

TREES

TREES: Definition; Tree terminologies –node, root node, parent node, ancestors of a node, siblings, terminal & non-terminal nodes, degree of a node, level, edge, path, depth. **BINARY TREE:** Type of binary trees - strict binary tree, complete binary tree, binary search tree and heap tree; Array representation of binary tree. Traversal of binary tree; preorder, inorder and postorder traversal.

What is Tree Data Structure?

A tree is a finite set of nodes in which there is a special node called the root form which the remaining nodes can be portioned into zero, one or more disjoint subsets each of which itself is a tree known as sub tree of T.

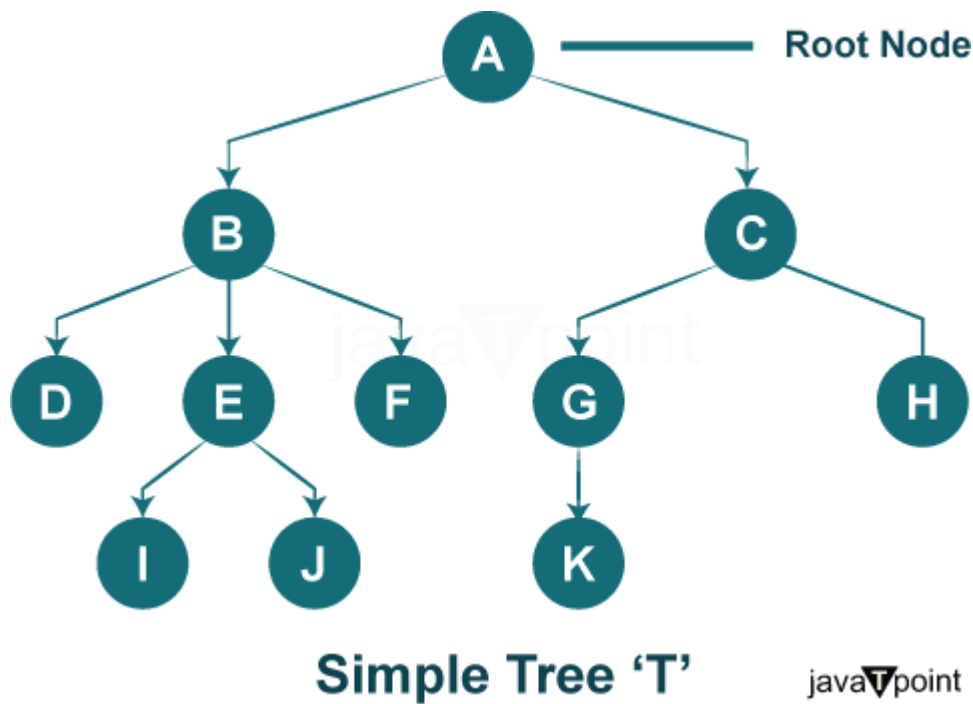
The above tree shows the **organization hierarchy** of some company. In the above structure, *john* is the **CEO** of the company, and John has two direct reports named as *Steve* and *Rohan*. Steve has three direct reports named *Lee*, *Bob*, *Ella* where *Steve* is a manager. Bob has two direct reports named *Sal* and *Emma*. *Emma* has two direct reports named *Tom* and *Raj*. Tom has one direct report named *Bill*. This particular logical structure is known as a **Tree**. Its structure is similar to the real tree, so it is named a **Tree**. In this structure, the **root** is at the top, and its branches are moving in a downward direction. Therefore, we can say that the Tree data structure is an efficient way of storing the data in a hierarchical way.

Let's understand some key points of the Tree data structure.

- A tree data structure is defined as a collection of objects or entities known as nodes that are linked together to represent or simulate hierarchy.
- A tree data structure is a non-linear data structure because it does not store in a sequential manner. It is a hierarchical structure as elements in a Tree are arranged in multiple levels.
- In the Tree data structure, the topmost node is known as a root node. Each node contains some data, and data can be of any type. In the above tree structure, the node contains the name of the employee, so the type of data would be a string.
- Each node contains some data and the link or reference of other nodes that can be called children.
- In Tree data structure node does not have any cycles.

