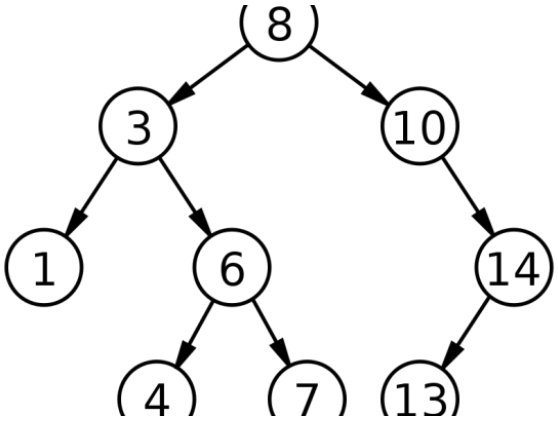
What is a Tree

A tree is a data structure that arranges the data similar to a tree. A node is a data item in the tree. The major node is the root, and the other nodes are its children nodes. All these other nodes are arranged into the non-empty sets where each one of them is a subtree. Moreover, there is a parent-child relationship between the nodes. One parent node can have multiple child nodes, and there can only be one parent node for each child node.

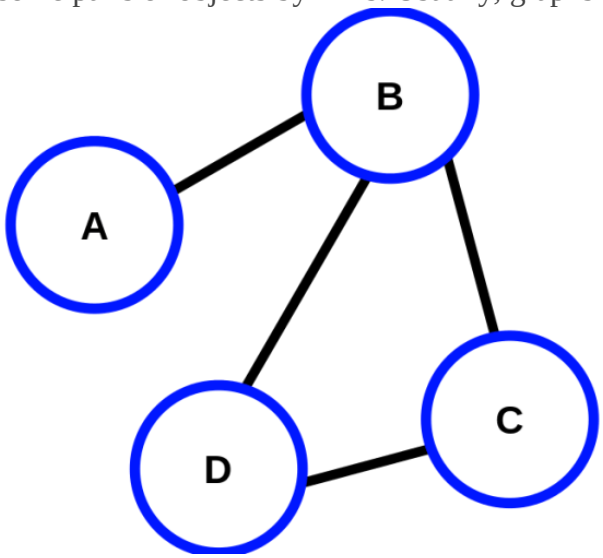


Some important terms related to a tree are as follows. You can see these features and examples in the above tree.

Root node is the topmost data item in the tree. Element 8 is the root node in the above image.
Edge helps to connect nodes. For example, in the above tree, edges connect 8 and 3, 8 and 10.
Parent node is a node other than the root node that connects upwards by an edge. For instance, 3 is the parent node of 1 and 6. Similarly, 6 is the parent node of 4 and 7.
Child node is a node that connects downwards by an edge. For example, 4 and 7 are child nodes of 6.
Leaf node is a node that does not have any child nodes. 1, 4,7,13 are leaf nodes in the above tree.
Subtree is a descendant of a node. For example, the section to the left of the root node (8) that begins from 3 is a subtree. Similarly, the section to the right of the root node that begins from 10 is a subtree.
Level represents the generation of nodes. As an example, the root node belongs to level 0. 3 and 10 belongs to level 1 and so on.
Furthermore, there are two major tree types as binary tree and binary search tree. In a binary tree, each node can have a maximum of 2 child nodes. A binary search tree is an ordered binary tree.

What is a Graph

A graph is a data structure that represents a pictorial structure of a set of objects that connects some pairs of objects by links. Usually, graphs help to represent networks.



Some important terms related to the graph are as follows.

Vertices are objects or the data items. The circles represent them. In the above graph, A, B, C, and D are vertices. We can also write vertices as V = {A,B,C,D}.
Edges are the links that connect the vertices. For example, edges above connect A and B vertices, B and D vertices, etc. We can also write edges as E = {AB, BC, BD, DC}
Path represents the sequence of nodes to follow in order to reach the destination node. For example, ABD represents the path from vertex A to D.
When two nodes connect to each other through an edge, they are **adjacent nodes**. For instance, A and B are adjacent nodes. Similarly, B and D are adjacent nodes.
The major operations we can perform on graphs are adding vertexes, adding edges and displaying vertexes.

Mainly, there are two types of graphs as directed and undirected graphs. When a graph contains an ordered pair of vertices, it is a directed graph, and when a graph contains an unordered pair of vertices, it is an undirected graph.

TREE VERSUS GRAPH

TREE	GRAPH
A data structure that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node	A data structure that consists of a group of vertices connected through edges
There is a root node	There are no root nodes
There are no loops	There can be loops
Represents data in the form of a tree structure, in a hierarchical manner	Represents data similar to a network
Two major types are binary tree and binary search tree	Two major types directed and undirected graphs
Less complex	More complex

Visit www.PEDIAA.com

"A Tree can be a Graph, but a Graph can't always be a Tree"

What is a Graph?

A **graph** is a collection of:

- **Vertices (nodes)** and
- **Edges (connections)** between those vertices.

Graphs can be:

- **Directed or Undirected**
- **Cyclic or Acyclic**
- **Connected or Disconnected**

What is a Tree?

A **tree** is a **special type of graph** with the following properties:

- It is **connected**: there is a path between any two nodes.
- It is **acyclic**: it does **not** contain any cycles.
- It has **exactly N-1 edges** for **N nodes**.
- It has **one root** and a **hierarchical structure**.
- Each child node has **only one parent**.

Why Every Tree is a Graph

A **tree satisfies all conditions of a graph**:

- It has vertices and edges.
- It is a connected structure.
- Therefore, **every tree is a graph**.

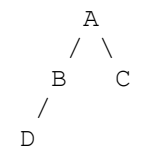
□ Why Every Graph is NOT a Tree

While graphs are more flexible, **not all graphs are trees**, because:

- A graph **can have cycles** → a tree **cannot**.
- A graph can be **disconnected** → a tree **must be connected**.
- A graph can have **multiple paths** between two nodes → a tree has **exactly one unique path**.
- A graph can have **more or fewer than N-1 edges** → a tree must have exactly **N-1** edges.

Example to Understand the Difference

1. Tree (Graph Example)



- Connected
- Acyclic
- 4 nodes, 3 edges → satisfies N-1 rule
 This is both a Tree and a Graph.

2. Graph (Not a Tree)

- Contains a cycle (A → B → C → A)
 This is a Graph but NOT a Tree.

Conclusion:

- **Every tree is a graph** because it meets all graph criteria.
- **Not every graph is a tree** because it may have cycles, may be disconnected, or may not follow the structure and edge rule of a tree.