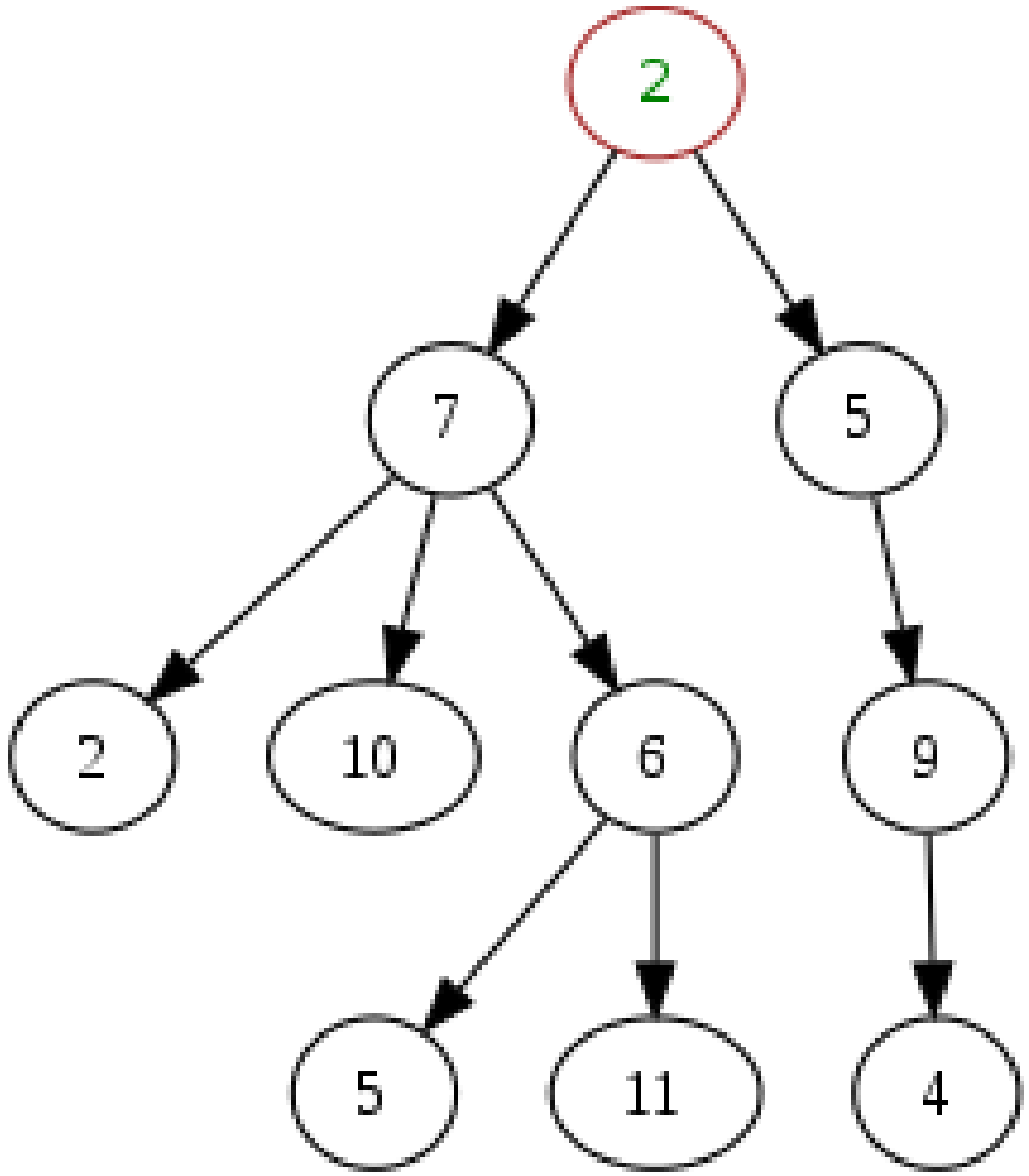
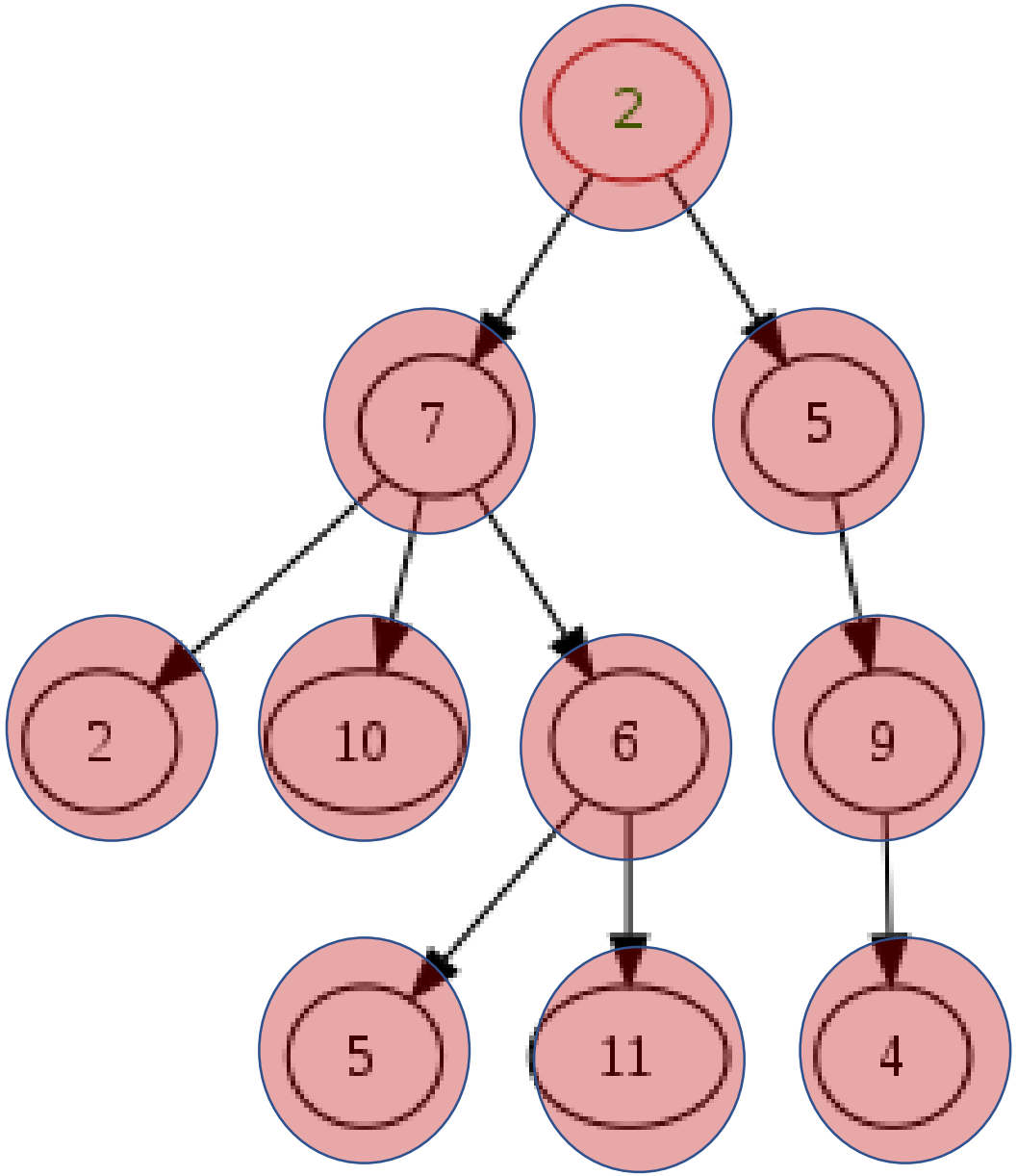
Trees

