Data Structures and Analysis of Algorithms
CST 225-3

Uva Wellassa University

Introduction to Tree Data Structure

Why Trees?

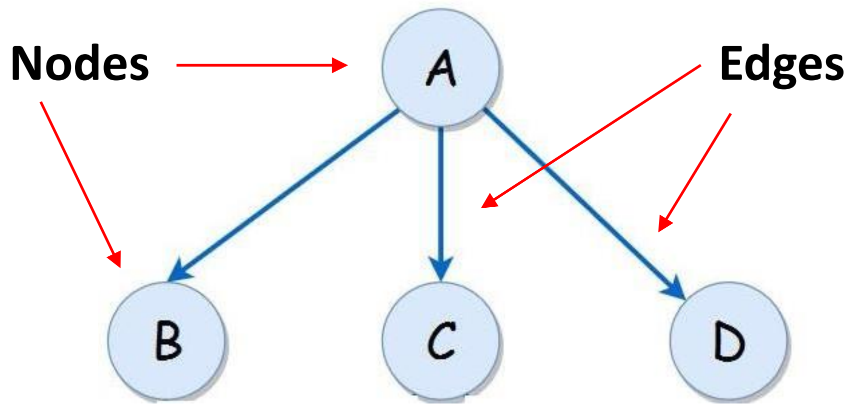
- Used to represent hierarchical data.
- It offers an efficient search and insertion procedure.
- The trees are flexible. This allows subtrees to be relocated with minimal effort.
- Allows to implement faster algorithms (compared with algorithms using linear data structures).

What is a Tree?

- A tree data structure is a non linear data structure.
- It is a hierarchical structure as elements in a tree are arranged in multiple levels.
- Set of nodes storing elements in a parent-child relationship with some more properties.

Tree Properties

- A tree is a collection of entities called nodes.
- Nodes are connected by edges/links.
- Nodes contain values or data.



