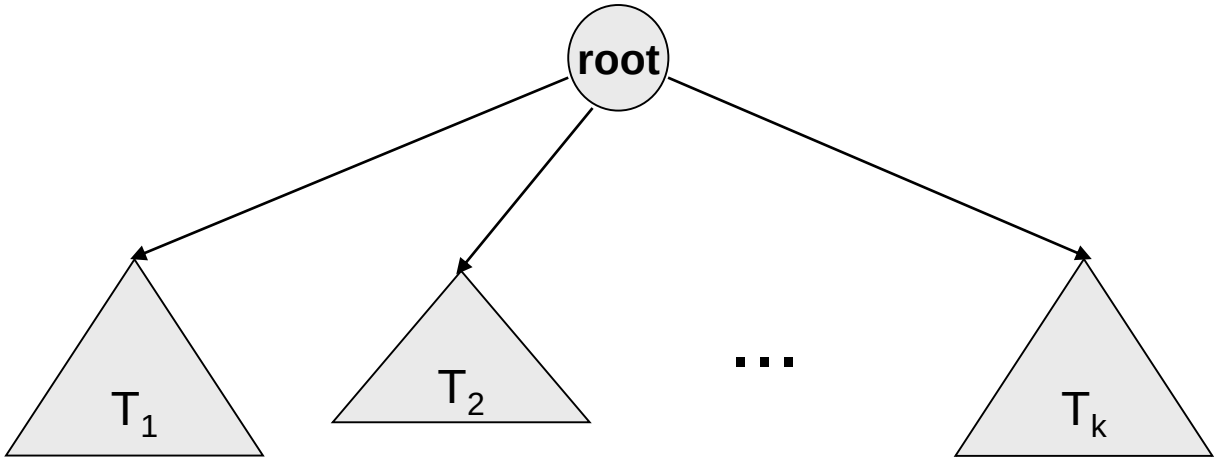
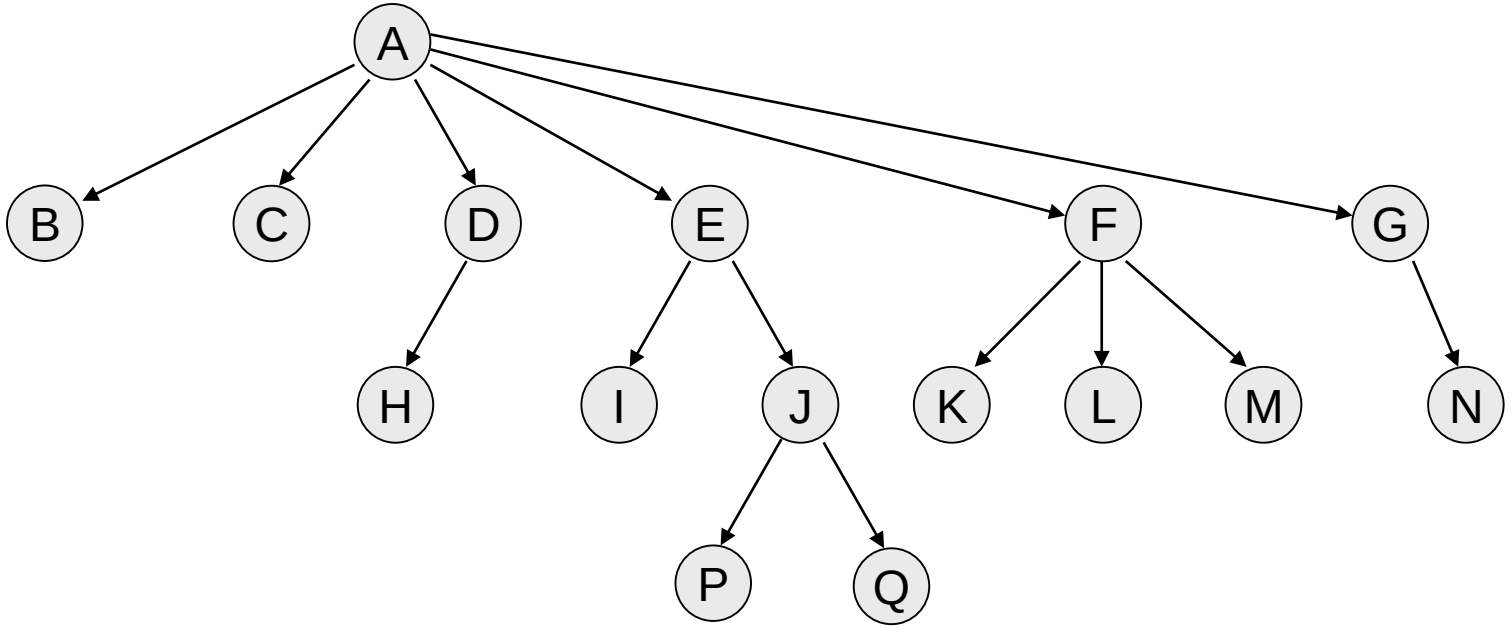
# Trees

# What is a Tree?

- A tree is a collection of nodes with the following properties:
  - The collection can be empty and the tree is called a null tree.
  - Otherwise, a tree consists of a distinguished node r, called *root*, and the remaining nodes are partitioned into k subsets representing subtrees $T_1$, $T_2$, … , $T_k$, each of whose roots are connected by a *directed edge* from r.
- The root of each sub-tree is said to be *child* of r, and r is the *parent* of each sub-tree root.
- If a tree is a collection of N nodes, then it has N-1 edges.



