Sunbeam Institute of Information Technology
Pune and Karad
PreCAT

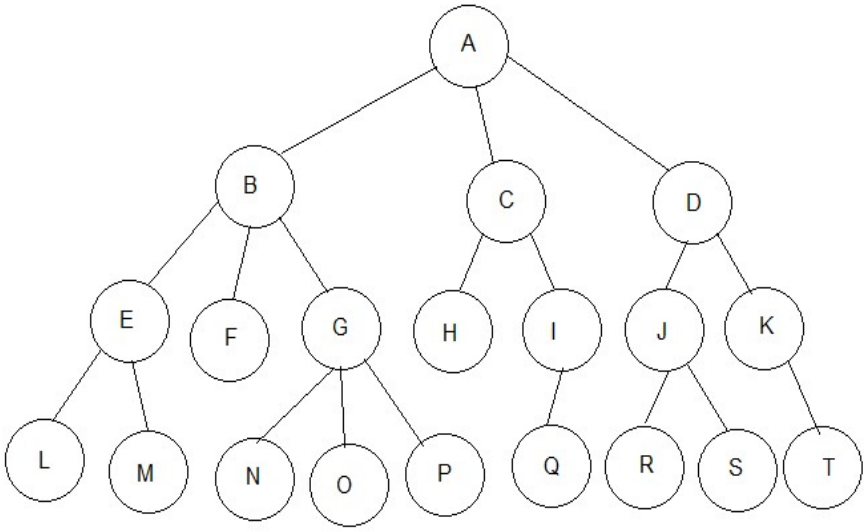
Module – Data Structures

Trainer - Devendra Dhande

Email – devendra.dhande@sunbeaminfo.com

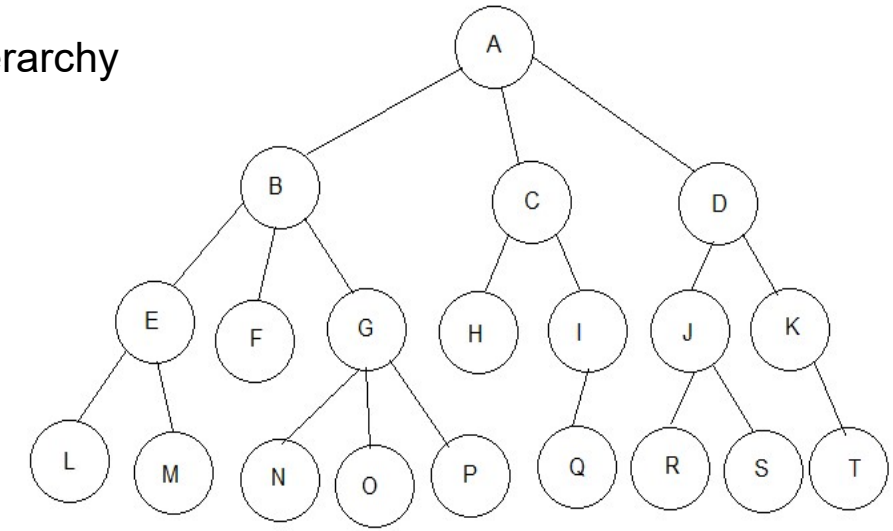
Tree : Terminologies

- **Tree** is a **non linear** data structure in which one specially designated node is called as "**root**".
- **Root** is a **starting point** of the tree.
- Remaining elements can be partitioned into m disjoint subsets where each of subset is a tree.
- All elements are connected in **Hierarchical manner**.
- Every element of a tree is called as **node** of the tree.
- **Parent node:-** having other child nodes connected
- **Child node:-** immediate descendant of a node
- **Leaf node:-**
  - Terminal node of the tree.
  - Leaf node does not have child nodes.
- **Ancestors:-** all nodes in the path from root to that node.
- **Descendants:-** all nodes accessible from the given node
- **Siblings:-** child nodes of the same parent



Tree : Terminologies

- **Degree of a node :-** number of child nodes for any given node.
- **Degree of a tree :-** Maximum degree of a child in tree.
- **Level of a node :-** indicates position of the node in tree hierarchy
  - Level of child = Level of parent + 1
  - Level of root = 1
- **Height of node :-** level of given node
- **Depth of node :-** level of node - 1
- **Height of a tree :-** Maximum height of a node
- **Depth of a tree :-** Maximum depth of a node
- Tree with zero nodes (ie empty tree) is called as