Introduction to trees



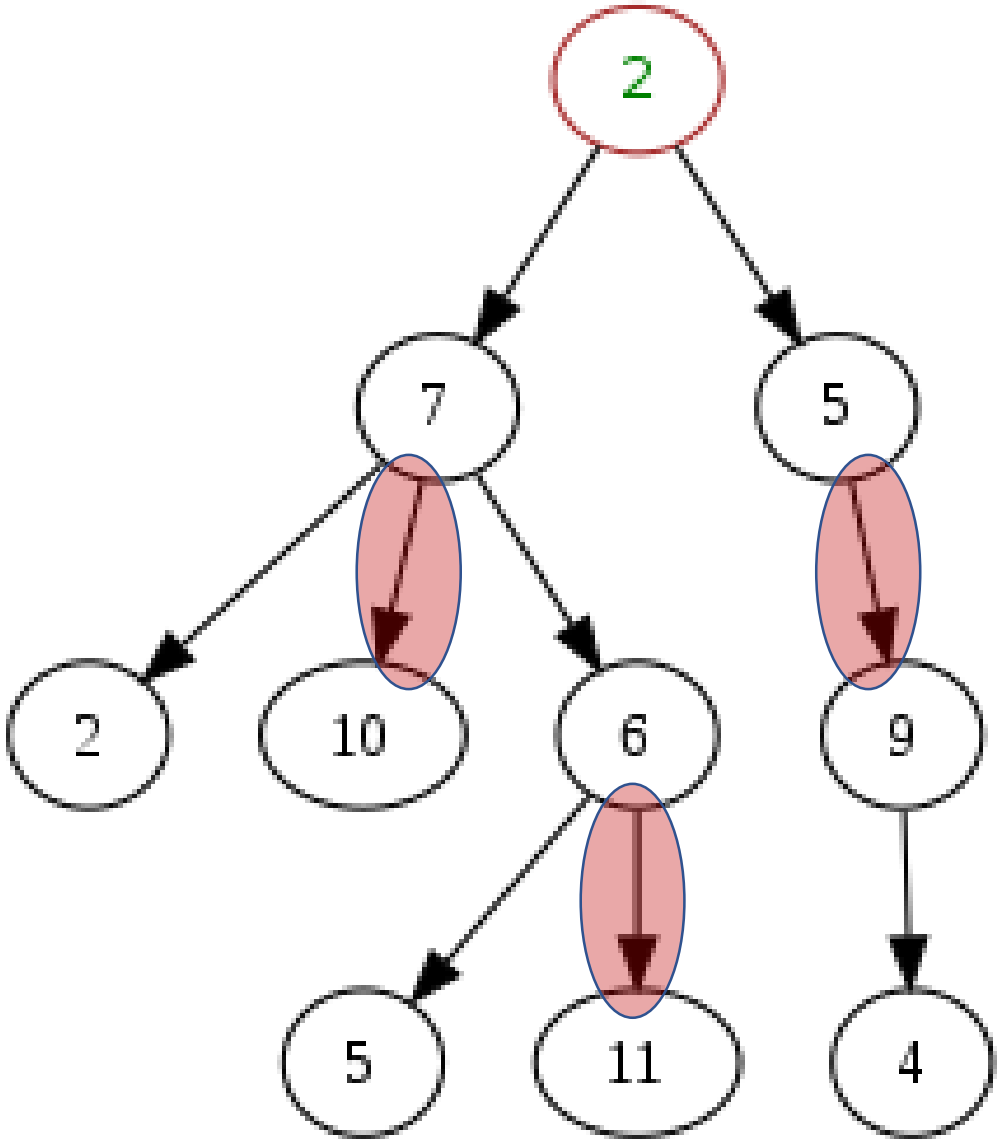
Terminology

- ***Node***
- Edge
- Root
- Path
- Children
- Parent
- Sibling
- Subtree
- Leaf Node



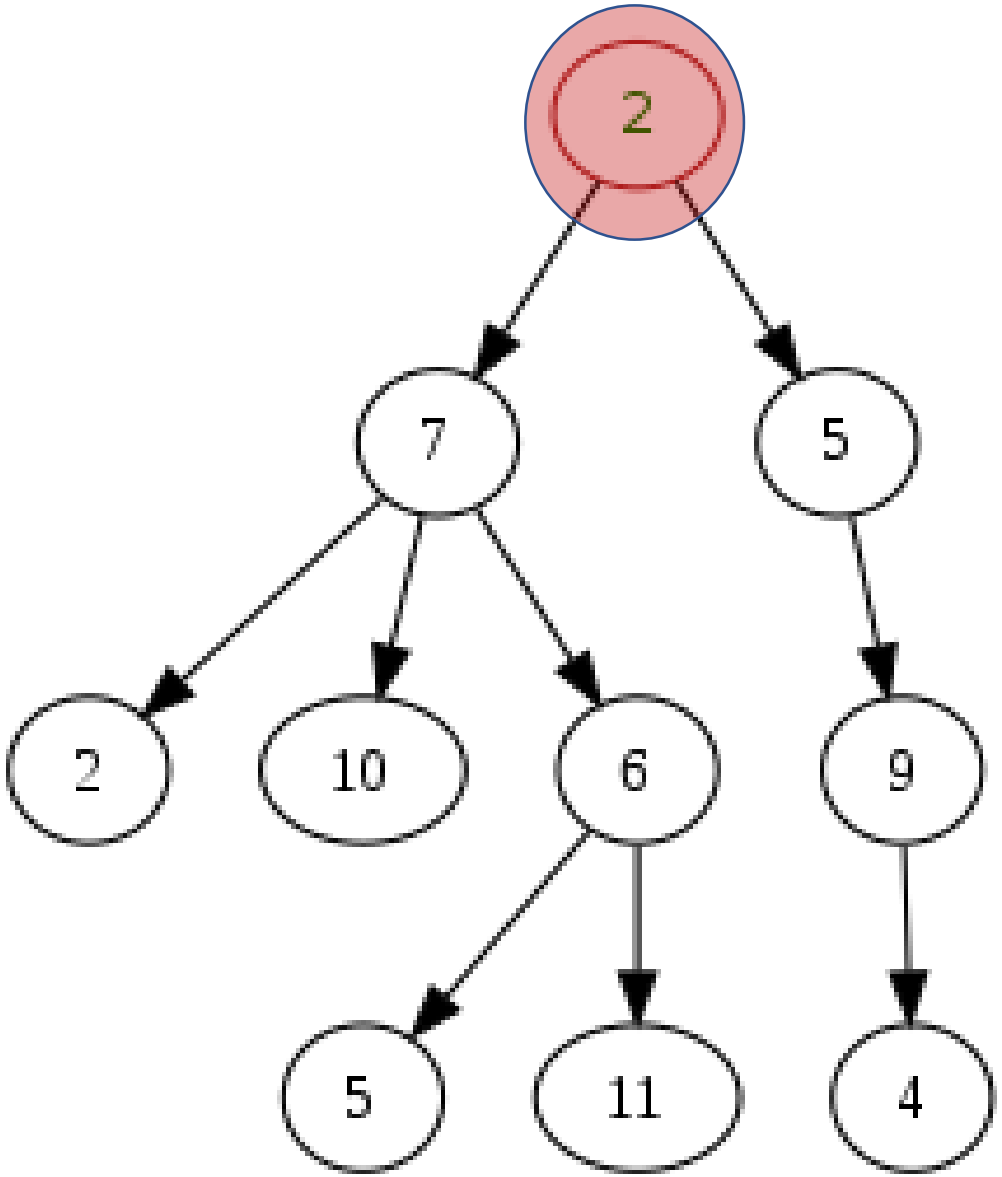
Terminology

- Node
- ***Edge***
- Root
- Path
- Children
- Parent
- Sibling
- Subtree
- Leaf Node