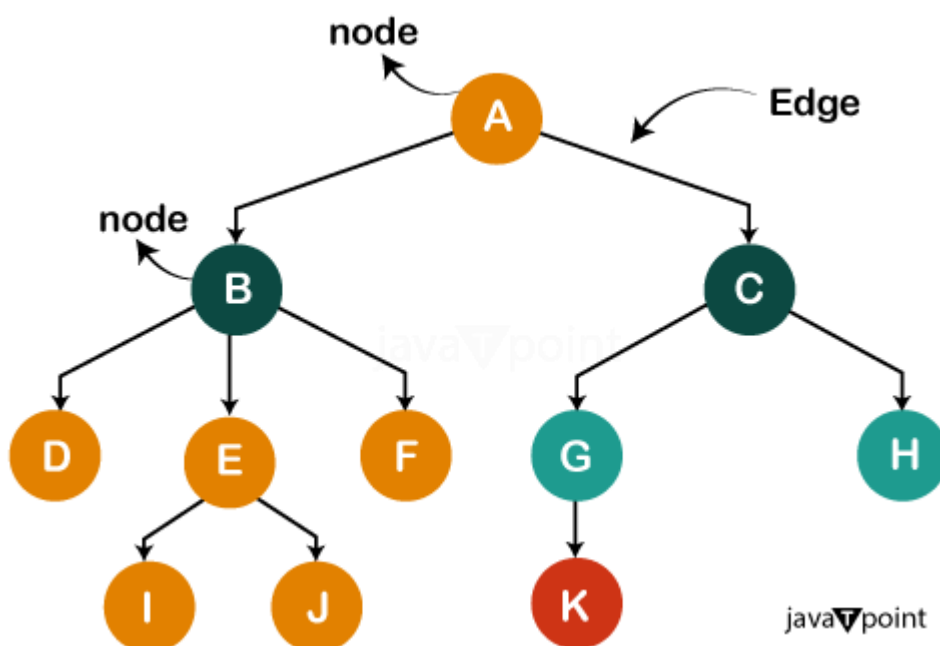
Some basic terms used in Tree data structure.

Let's consider the tree structure, which is shown below:



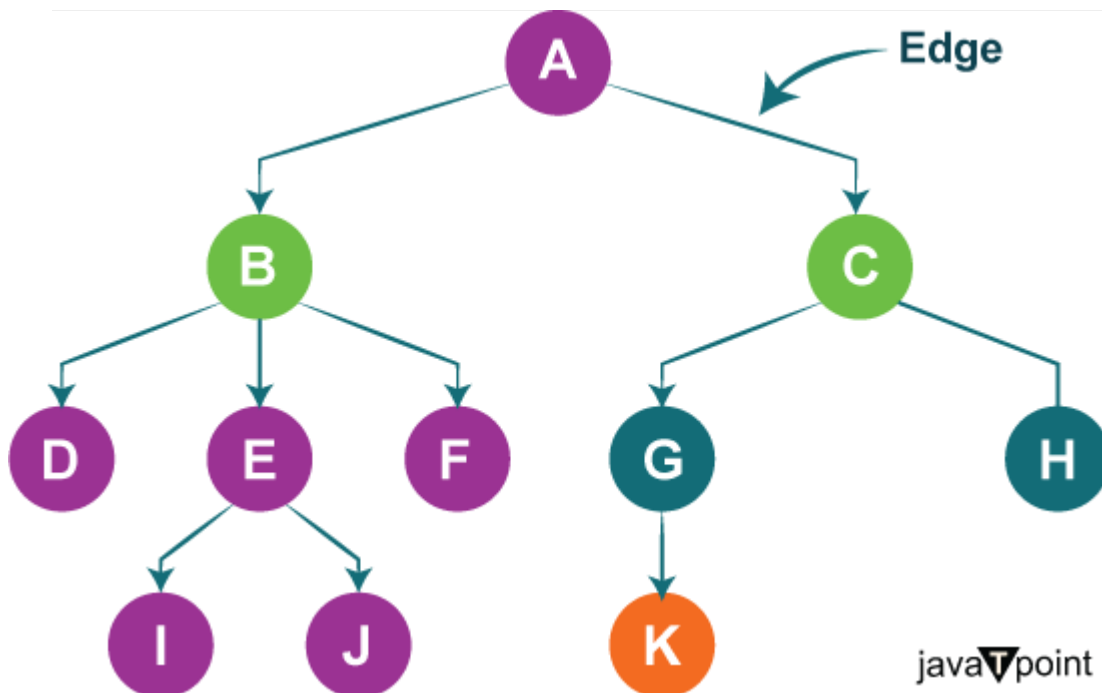
In the above structure, each node is labeled with alphabets. Each arrow shown in the above figure is known as a **link** between the two nodes.