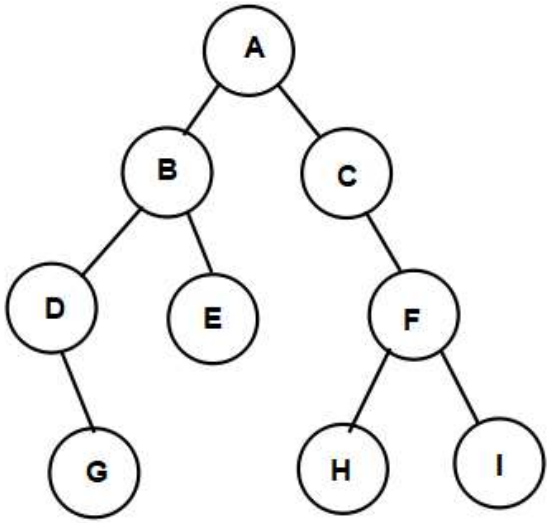


"**Null tree**". Height of Null tree is 0.

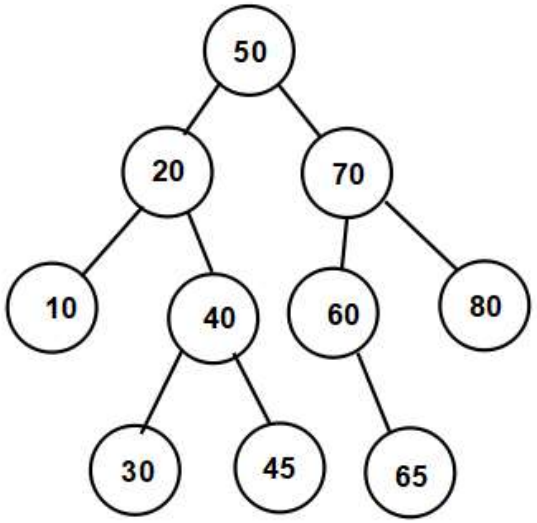
- Tree can grow up to any level and any node can have any number of Childs.
- That's why operations on tree becomes un efficient.
- Restrictions can be applied on it to achieve efficiency and hence there are different types of trees.

Tree : Types

- **Binary Tree**
  - Tree in which each node has maximum two child nodes
  - Binary tree has degree 2. Hence it is also called as 2- tree



- **Binary Search Tree**
  - Binary tree in which left child node is always smaller and right child node is always greater or equal to the parent node.
  - Searching is faster
  - Time complexity : O(h) h – height of tree



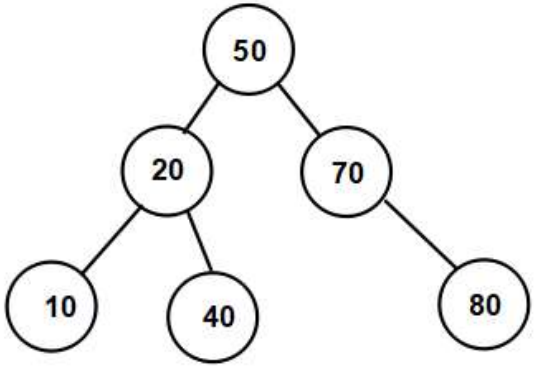
Tree : Types

- **Complete Binary Tree**
  - Binary tree in which all leaf nodes are at same level.

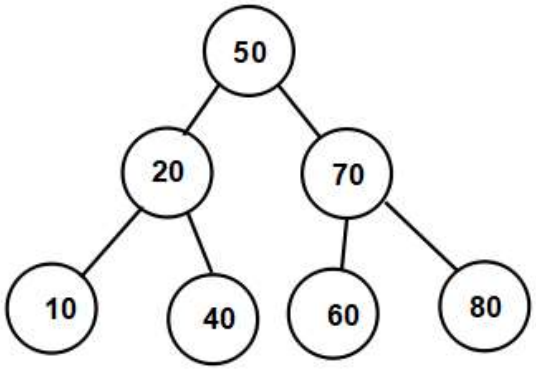
- **Strictly Binary Tree**
  - Binary tree in which each non leaf node has exact two child nodes.

- **Full Binary Tree**
  - Binary tree with its full capacity foe the given height.
  - In other words, adding one more node will increase height of the tree.
  - It is always complete as well as strictly binary tree.
  - Number of elements = 2^h – 1

- **Almost Complete Binary Tree**
  - The binary tree which follows two conditions
    - All leaf nodes are at level h or h-1.
    - All leaf nodes at last level (h) are aligned to left as much as possible.