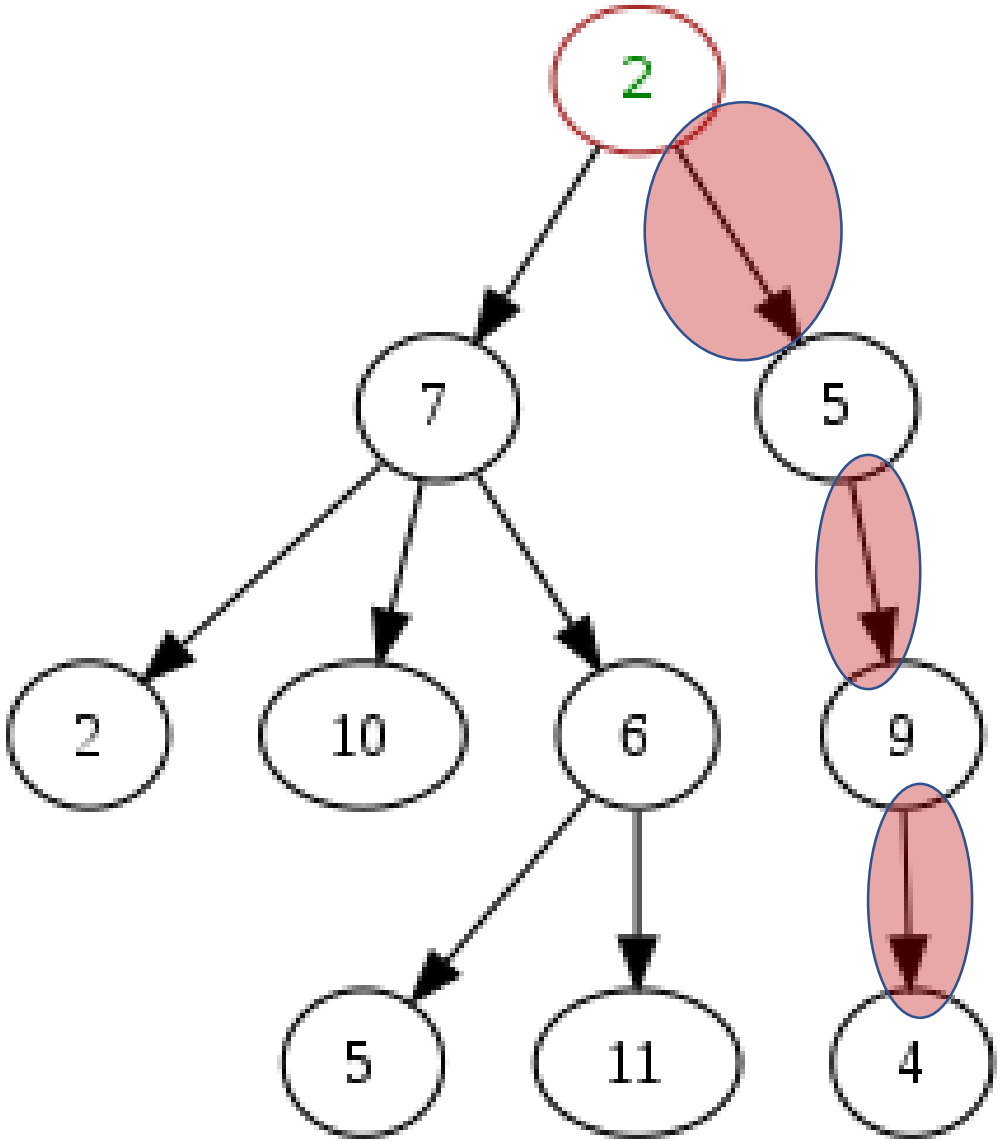
Terminology

- Node
- Edge
- ***Root***
- Path
- Children
- Parent
- Sibling
- Subtree
- Leaf Node



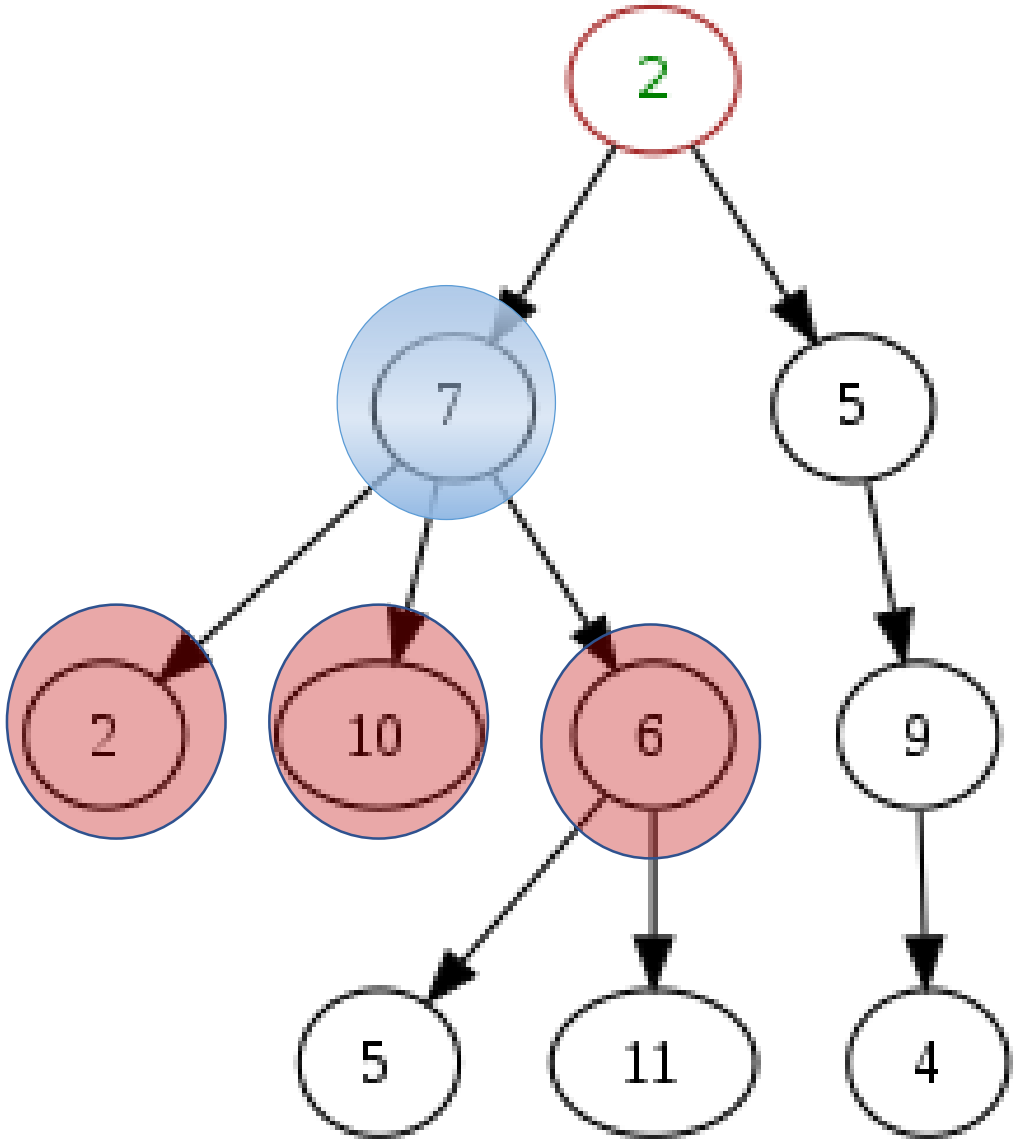
Terminology

- Node
- Edge
- Root
- ***Path***
- Children
- Parent
- Sibling
- Subtree
- Leaf Node



