SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)
Approved by AICTE, New Delhi, Affiliated to Anna University, Chennai
Accredited by NAAC-UGC with ‘A++’ Grade (Cycle III) &
Accredited by NBA (B.E - CSE, EEE, ECE, Mech & B.Tech.IT)
COIMBATORE-641 035, TAMIL NADU

COURSE NAME : 23CAT602 – Data Structures and Algorithms

I YEAR / I SEMESTER

UNIT – II

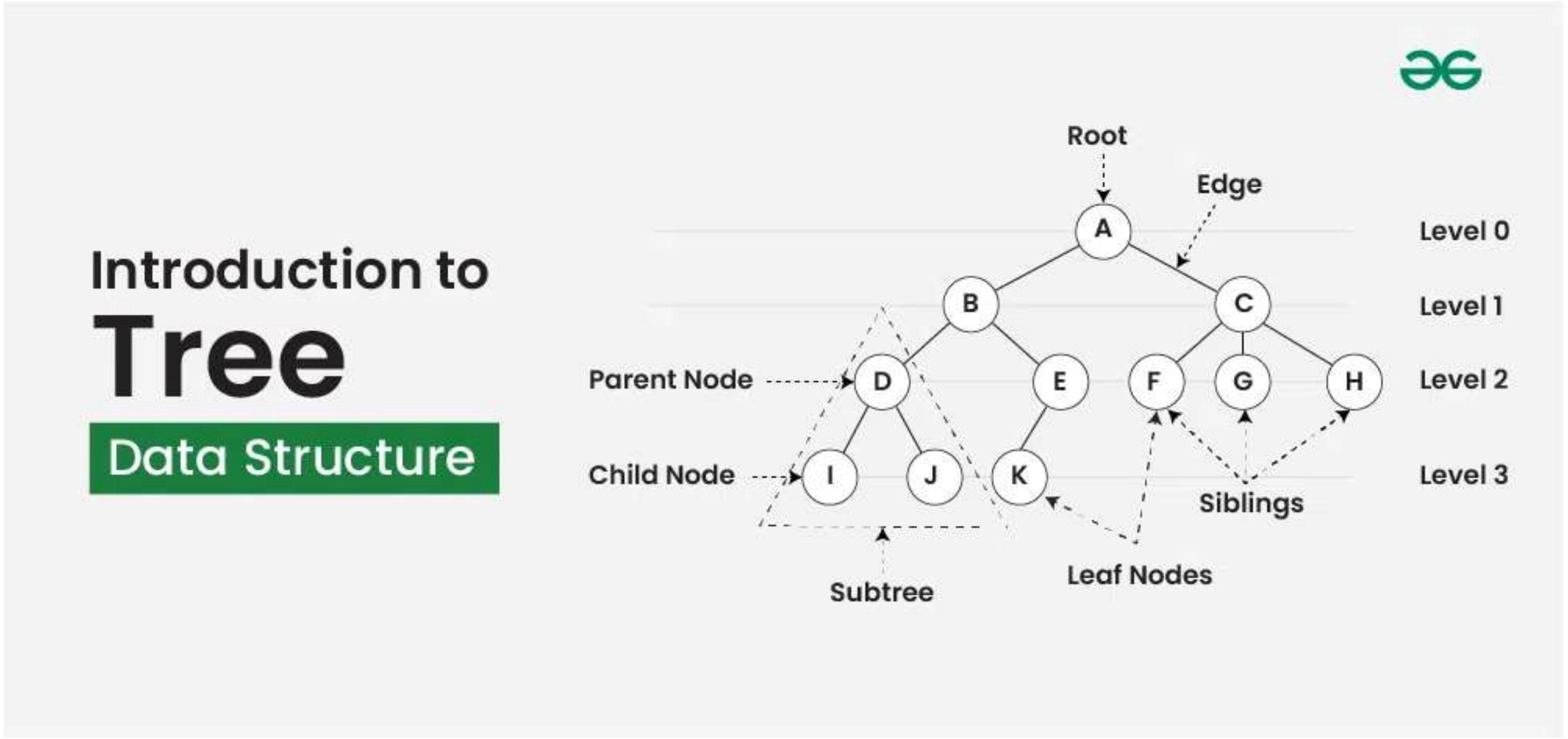
Topic: Binary Tree

Ms.B.Sumathi
Assistant Professor
Department of Computer Applications

Introduction to Tree

- **Tree data structure** is a specialized data structure to store data in hierarchical manner. It is used to organize and store data in the computer to be used more effectively. It consists of a central node, structural nodes, and sub-nodes, which are connected via edges. We can also say that tree data structure has roots, branches, and leaves connected.