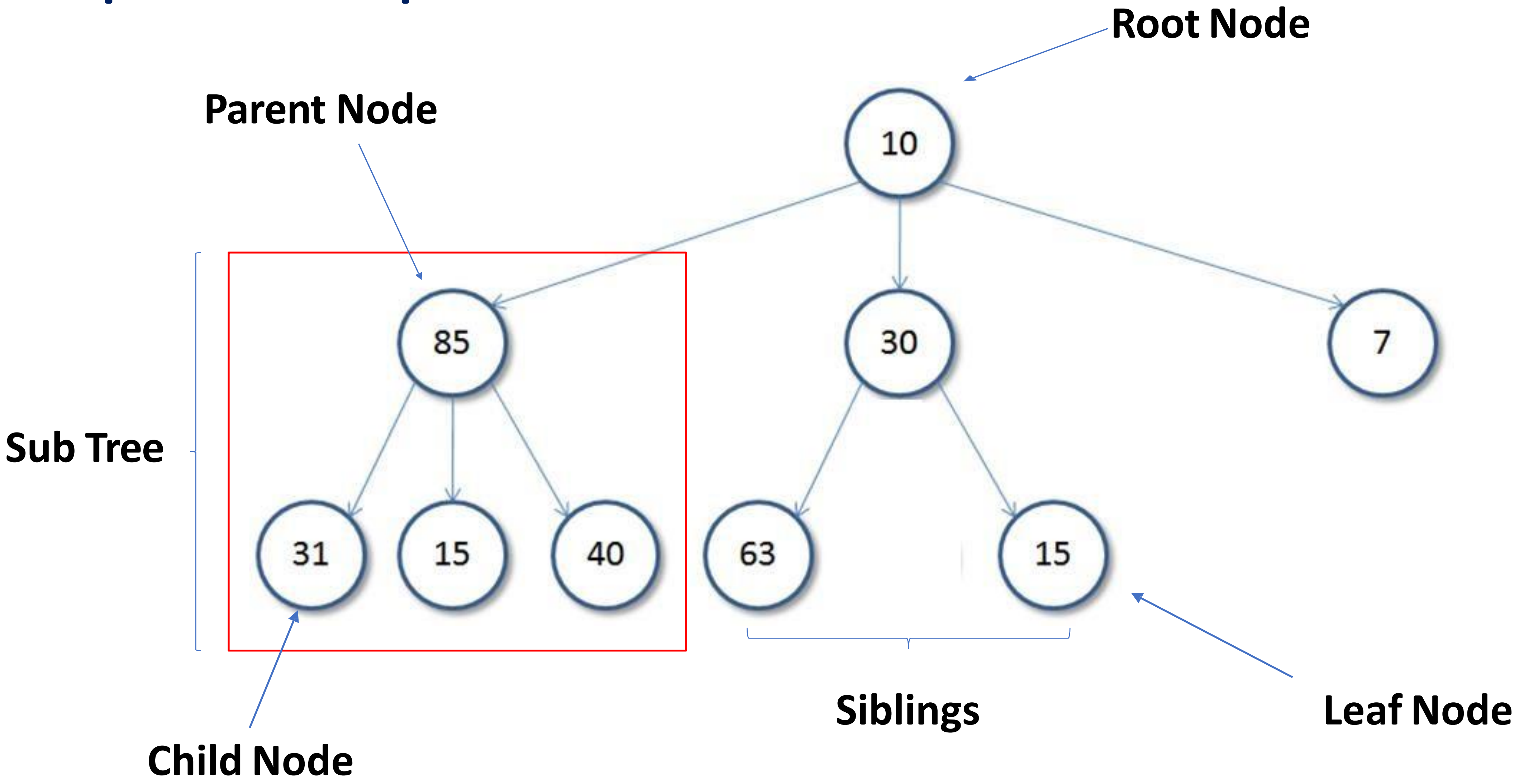
Applications

- In searching algorithms and machine learning algorithms
- Evaluating a mathematical expression
- Database uses tree data structure for indexing
- Social Networking sites uses trees to store data
- Hash map in software applications
- Store hierarchical data, like folder structure, organization structure, XML/HTML data
- Used in Compilers