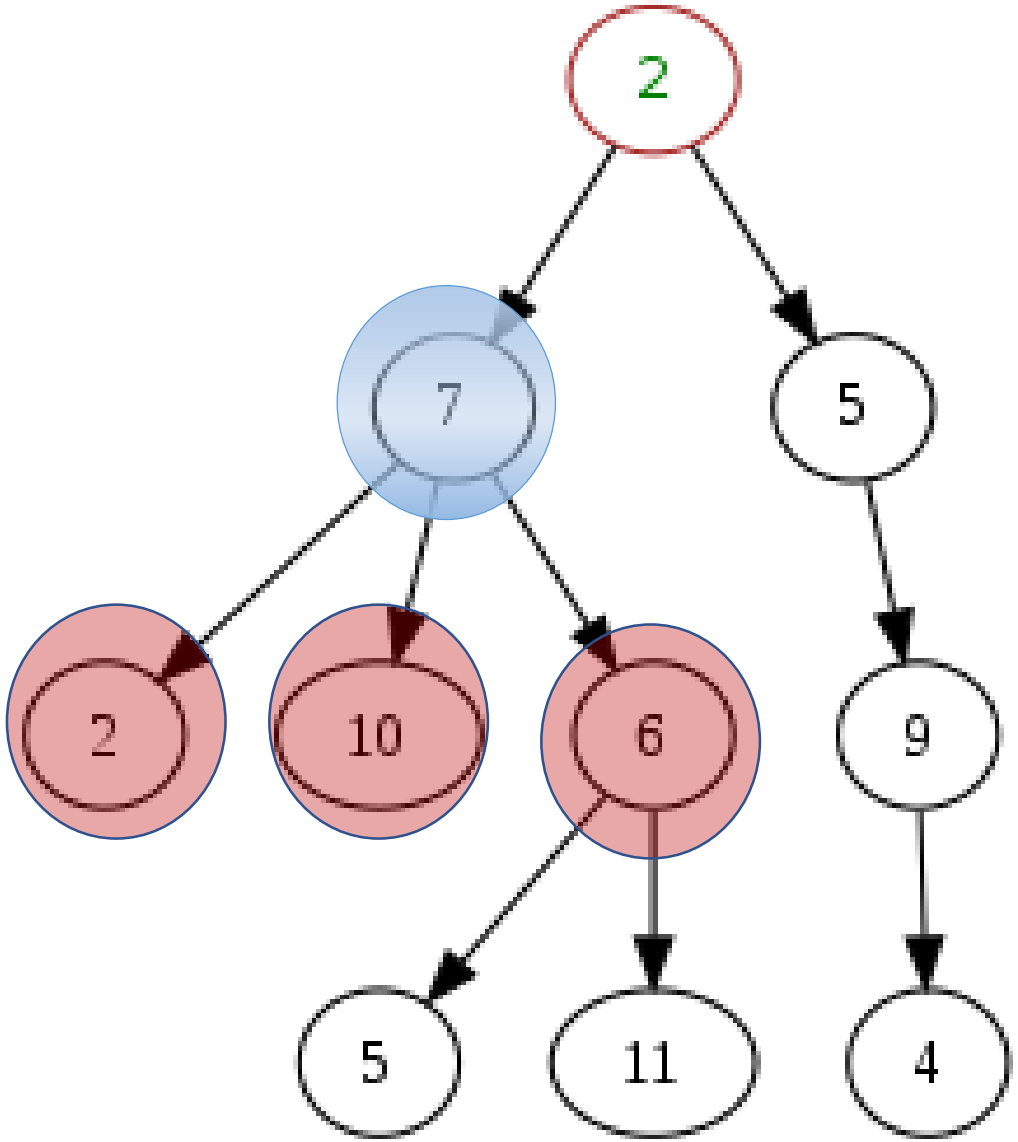
Terminology

- Node
- Edge
- Root
- Path
- ***Children***
- Parent
- Sibling
- Subtree
- Leaf Node



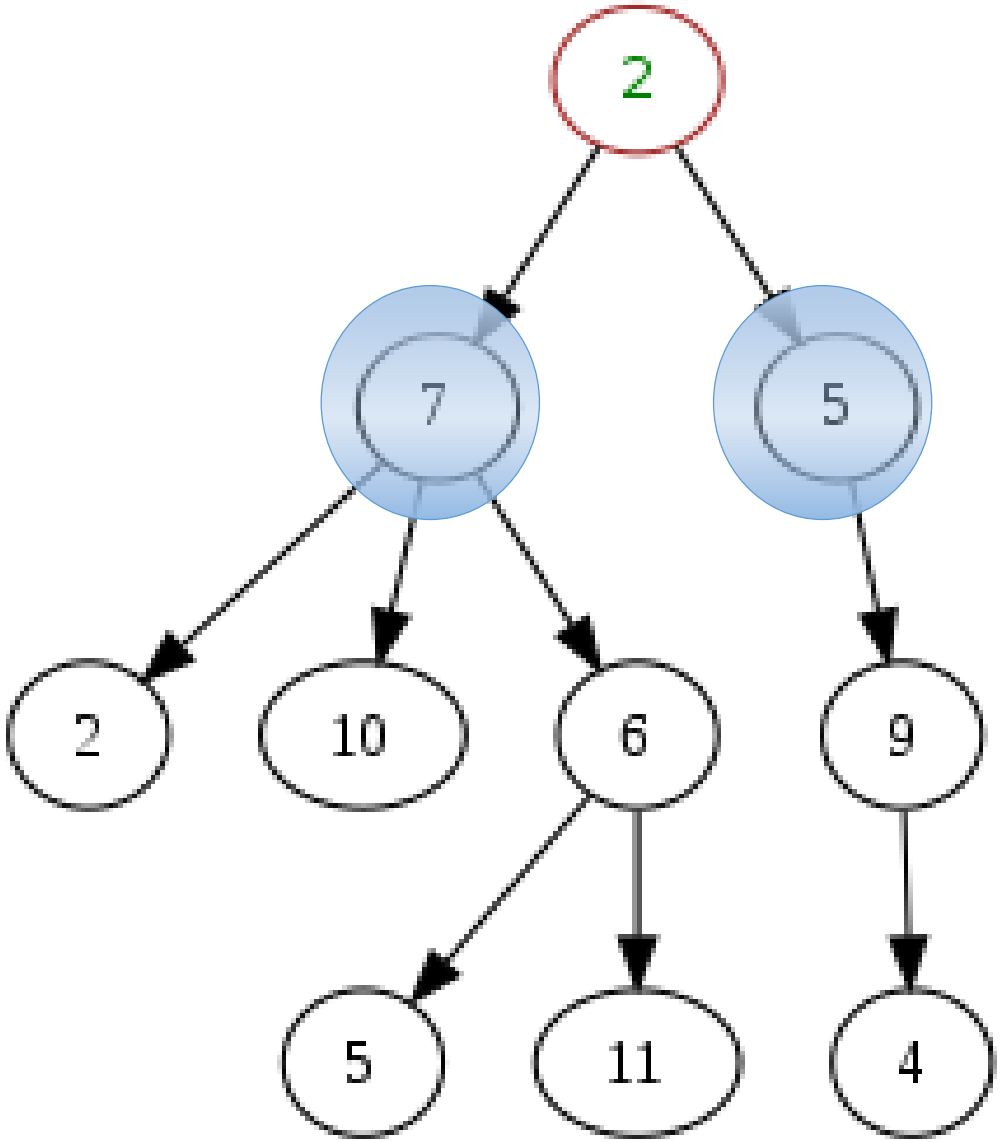
Terminology

- Node
- Edge
- Root
- Path
- Children
- ***Parent***
- Sibling
- Subtree
- Leaf Node