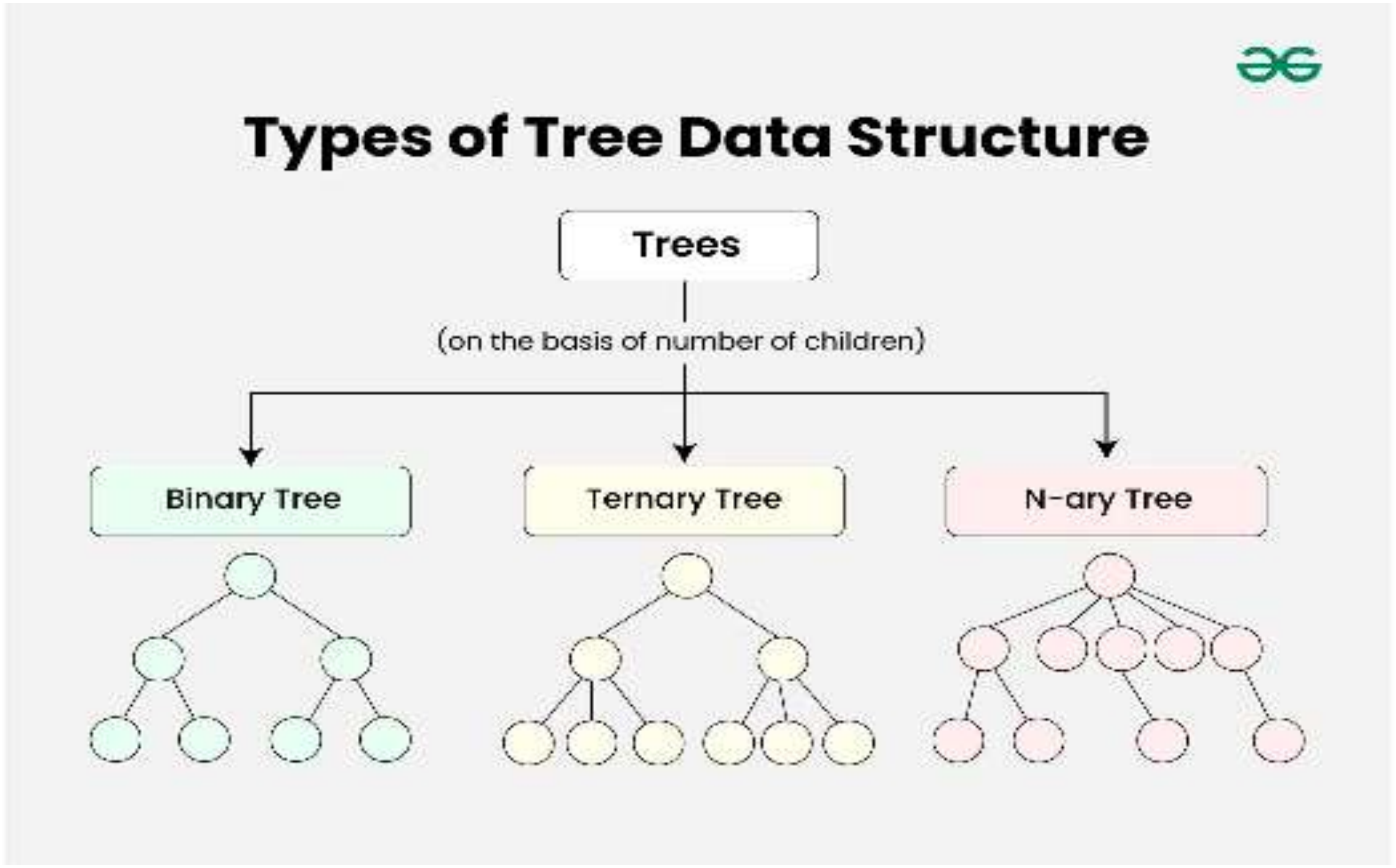
Representation of Tree Data Structures



Basic Terminologies Tree

- **Parent Node:** The node which is a predecessor of a node is called the parent node of that node. **{B}** is the parent node of **{D, E}**.
- **Child Node:** The node which is the immediate successor of a node is called the child node of that node. Examples: **{D, E}** are the child nodes of **{B}.**
- **Root Node:** The topmost node of a tree or the node which does not have any parent node is called the root node. {A**}** is the root node of the tree. A non-empty tree must contain exactly one root node and exactly one path from the root to all other nodes of the tree.
- **Leaf Node or External Node:** The nodes which do not have any child nodes are called leaf nodes. **{I, J, K, F, G, H}** are the leaf nodes of the tree.
- **Ancestor of a Node:** Any predecessor nodes on the path of the root to that node are called Ancestors of that node. **{A,B}** are the ancestor nodes of the node **{E}**
- **Descendant:** A node x is a descendant of another node y if and only if y is an ancestor of x.
- **Sibling:** Children of the same parent node are called siblings. **{D,E}** are called siblings.
- **Level of a node:** The count of edges on the path from the root node to that node. The root node has level **0**.
- **Internal node:** A node with at least one child is called Internal Node.
- **Neighbour of a Node:** Parent or child nodes of that node are called neighbors of that node.
- **Subtree**: Any node of the tree along with its descendant.