- **Root:** The root node is the topmost node in the tree hierarchy. In other words, the root node is the one that doesn't have any parent. In the above structure, node numbered A is **the root node of the tree**. If a node is directly linked to some other node, it would be called a parent-child relationship.
- **Node:** The node of a tree that stores the data element and may have zero, one, or more links to other descendant nodes for connectivity. It can be a parent, a child, or both.



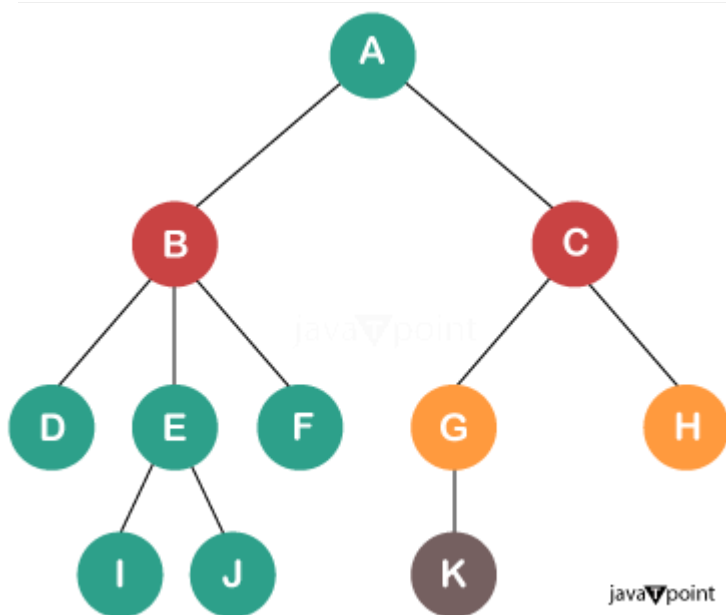
In the above example, A, B, C etc are the Nodes of a Tree.

- **Edge:** The link between any of the two nodes is known as Edge. A directed line from one node to another successor node is called an edge of the tree. A tree with 'N' number of vertices can have maximum number of edges which we can denote as 'N-1'.



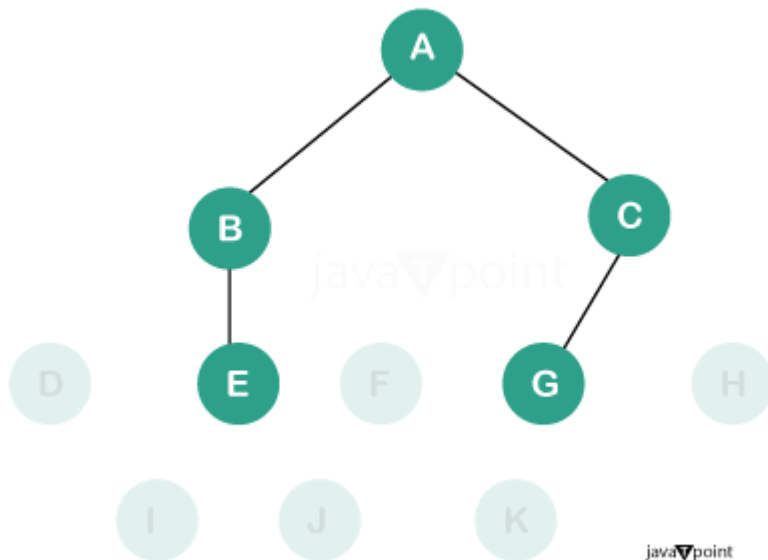
In the above diagrammatic representation it shows various edges between the nodes. **For Example:** An edge between vertex A and Vertex C, An edge between vertex B and Vertex D etc.