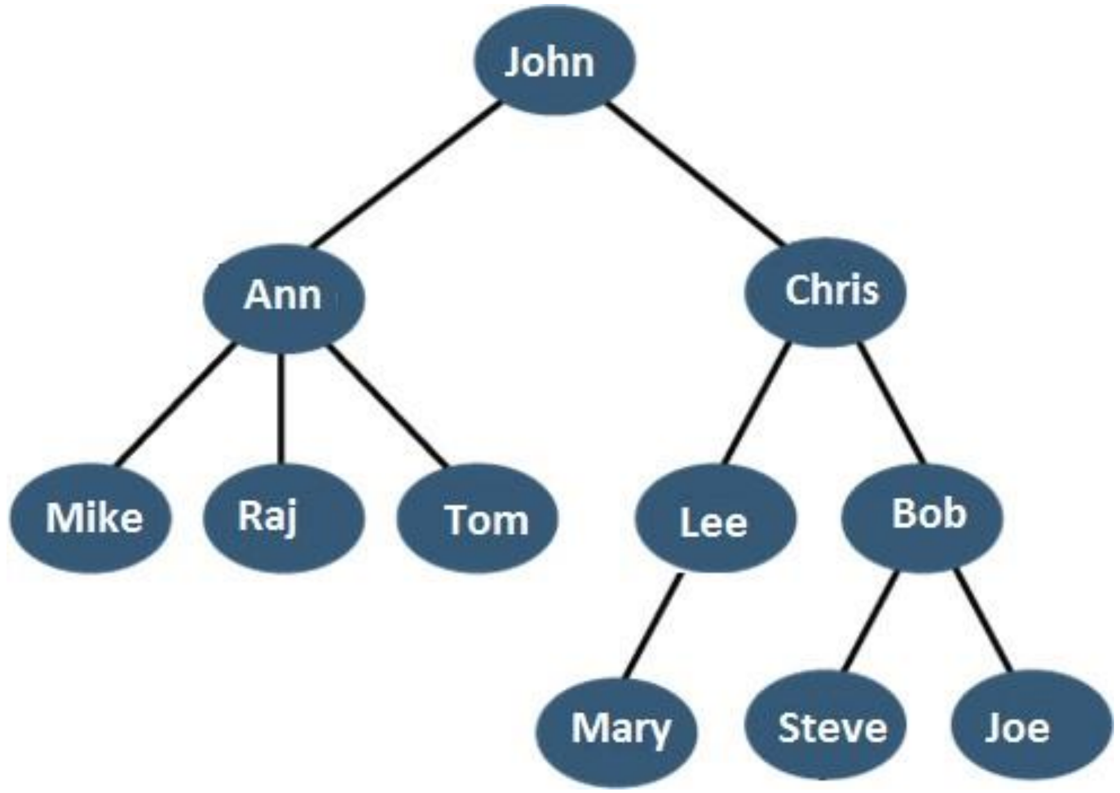
Tree Node Categories

- Root – Only node without parent, the topmost node in the tree hierarchy , Only one root node is allowed in a tree
- Parent- Has child nodes(at least one child-node)
- Child - The node which is descendant of any node
- Siblings – Nodes with same parent
- Leaf /terminal/ external– A node of the tree, which doesn't have any child nodes
- Internal - The node which has at least one child
- Ancestor – Any predecessor node on the path from root to a given node
- Descendent – Any successor node on the path from a given node to leaf node