# Preliminaries



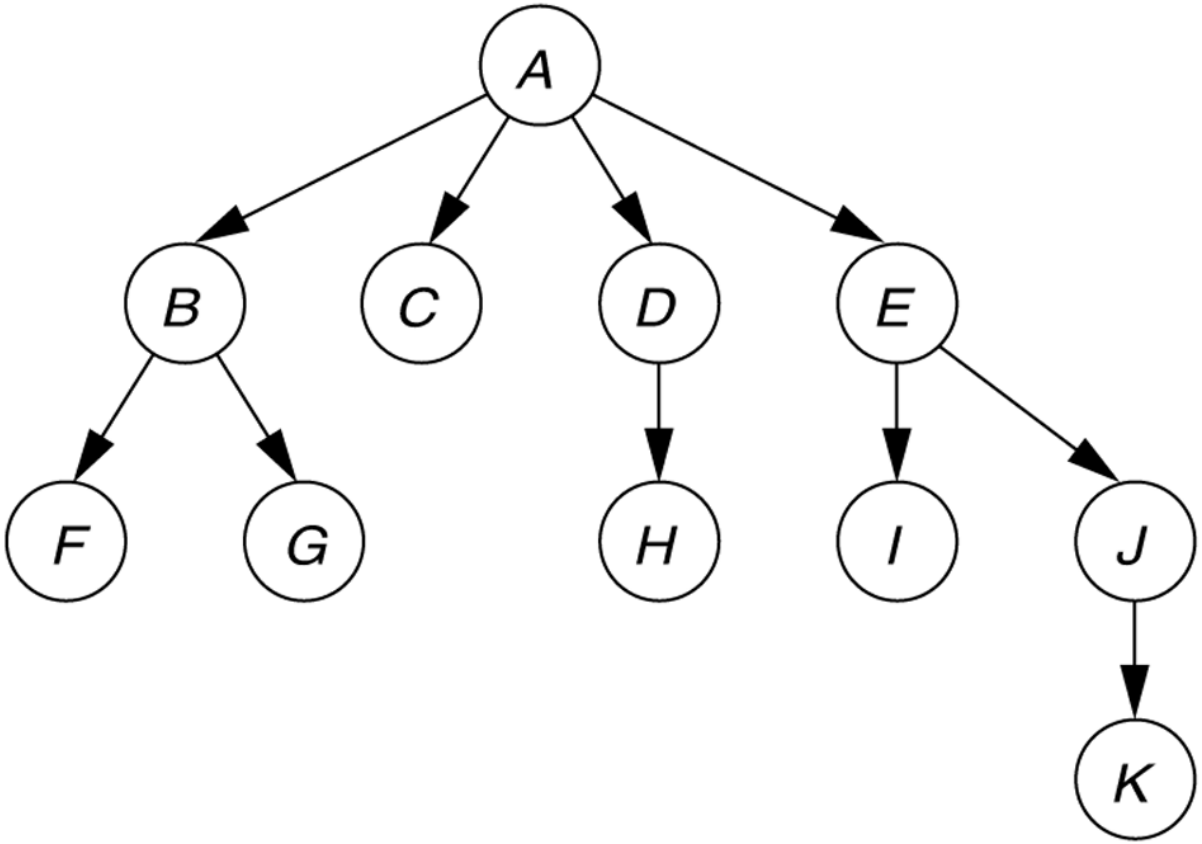

- Node *A* has 6 *children*: B, C, D, E, F, G.
- B, C, H, I, P, Q, K, L, M, N are *leaves* in the tree above.
- K, L, M are *siblings* since F is parent of all of them.

# Preliminaries (continued)

- A ***path*** from node $n_1$ to $n_k$ is defined as a sequence of nodes $n_1, n_2, \ldots, n_k$ such that $n_i$ is parent of $n_{i+1}$ $(1 \leq i < k)$
  - The ***length*** of a path is the number of edges on that path.
  - There is a path of length zero from every node to itself.
  - There is exactly one path from the root to each node.
- The ***depth*** of node $n_i$ is the length of the path from *root* to node $n_i$
- The ***height*** of node $n_i$ is the length of longest path from node $n_i$ to a *leaf*.
- If there is a path from $n_1$ to $n_2$, then $n_1$ is ***ancestor*** of $n_2$, and $n_2$ is ***descendent*** of $n_1$.
  - If $n_1 \neq n_2$ then $n_1$ is *proper ancestor* of $n_2$, and $n_2$ is *proper descendent* of $n_1$.