- **Child node:** If the node is a descendant of any node, then the node is known as a child node. A parent node can have any number of child nodes. Each node in a tree can have zero or more child nodes.



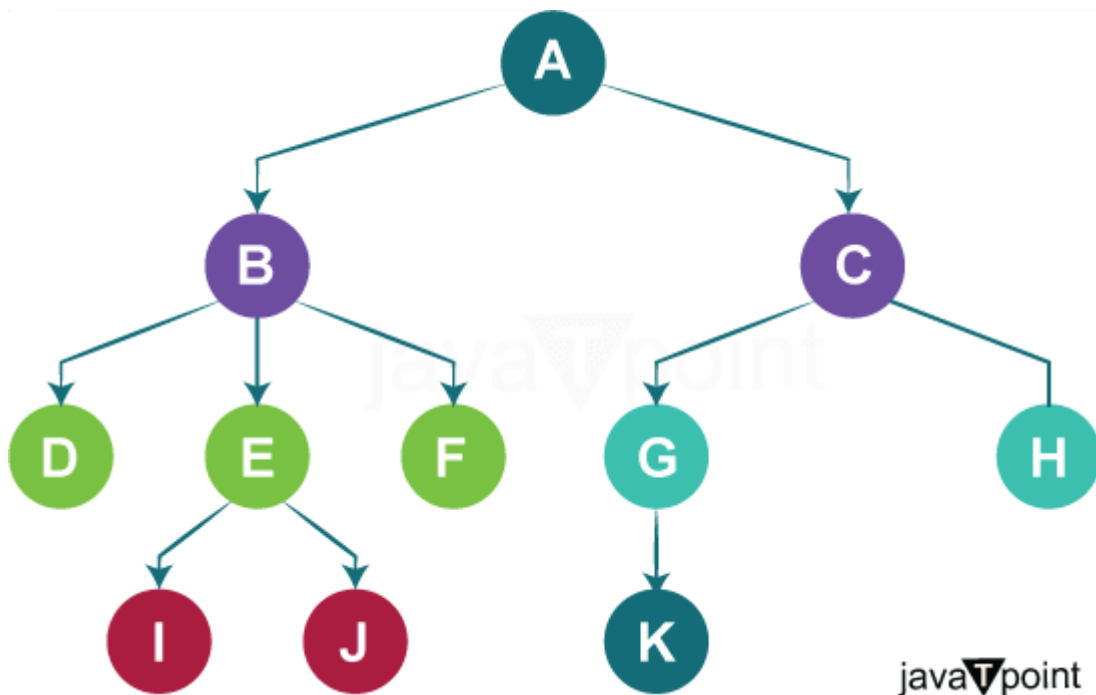
In the above example, descendants of any node show that child node. **For example:** node G, node K are children of node D and children of node B. G and H are the left and right child of node C.

- **Parent:** A parent node can also be denoted as "The node that has child". If the node contains any sub-node, then that node is said to be the parent of that sub-node.