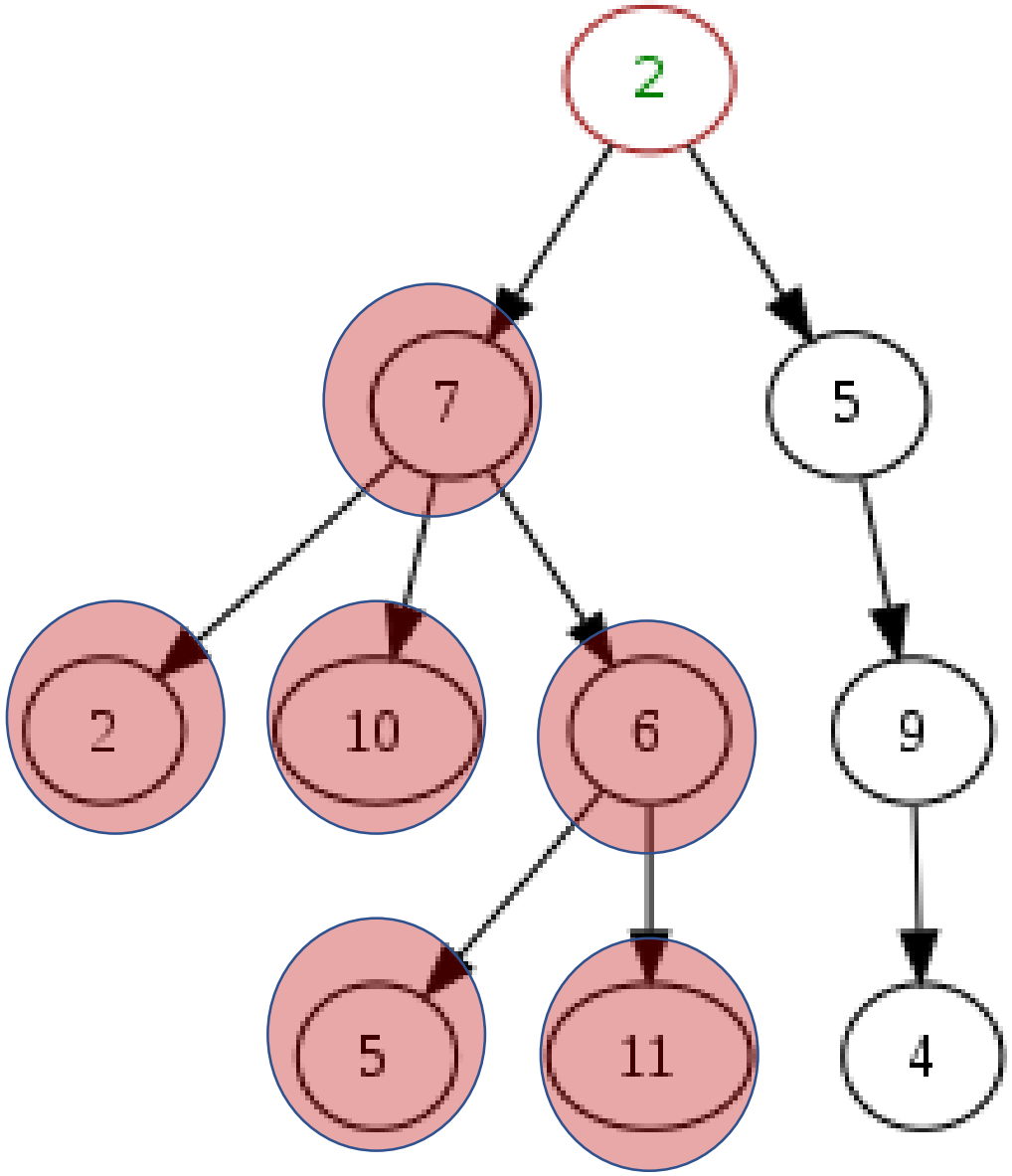
Terminology

- Node
- Edge
- Root
- Path
- Children
- Parent
- ***Sibling***
- Subtree
- Leaf Node



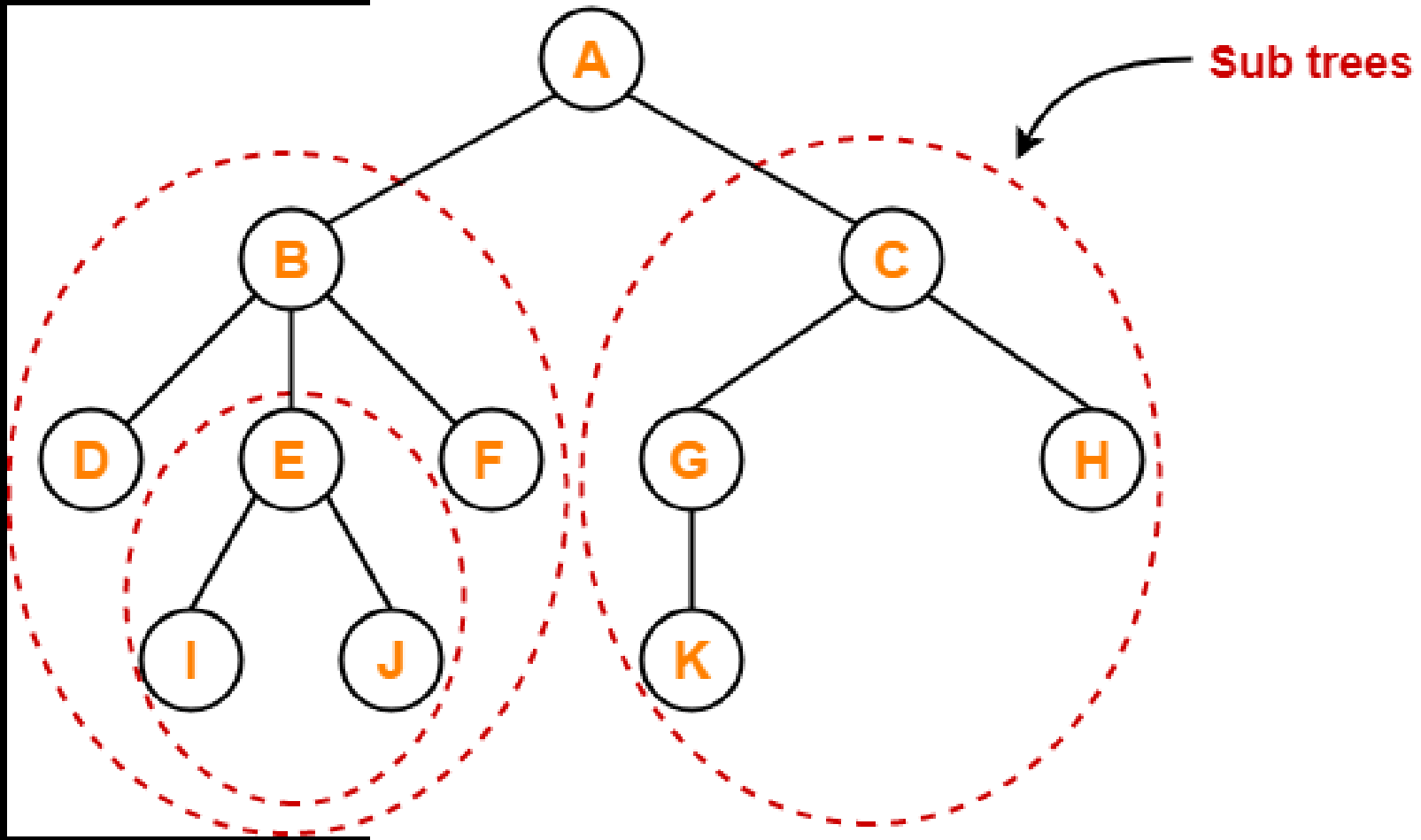
Terminology

- Node
- Edge
- Root
- Path
- Children
- Parent
- Sibling
- ***Subtree***
- Leaf Node