Types of Tree Data Structure


Types of Tree Data Structure
Trees
(on the basis of number of children)
Binary Tree
Ternary Tree
N-ary Tree

Basic Operations on Tree Data Structure

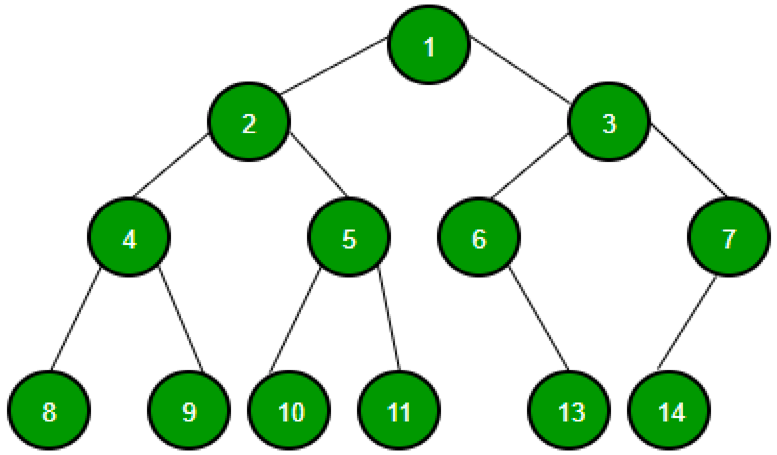
- **Create** – create a tree in the data structure.
- **Insert** – Inserts data in a tree.
- **Search** – Searches specific data in a tree to check whether it is present or not.
- **Traversal**:
  - Depth-First-Search Traversal
  - Breadth-First-Search Traversal

Binary Tree

- A **Binary Tree Data Structure** is a hierarchical data structure in which each node has at most two children, referred to as the left child and the right child. It is commonly used in computer science for efficient storage and retrieval of data, with various operations such as insertion, deletion, and traversal.

Binary Tree

- A **Binary Tree Data Structure** is a hierarchical data structure in which each node has at most two children, referred to as the left child and the right child. It is commonly used in computer science for efficient storage and retrieval of data, with various operations such as insertion, deletion, and traversal.



Properties of Binary Tree

1) The maximum number of nodes at level ‘l’ of a binary tree is 2^l
2) The Maximum number of nodes in a binary tree of height ‘h’ is $2^h – 1$
3) In a Binary Tree with N nodes, the minimum possible height or the minimum number of levels is $Log_2(N+1)$
4) A Binary Tree with L leaves has at least $| Log_2L |+ 1$ levels
5) In a Binary tree where every node has 0 or 2 children, the number of leaf nodes is always one more than nodes with two children
6) In a non-empty binary tree, if n is the total number of nodes and e is the total number of edges, then e = n-1

Types of Binary Tree

- Full Binary Tree
- Degenerate Binary Tree
- Skewed Binary Trees
- Full Binary Tree
- Degenerate Binary Tree
- Skewed Binary Trees

Advantages of Binary Tree

- **Efficient searching**: Binary trees are particularly efficient when searching for a specific element, as each node has at most two child nodes, allowing for binary search algorithms to be used. This means that search operations can be performed in O(log n) time complexity.
- **Ordered traversal:** The structure of binary trees enables them to be traversed in a specific order, such as in-order, pre-order, and post-order. This allows for operations to be performed on the nodes in a specific order, such as printing the nodes in sorted order.
- **Memory efficient**: Compared to other tree structures, binary trees are relatively memory-efficient because they only require two child pointers per node. This means that they can be used to store large amounts of data in memory while still maintaining efficient search operations.
- **Fast insertion and deletion:** Binary trees can be used to perform insertions and deletions in O(log n) time complexity. This makes them a good choice for applications that require dynamic data structures, such as database systems.
- **Easy to implement:** Binary trees are relatively easy to implement and understand, making them a popular choice for a wide range of applications.
- **Useful for sorting:** Binary trees can be used to implement efficient sorting algorithms, such as heap sort and binary search tree sort.

Disadvantages of Binary Tree

- **Limited structure:** Binary trees are limited to two child nodes per node, which can limit their usefulness in certain applications. For example, if a tree requires more than two child nodes per node, a different tree structure may be more suitable.
- **Unbalanced trees:** Unbalanced binary trees, where one subtree is significantly larger than the other, can lead to inefficient search operations. This can occur if the tree is not properly balanced or if data is inserted in a non-random order.
- **Space inefficiency:** Binary trees can be space inefficient when compared to other data structures. This is because each node requires two child pointers, which can be a significant amount of memory overhead for large trees.
- **Slow performance in worst-case scenarios:** In the worst-case scenario, a binary tree can become degenerate, meaning that each node has only one child. In this case, search operations can degrade to O(n) time complexity, where n is the number of nodes in the tree.
- **Complex balancing algorithms:** To ensure that binary trees remain balanced, various balancing algorithms can be used, such as AVL trees and red-black trees. These algorithms can be complex to implement and require additional overhead, making them less suitable for some applications.

Applications of Binary Tree

- DOM in HTML.
- File explorer.
- Used as the basic data structure in Microsoft Excel and spreadsheets.
- Editor tool: Microsoft Excel and spreadsheets.
- Evaluate an expression
- Routing Algorithms

Thank you