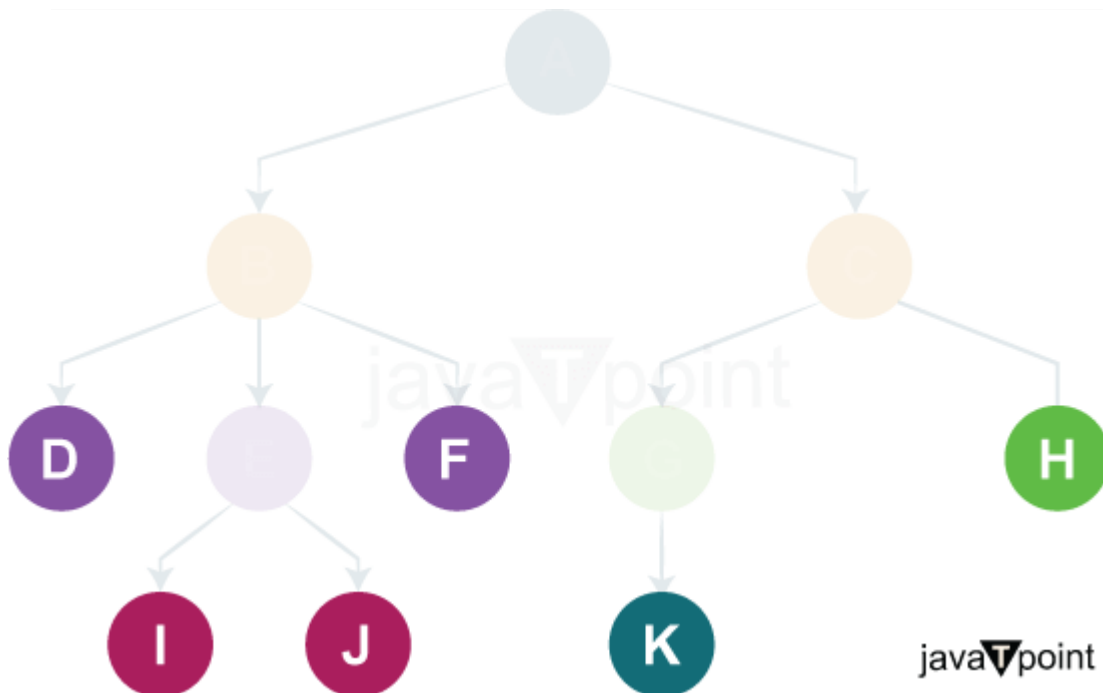
In the above example, the predecessor of any node shows that Parent node **For example:** node A, node B and node G etc. are parent nodes.

- **Sibling:** The nodes that have the same parent are known as siblings.



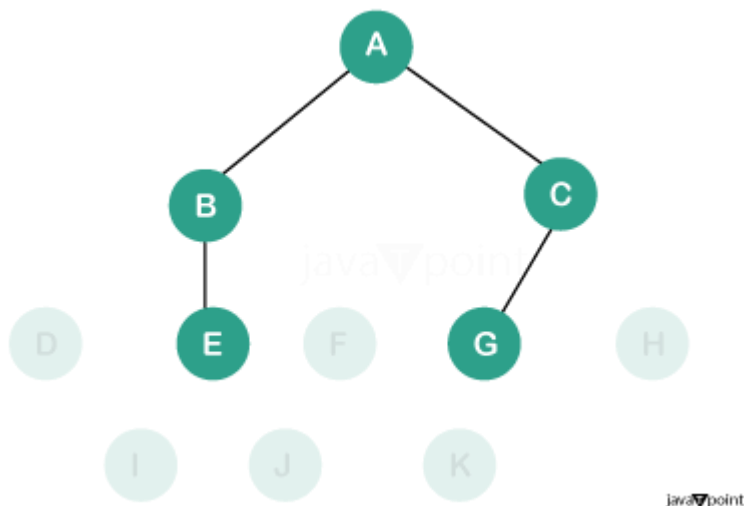
In the above example, it shows that nodes which have the same Parent are known as siblings. **For example:** node G node H are siblings.