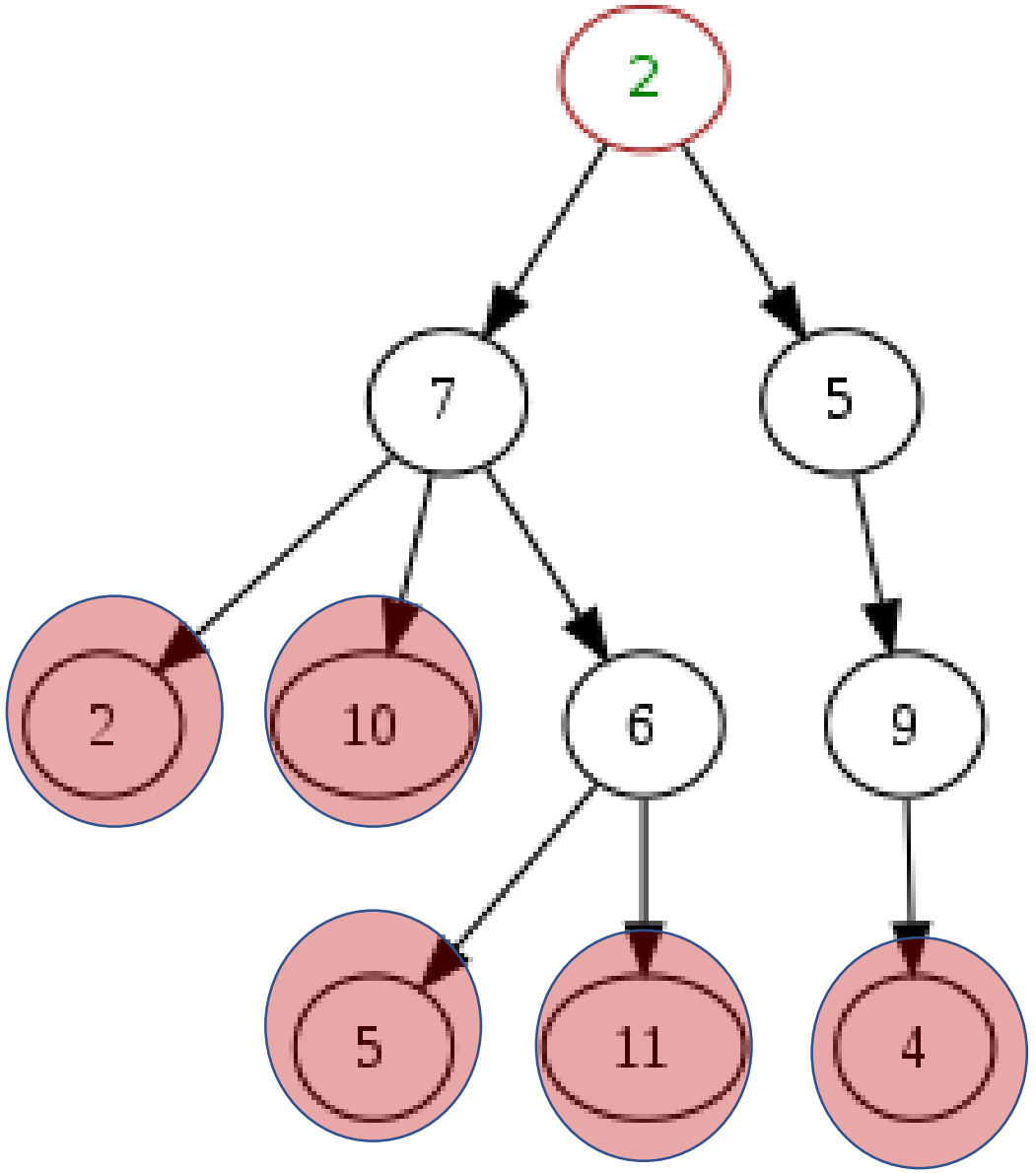
Terminology

- Node
- Edge
- Root
- Path
- Children
- Parent
- Sibling
- ***Subtree***
- Leaf Node

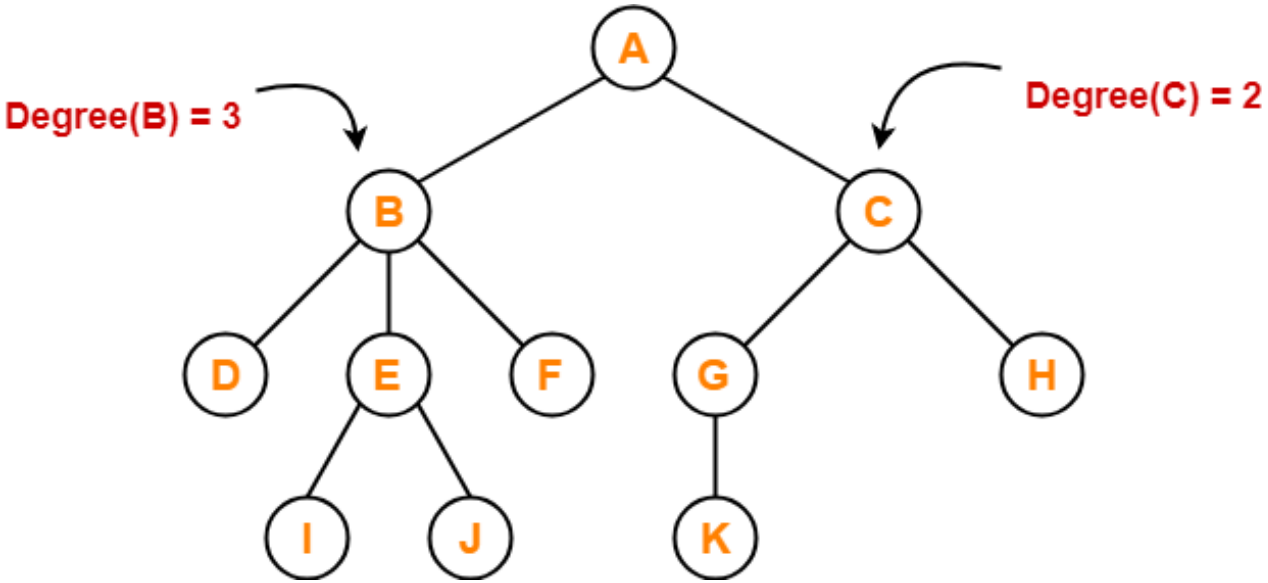


Terminology

- Node
- Edge
- Root
- Path
- Children
- Parent
- Sibling
- Subtree
- ***Leaf Node***