Graphical Representation



Tree Node Categories



Leaves =

Parent(Ann) =

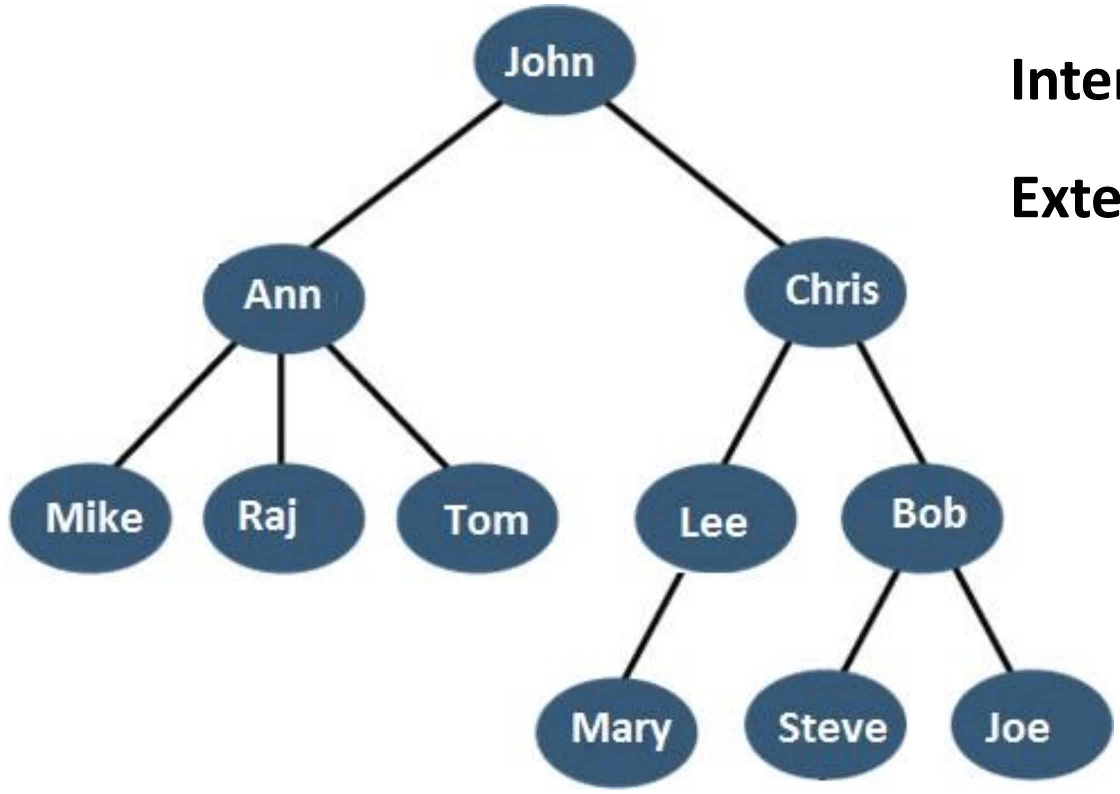
Grandparent(Joe) =

Ancestors(Tom) =

Descendants(Lee) =

Siblings(Raj) =

Tree Node Categories



Internal Nodes =

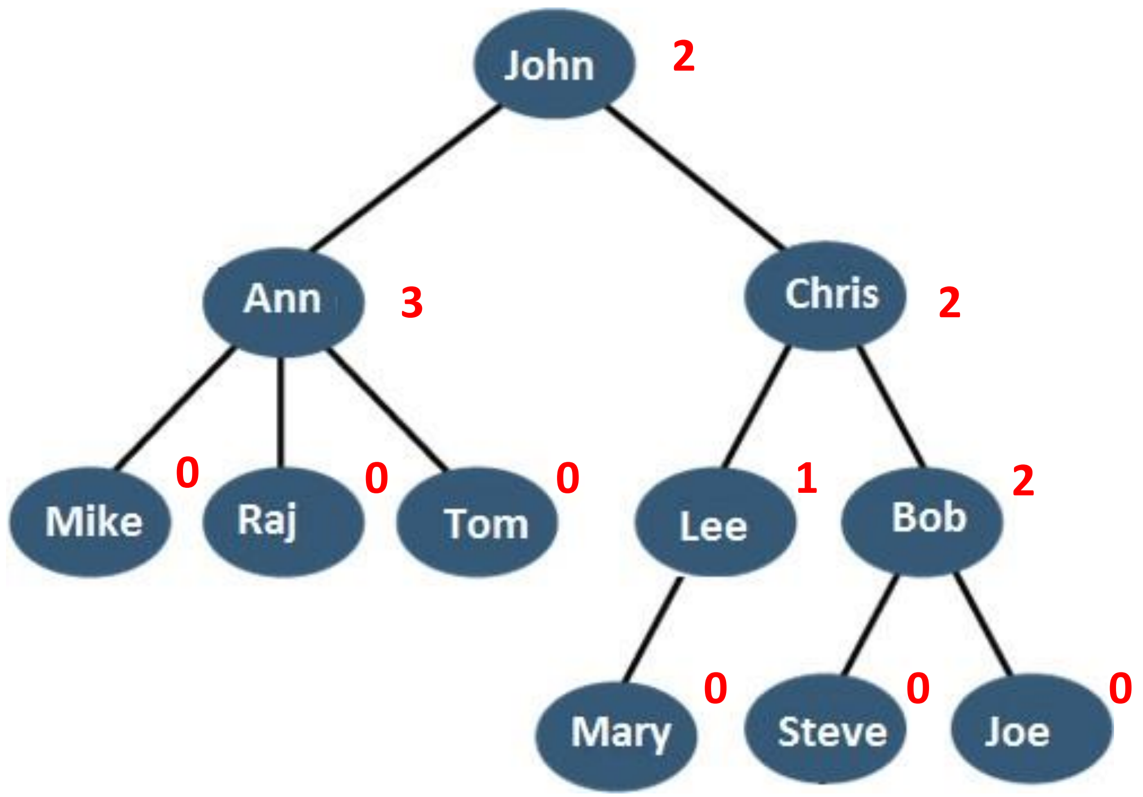
External Nodes =

Tree Properties

- Tree data structure can have following constraints,
  - N number of nodes
  - N-1 edges

Node Degree

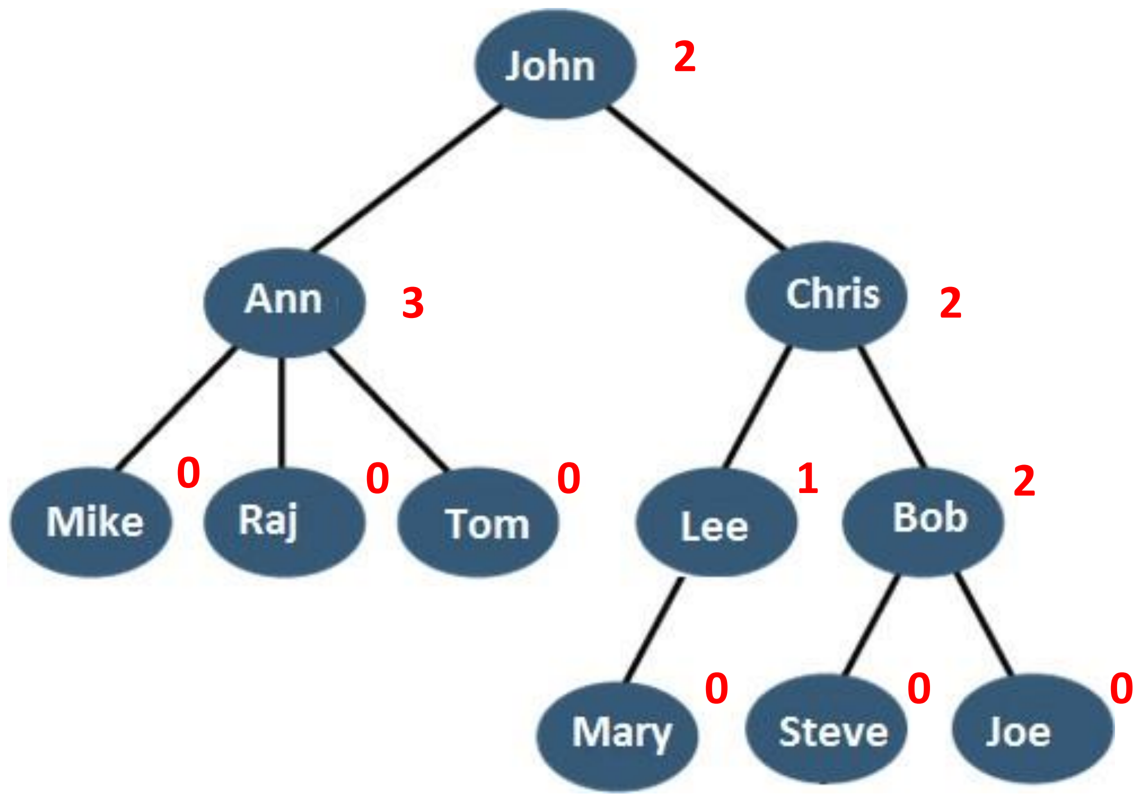
- The total number of children of a node is called as degree of that node.