- **Leaf Node:-** The node of the tree, which doesn't have any child node, is called a leaf node. A leaf node is the bottom-most node of the tree. There can be any number of leaf nodes present in a general tree. Leaf nodes can also be called external nodes and Terminal nodes.



In the above example, it shows the various leaf nodes such as node D, F, K etc.

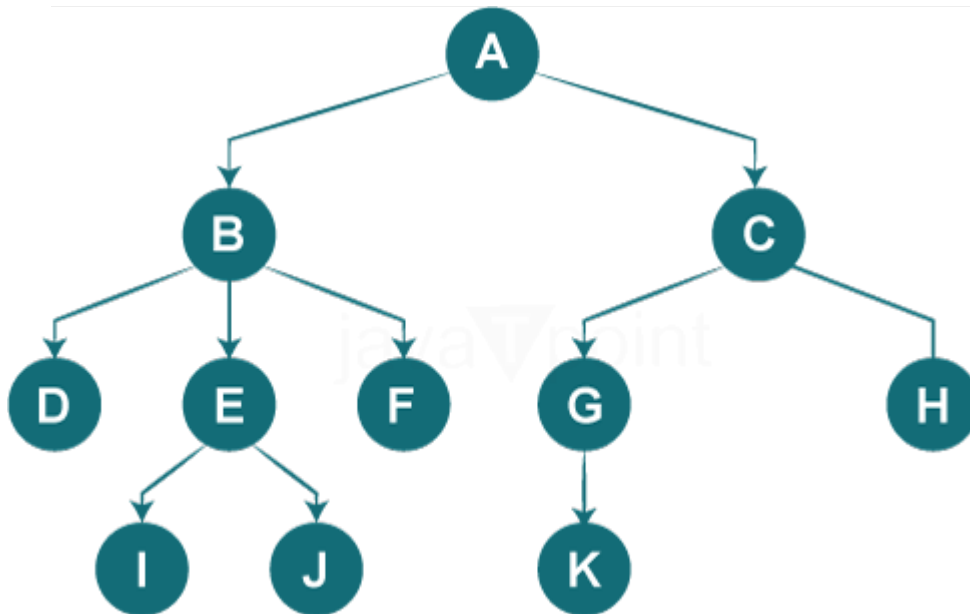
- **Internal nodes:** A node has atleast one child node known as an *internal* node and non-terminal nodes in tree data structure.



In the above example, it shows the various internal nodes such as node A, B, C, G etc.

- **Ancestor node:-** An ancestor of a node is any predecessor node on a path from the root to that node. The root node doesn't have any ancestors. In the tree shown in the above image, nodes 1, 2, and 5 are the ancestors of node 10.
- **Descendant:** The immediate successor of the given node is known as a descendant of a node. In the above figure, 10 is the descendant of node 5.

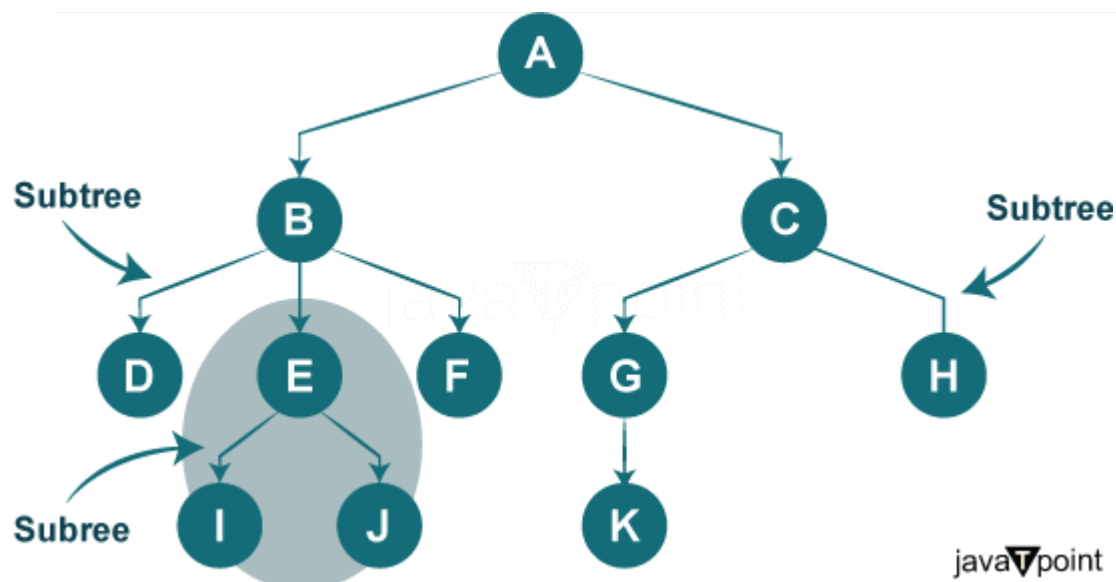
- **Degree:** The total number of node's children is called the degree of that node in tree data structure.