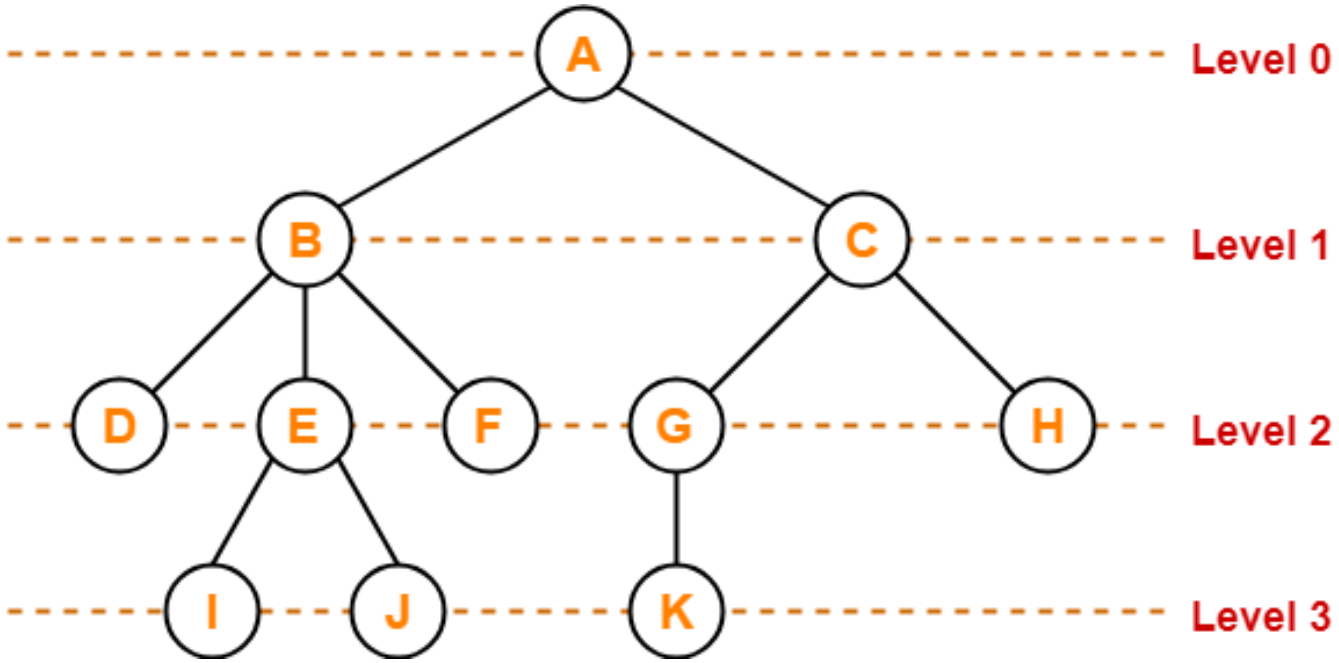


Degree



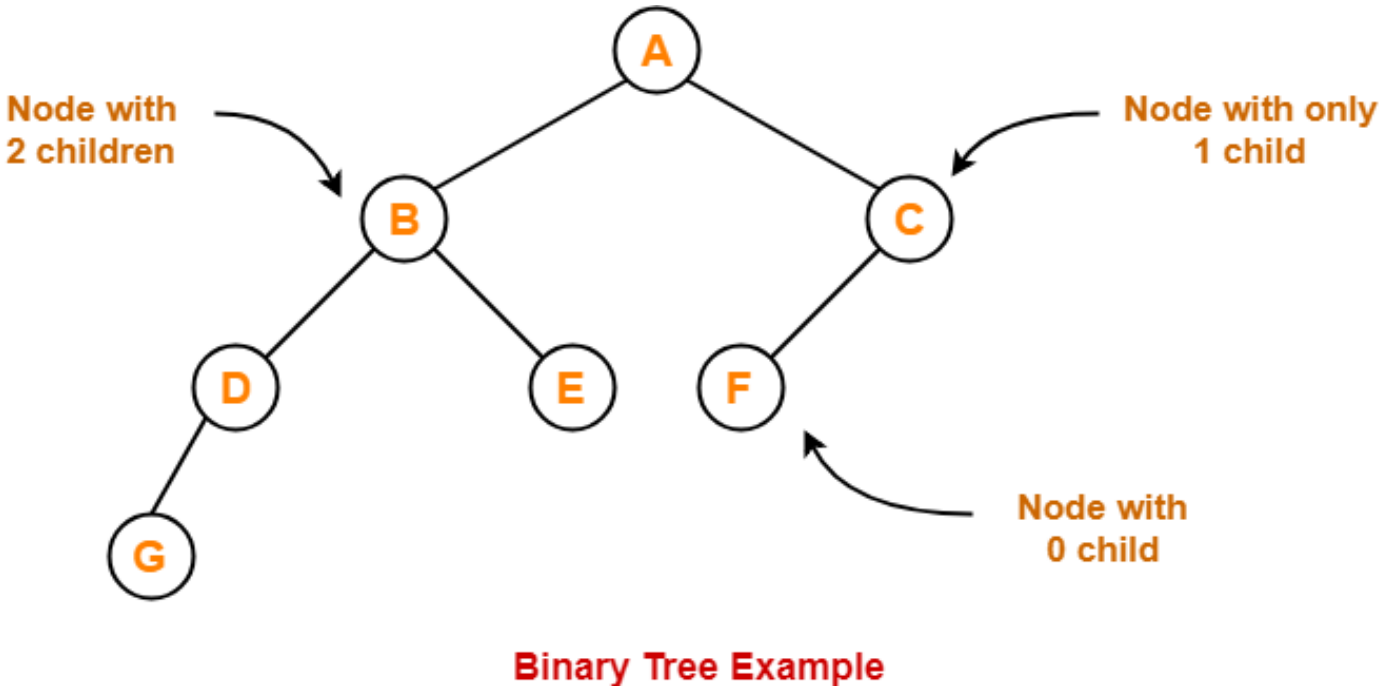
- **Degree of a node** is the total number of children of that node.
- **Degree of a tree** is the highest degree of a node among all the nodes in the tree.

Level



- In a tree, each step from top to bottom is called as **level of a tree**.
- The level count starts with 0 and increments by 1 at each level or step.

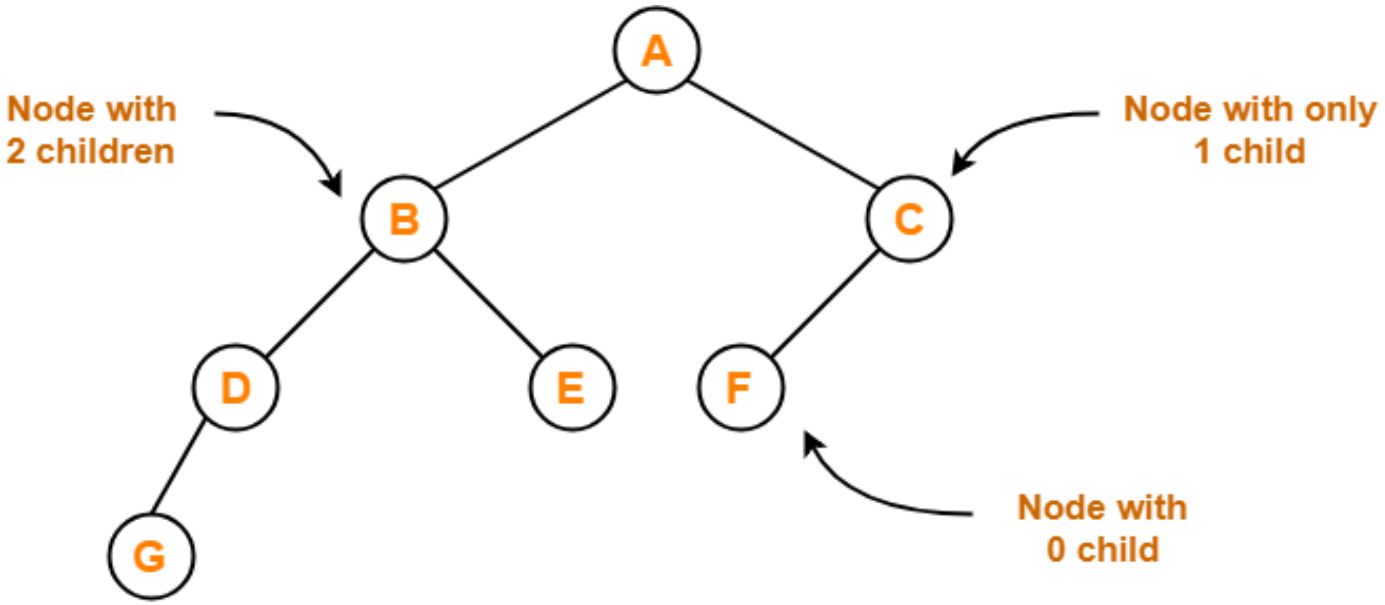
Binary Tree



Binary Tree Example

- Binary tree is a special tree data structure in which each node can have at most 2 children.
- Thus, in a binary tree, each node has either 0 child or 1 child or 2 children.

