largest level. In other words, the depth of a tree is the number of nodes in the longest branch of that tree.
The depth of a tree in figure 1 is 3+1 = 4 (1 more than the largest level, which is 3).

Binary Tree

A binary tree is an abstract data type, non-primitive and non-linear data structure consisting of a finite set of nodes such as:

i. **It is either empty or**
ii. **It consists of a node called the root node with, at the most, two disjoint binary trees connected to it called the left subtree and the right subtree.**

Properties of Binary Trees

The properties of a binary tree are mentioned below:

1) The Maximum Number of Nodes at the Level l of a Binary Tree is 2^l.

The statement can be proved by the induction method.

The level is the number of nodes on the path from the root to the node.

For root, l = 0, the number of nodes = $2^0 = 1$

Assuming that maximum number of nodes on level l-1 is 2^{l-1}

Since in a binary tree, every node has at the most 2 children, the next level will have twice nodes, i.e., $2 * 2^{l-1} = 2^l$

2) Maximum Number of Nodes in a Binary Tree of Height h is $2^h – 1$.

The height or depth of a tree is the maximum number of nodes on the root to the leaf path. The height of a leaf node is 1.

If all the levels have no space left then a tree will have the maximum no. of nodes.

Therefore, maximum number of nodes in a binary tree of height h is $1+2+4+ \ldots\ldots\ldots\ldots +2^{h-1}$.

This is a geometric series with h terms and the sum of this series is $2^h – 1$.

3) In a Binary Tree with n Nodes, Minimum Possible Height or Minimum Number of Levels is Floor($\log_2(n+1)$).

For example, if a binary tree has 7 nodes and all the levels have the maximum nodes then the minimum height of a binary tree will be $\lceil \log_2(7+1) \rceil = \lceil \log_2(2^3) \rceil = \lceil 3 \times \log_2 2 \rceil = 3 \times 1 = 3$.

4) A Binary Tree with n Leaves has at least Floor($\log_2 n$)+1 Levels.

For example, if a binary tree has 10 nodes then it will have at least $\lceil \log_2 10 \rceil+1 = \lceil 3.3219 \rceil+1 = 4$ levels

5) In a Binary Tree, Number of Leaf Nodes is Always One More than the Number of Internal Nodes with Two Children.

L = T + 1 where, L = Number of leaf nodes

T = Number of internal nodes with two children

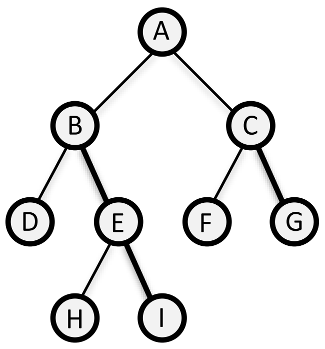
If a binary tree has 7 internal nodes with two children each then the total number of leaf nodes will be 8.

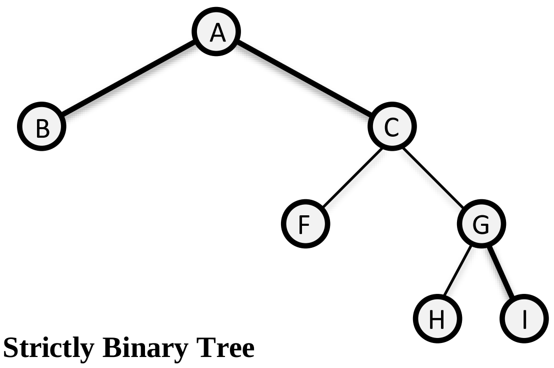
Various types of binary trees are as follows:

Strictly Binary Tree

A binary tree is called a strictly binary tree if every node has 0 or 2 children. Here, every non-leaf node in a binary tree has nonempty left and right subtrees. All of the nodes in a strictly binary tree are of degree zero or two, never degree one. A strictly binary tree with n leaves always contains 2n – 1 nodes.

Some examples of strictly binary trees are as follows:



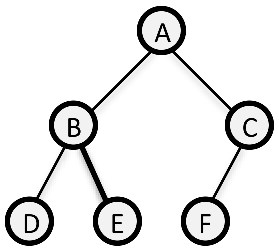


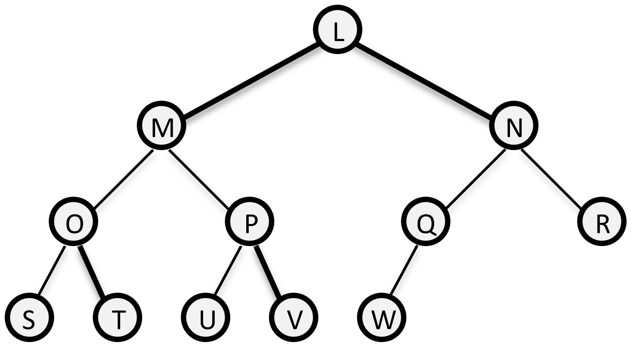
Strictly Binary Tree

Complete Binary Tree

A binary tree is known as a complete binary tree if it's all levels are completely filled except possibly the last level where all the nodes are stored as left as possible.