Degree of Tree

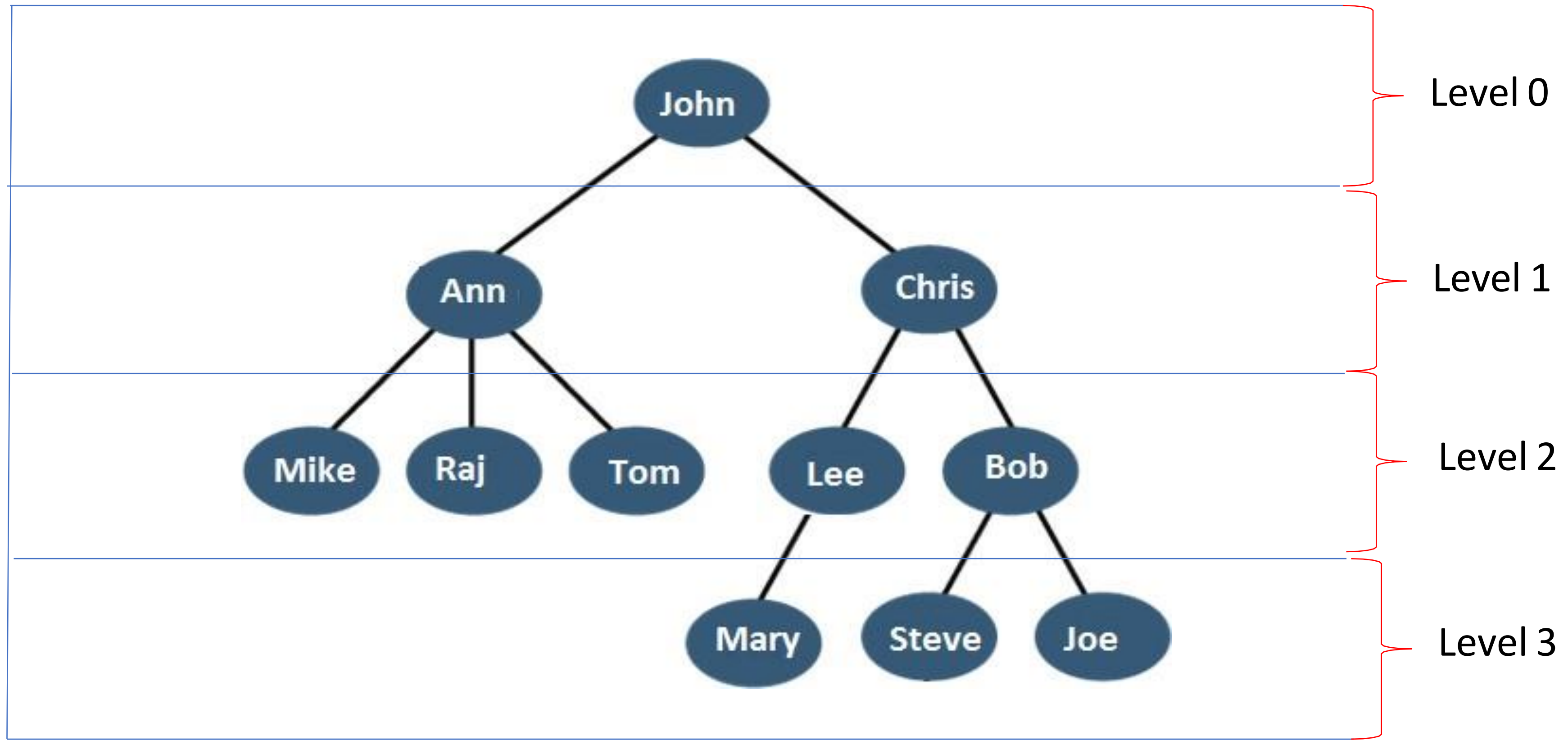
- The highest degree of a node among all the nodes in a tree is called as ‘Degree of Tree’.