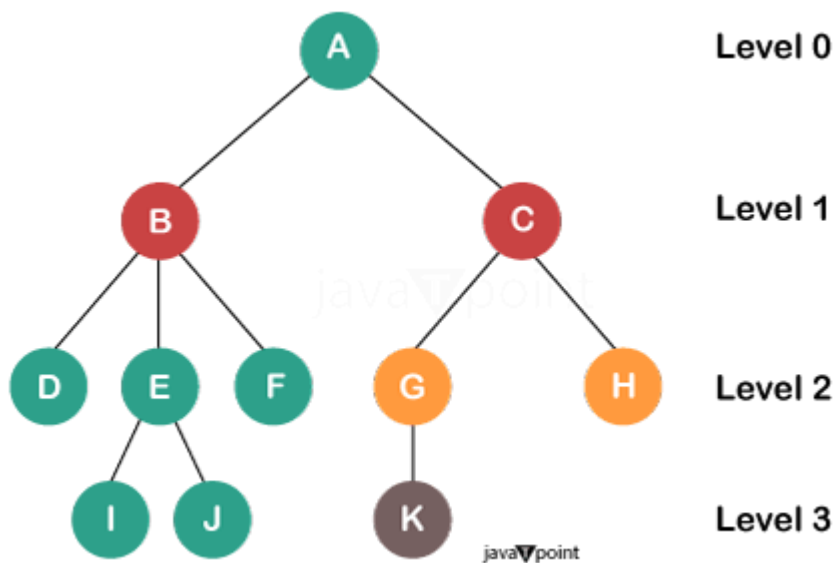
Degree of Tree is : 3 javaVpoint

In the above diagrammatic representation, we have seen the degree of the various nodes from its starting point to end point. **For Example:** The degree of node K is 0 and node E is 1 etc.

- **Subtree:** If a tree is further divided into smaller trees it is called a subtree in the tree data structure. A subtree is a part of a tree that can be viewed as a complete tree in that each child node forms a subtree on its parent node. The following example shows the different subtrees.