Some examples of complete binary trees are as follows:

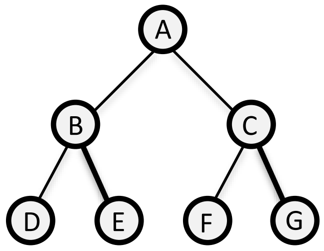


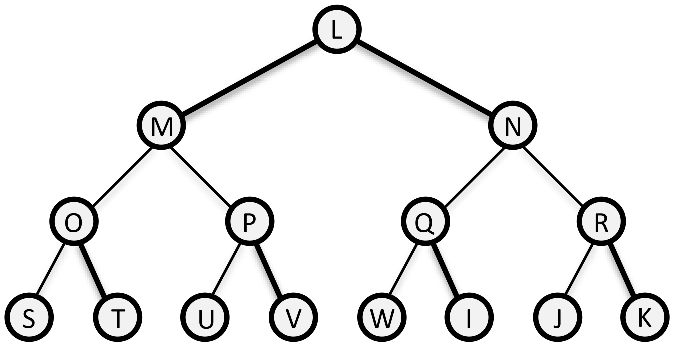


Complete Binary Trees

Perfect Binary Tree

It is a binary tree in which all internal nodes have two children and all the leaves are at the same level. Some examples of perfect binary trees are as follows:

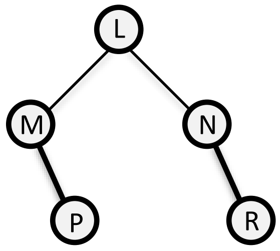


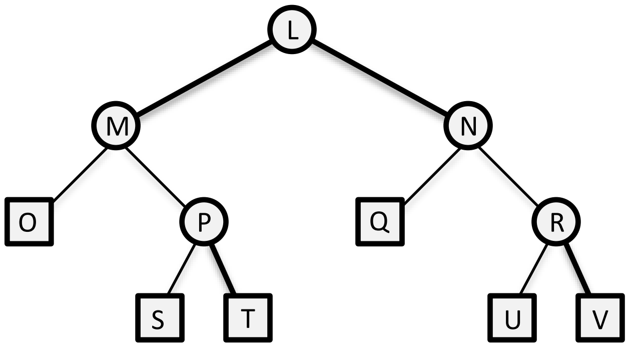


Fully Binary Trees

Extended Binary Tree (Two-tree)

An extended binary tree is a special form of binary tree. A binary tree is an extended binary tree if it has strictly either zero or two children. Any binary tree can be promoted to an extended binary tree by inserting new nodes as children to the nodes with null values in their address fields. The new nodes can be represented by rectangles.





Binary Tree **Extended Binary Tree**

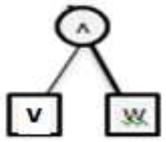
Binary Tree Representation of Algebraic Expression (Expression Tree)

Any algebraic expression can be represented in a binary tree form. All operators involved in this expression, for example, {+, -, *, /, ^, %} have some priority in making the calculations. In this expression tree, the operands are represented by the external nodes and the operators are represented by the internal nodes of an extended binary tree.

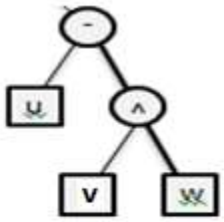
Let us consider an algebraic expression:

(x + y) – (z / (u – v ^ w))

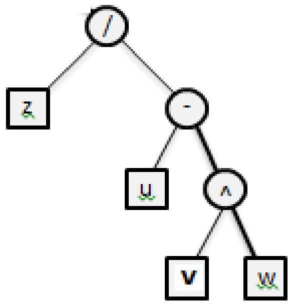
1.



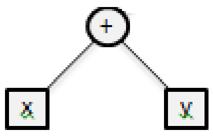
2.



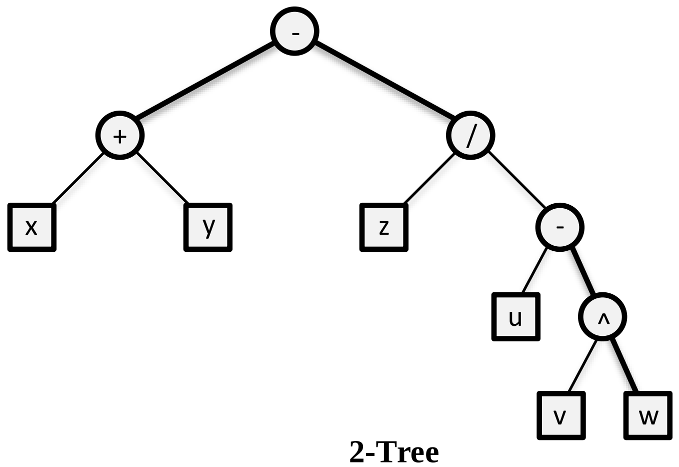
3.



4.



5.



2-Tree

Here, x, y, z, u, v, w are the operands and will be the external nodes. The operators +, -, /, ^ are the operators and will be the internal nodes.
(Note: If the nodes of such a tree are visited in preorder, inorder and postorder then prefix, infix and postfix expressions are obtained**)**

Draw the expression tree of the following expression:

(a * (b / c)) + (d + e) ^ f