Tree Degree = 3



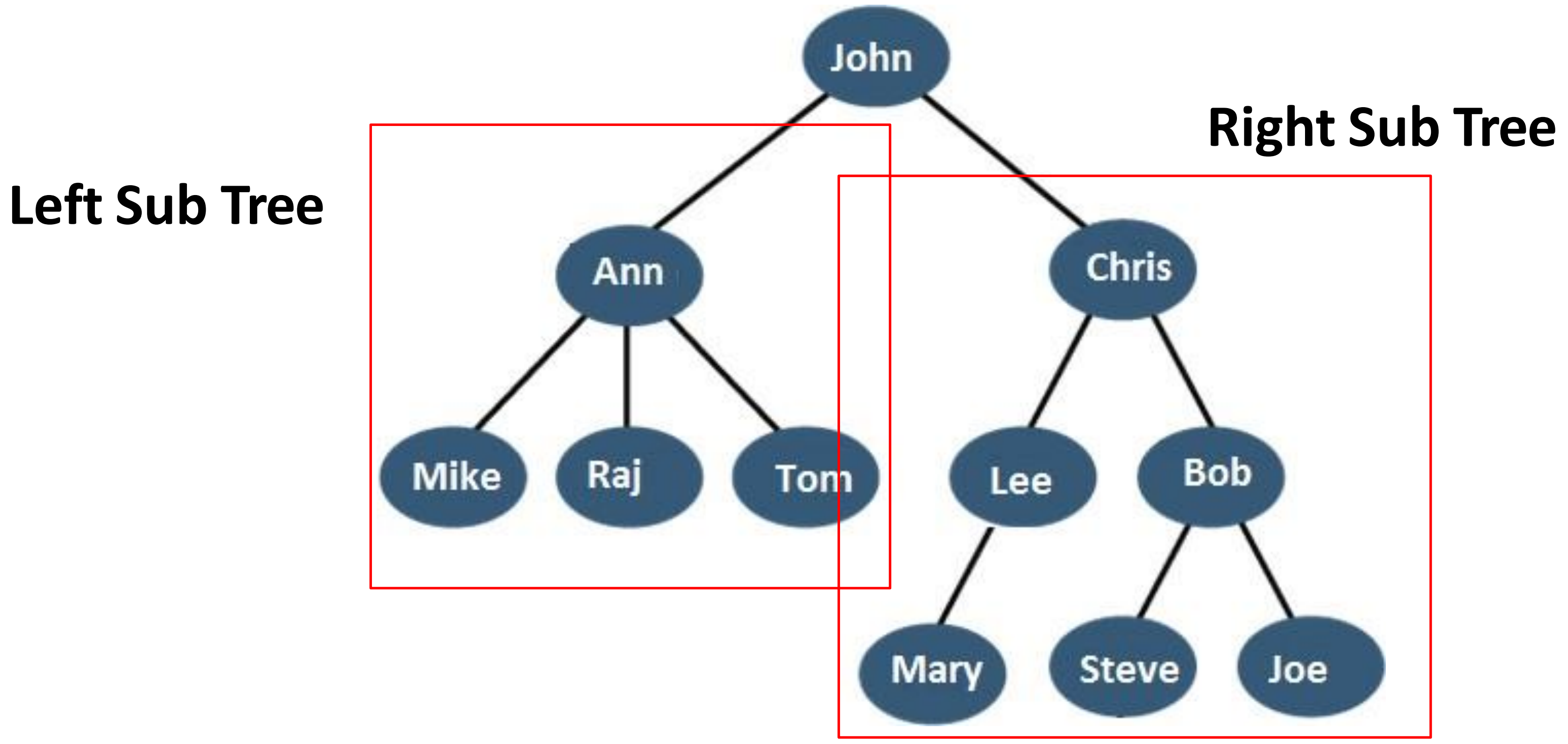
Levels of a Tree

- Level 0 for root node.
- Level 1, Level 2, Level 3,……. etc are for children of root node.