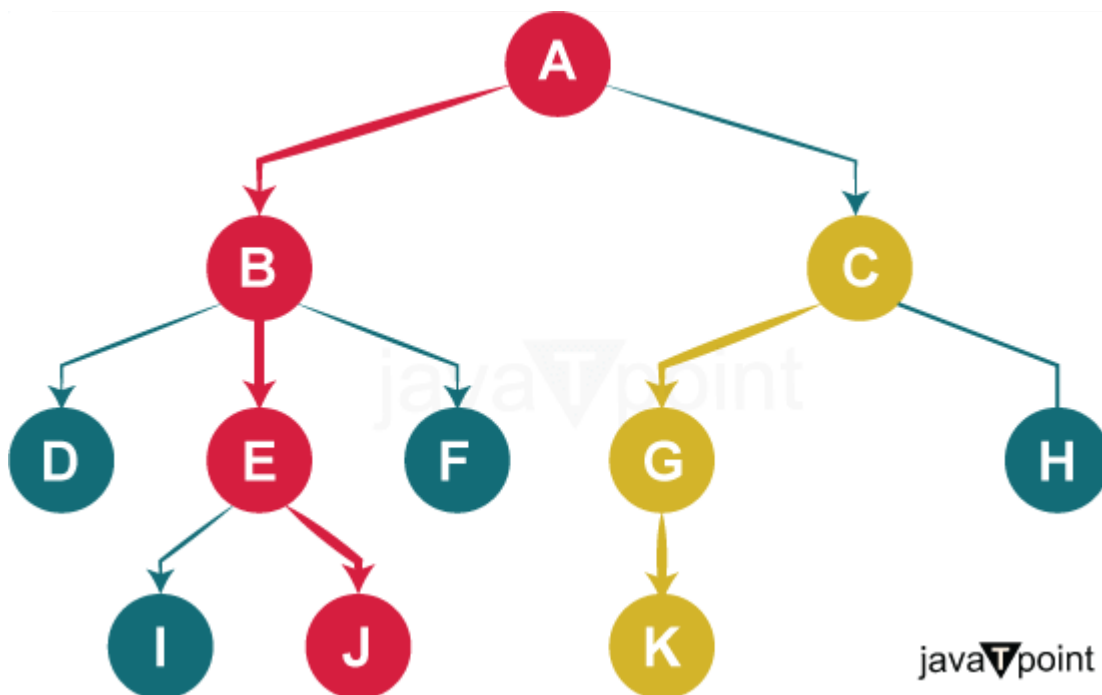


- **Level:** The node's level is an integer value that measures the node's distance from the root. **For example:** the root node is at level 0 and the children of the root node are at its next level (level 1) and so on.



Above diagrammatic representation represents the level 3 starts from the root node i.e. level 0.

- **Path:** The path is a sequence of consecutive edges and nodes from the source node to another node i.e. destination.



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.

Type of binary trees –

1. strict binary tree,
2. complete binary tree,

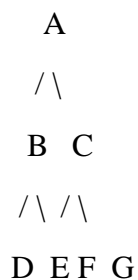
3. binary search tree and
4. heap tree

1. Strict Binary Tree (Proper or Full Binary Tree)

Definition:

A **strict binary tree** is a binary tree in which every node has either **0 or 2 children**. No node can have only one child.

Example Tree:



Every node has either 0 or 2 children.

2. Complete Binary Tree

Definition:

A **complete binary tree** is a binary tree in which **all levels are completely filled**, except possibly the **last level**, which is filled **from left to right**.

Example Tree:



The last level (D, E, F) is filled from left to right.

3. Binary Search Tree (BST)

Definition:

A **BST** is a binary tree where for every node:

- Left child $<$ Node
- Right child $>$ Node

Example Tree:

```

50
 / \
30  70
 / \ / \
20 40 60 80

```

4. Heap Tree (Min Heap / Max Heap)

Definition:

A heap is a complete binary tree satisfying the heap property:

- Min Heap: Parent \leq children
- Max Heap: Parent \geq children

Example Min Heap:

```

10
 / \
20  15
 / \
30  40

```

A **Max Heap** is a **complete binary tree** where every parent node is **greater than or equal to** its children.

- It is usually implemented using **arrays**.
- Used in **priority queues**, **heapsort**, and **scheduling algorithms**.

Example:

```

50
 / \

```

30 40

/\ /\

10 5 20 30

Important Question:

1. **Define** Binary Search Tree (BST) and describe its structure with a neat diagram
2. **Explain** terminal and non- terminal nodes in binary tree with a neat diagram.
3. **Develop** a program to represent a binary tree and implement Preorder, Inorder, and Postorder traversal methods.
4. **Discuss** the types of binary tree and explain any one type with example
5. **Explain** different tree terminologies with a neat diagram of Complete binary tree.
6. **Write** a program to represent a binary tree and implement Preorder, Inorder, Postorder traversal methods.