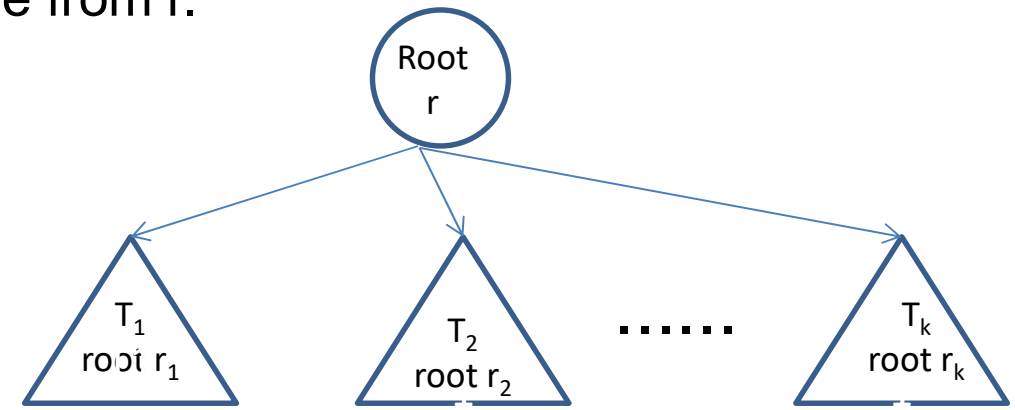
# Figure 1

A tree, with height and depth information




| Node | Height | Depth |
|---|---|---|
| *A* | 3 | 0 |
| *B* | 1 | 1 |
| *C* | 0 | 1 |
| *D* | 1 | 1 |
| *E* | 2 | 1 |
| *F* | 0 | 2 |
| *G* | 0 | 2 |
| *H* | 0 | 2 |
| *I* | 0 | 2 |
| *J* | 1 | 2 |
| *K* | 0 | 3 |

# Lecture – 04 - Tree

- A tree T is a collection of n nodes.
  - If the collection is empty, then the tree is called a null tree.
  - One of the nodes, r is specially designated as root of the tree
  - The remaining nodes are partitioned in to k subsets, $T_1$, $T_2$, ..., $T_k$
  - Each subset represents a tree with $r_1$, $r_2$, ..., $r_k$ as roots.
  - Each of these roots $r_1$, $r_2$, ..., $r_k$ is connected through a directed edge from r.
- 

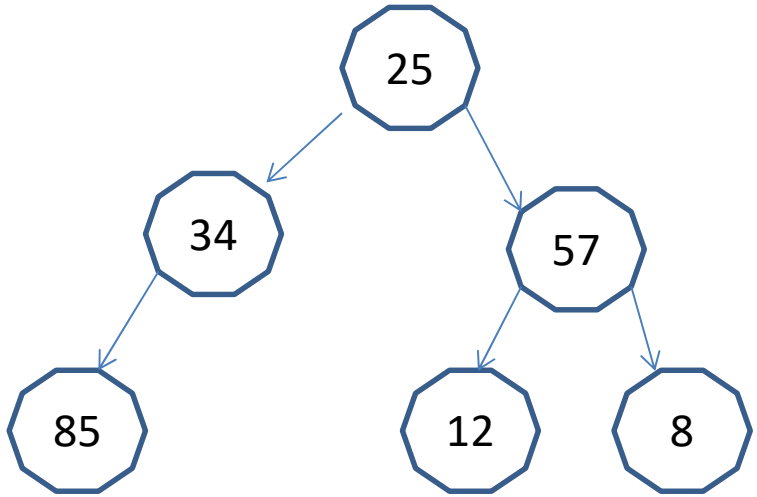
# Binary Tree

- A node in a tree is connected to any number of nodes. These nodes are called child nodes and the node connecting them is called parent.
- A tree can be represented through a structure called First-child Next-sibling.
- This kind of a general tree has very limited applications
- A tree such that every nodes has at most two child nodes is called Binary tree.
- Any node in a binary tree can have
  - No children
  - Only left child
  - Only right child
  - Both children

# Binary Tree

- A node in a tree can be represented as


- A binary tree with elements 25, 34, 67, 85, 12, 8 can be represented as




- In this tree, we can traverse in inorder, preorder or postorder.

# Traversing in a binary tree

- Inoder
  - Traverse the left subtree
  - Traverse the root
  - Traverse the right subtree
- Preorder
  - Traverse the root
  - Traverse the left subtree
  - Traverse the right subtree
- Postorder
  - Traverse the **left** subtree
  - Traverse the **right** subtree
  - Traverse the **root**

# Binary tree – level of nodes

- Root of a tree is considered as a node at level 0.
- Its immediate children are at level 1. Children of these nodes are at level 2 and so on.
- If a tree has n nodes,
  - what can be the maximum level of any node in the tree?
- Height of a tree
  - Height of a tree with one node is 0
  - Height of a tree with zero nodes is -1
  - Height of a tree is

    1 + max( height(T → left), height(T → right) )