Sub Tree

- A subtree of a given node includes one of its children and all of that child's descendants.
- Each child from a node forms a sub tree recursively.



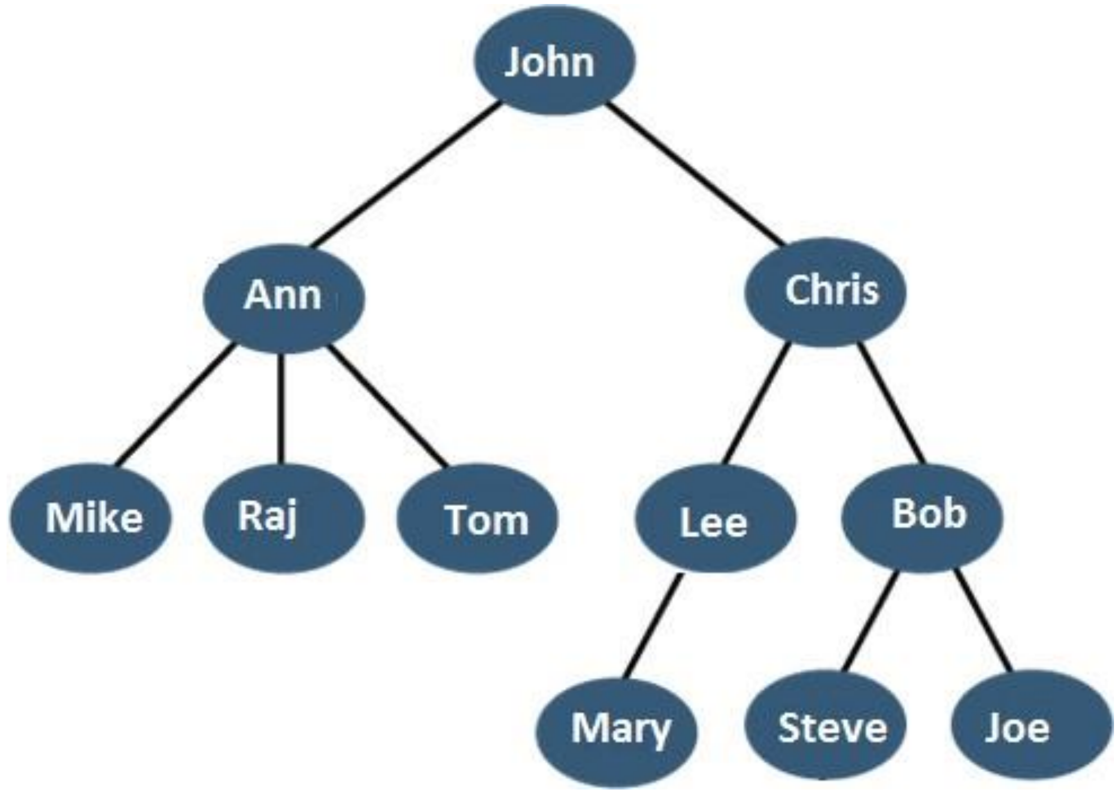
Height

- The total number of edges from a particular node to a leaf node in the longest path is called as height of that node.
- Height of x node = No. of edges in longest path from x to a leaf.
- Height of tree = Height of root node

Depth

- Depth is the total number of edges from root node to a particular node.
- Depth of x node = length of path/ no of edges from root node to x node.
- Depth of the tree = The total number of edges from root node to a leaf node in the longest path.

Exercise



Height of the tree =

Height(Ann) =

Height(Chris) =

Depth(Joe) =

Depth(Tom) =

Path

- Path refers to the sequence of nodes along the edges of a tree.
- Node to node (itself) – the length of path is zero.
- Path length for two nodes in the tree is the number of edges on the path and for two adjacent nodes in the tree, the length of the path is 1.

Types of Tree

- **General tree** - each node can have an arbitrary number of children
- **Binary tree** - each node has at most two children
- **Binary search trees** - binary tree extension with several optional restrictions
- **AVL trees** - self-balancing binary search tree
- Many more types.

Questions?