Complete Binary Tree

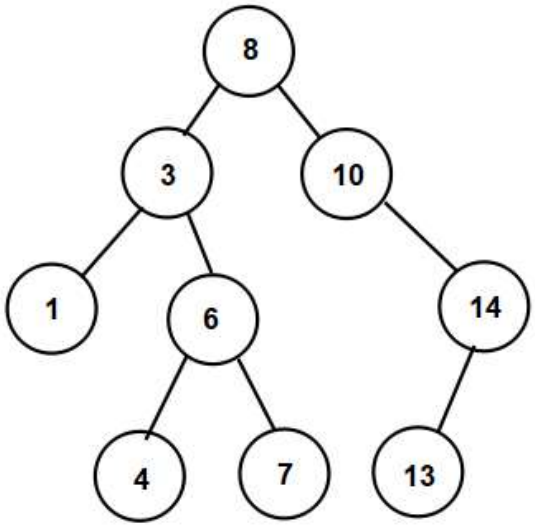


Strictly Binary Tree

Binary Search Tree : Traversal

- **Pre-Order:-** V L R
- **In-order:-** L V R
- **Post-Order:-** L R V
- The traversal algorithms can be implemented easily using recursion.
- Non-recursive algorithms for implementing traversal needs stack to store node pointers.

- **Pre-Order :-** 8 3 1 6 4 7 10 14 13
- **In-Order :-** 1 3 4 6 7 8 10 13 14
- **Post-Order :-** 1 4 7 6 3 13 14 10 8

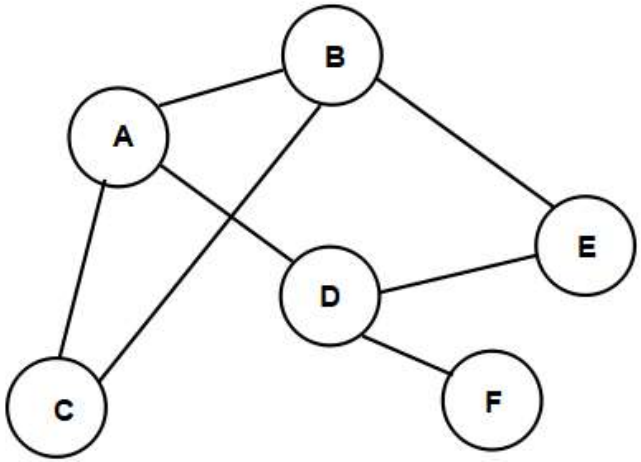


BST - Types

- **Skewed Binary tree**
  - In binary tree if only left or right links are used, then tree grows in only one direction and such tree is called as skewed binary tree.
    - Left skewed binary tree
    - Right skewed binary tree
  - Tree has maximum height that is same as number of elements
  - Time complexity of searching is O(n). (Like linked list)
- **Balanced BST**
  - If nodes in BST are arranged so that its height is kept as less as possible, is called as balanced BST.
  - Balance Factor = Height of left subtree – Height of right subtree
  - In balanced BST, balance factor of each node is -1, 0 or +1
  - A tree can be balanced by applying series of left or right rotations.
- **AVL Tree**
  - AVL tree is a self balancing binary search tree.
  - Node are balanced on each insert and delete operation.
  - Difference between heights of left and right subtrees can not be more than 1 for all nodes.

Graph : Terminologies

- **Graph** is a non linear data structure having set of vertices (nodes) and set of edges (arcs).
  - G = {V, E}
    Where V is a set of vertices and E is a set of edges
  - **Vertex (node)** is an element in the graph
    V = {A, B, C, D, E, F}
  - **Edge (arc)** is a line connecting two vertices
    E = {(A,B), (A,C), (B,C), (B,E), (D, E), (D,F),(A,D)}



- Vertex A is set be **adjacent** to B, if and only if there is an edge from A to B.
- **Degree of vertex :-** Number of vertices adjacent to given vertex
- **Path :-** Set of edges connecting any two vertices is called as path between those two vertices.
  - Path between A to D = {(A, B), (B, E), (E, D)}
- **Cycle :-** Set of edges connecting to a node itself is called as cycle.
  - {(A, B), (B, E), (E, D), (D, A)}
- **Loop :-** An edge connecting a node to itself is called as loop. Loop is smallest cycle.

Graph : Types

- **Simple Graph**
  - Graph not having multiple edges between adjacent nodes and no loops.
- **Complete Graph**
  - Simple graph in which node is adjacent with every other node.
  - Number of Edges = n (n -1) / 2 where, n – number of vertices
- **Connected Graph**
  - Simple graph in which there is some path exist between any two vertices
- **Weighted Graph**
  - A graph in which edge is associated with a number (ie weight)
- **Directed Graph (Di-graph)**
  - A graph in which each edge has some direction

Thank you!

Devendra Dhande
<devendra.dhande@sunbeaminfo.com>