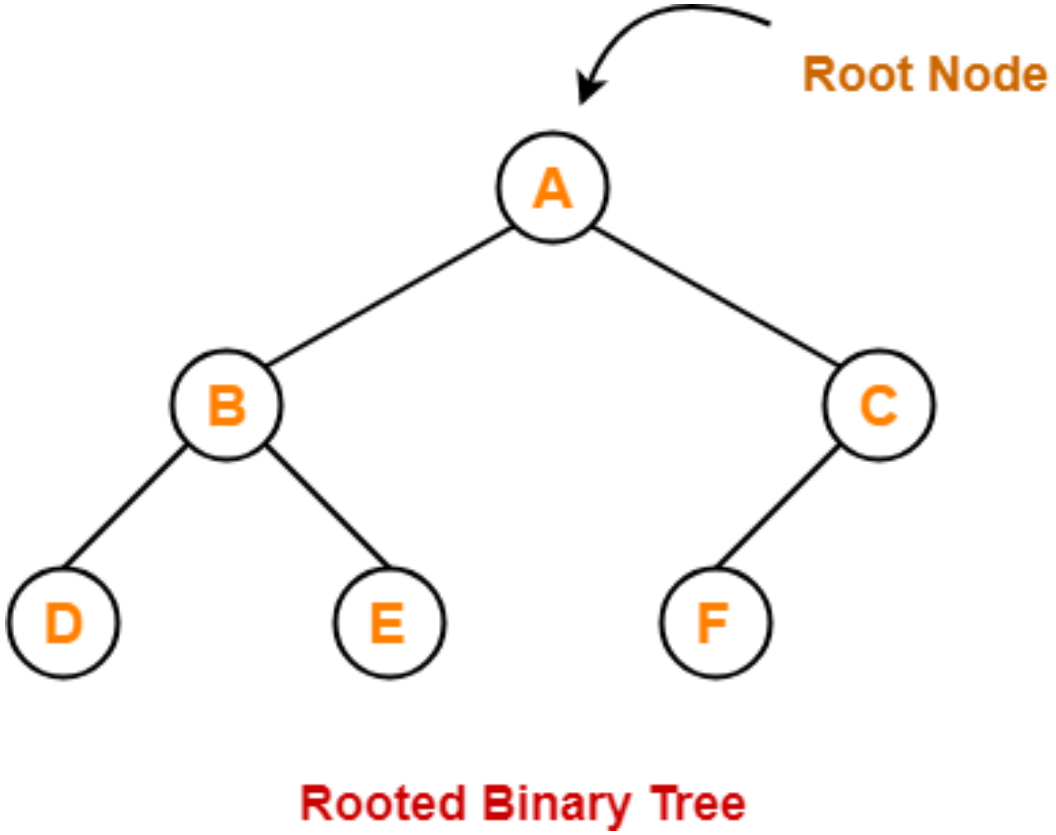
Types of Binary Trees



Binary Tree Example

- Rooted Binary Tree
- Full / Strictly Binary Tree
- Complete / Perfect Binary Tree
- Almost Complete Binary Tree
- Skewed Binary Tree

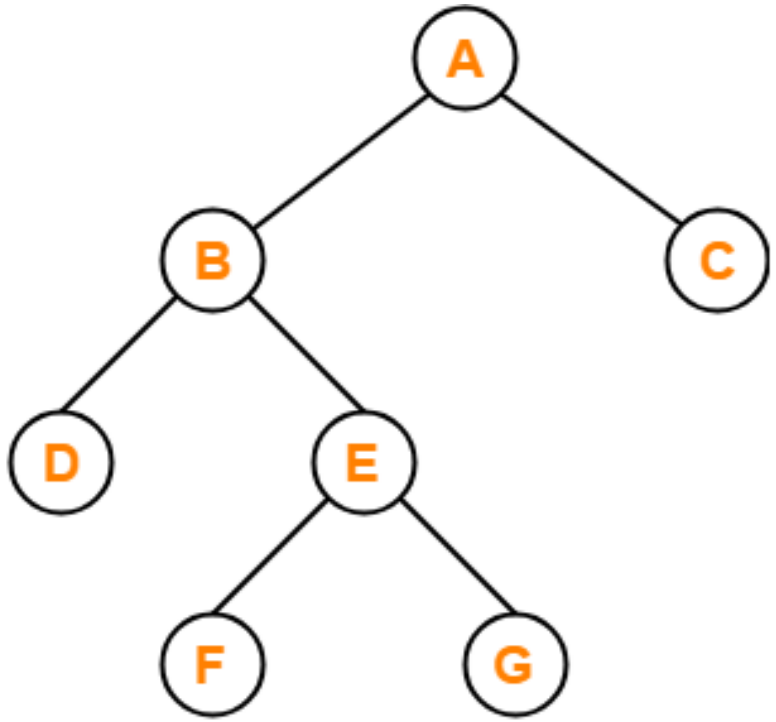
Types of Binary Trees



Rooted Binary Tree

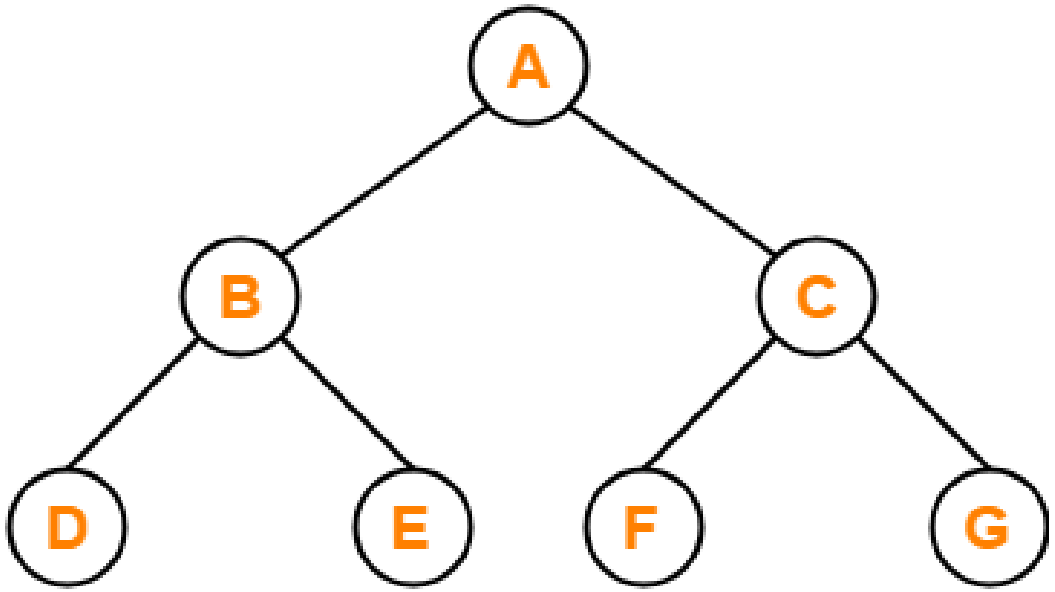
- Rooted Binary Tree
  - *It has a root node.*
  - *Each node has at most 2 children.*

Types of Binary Trees



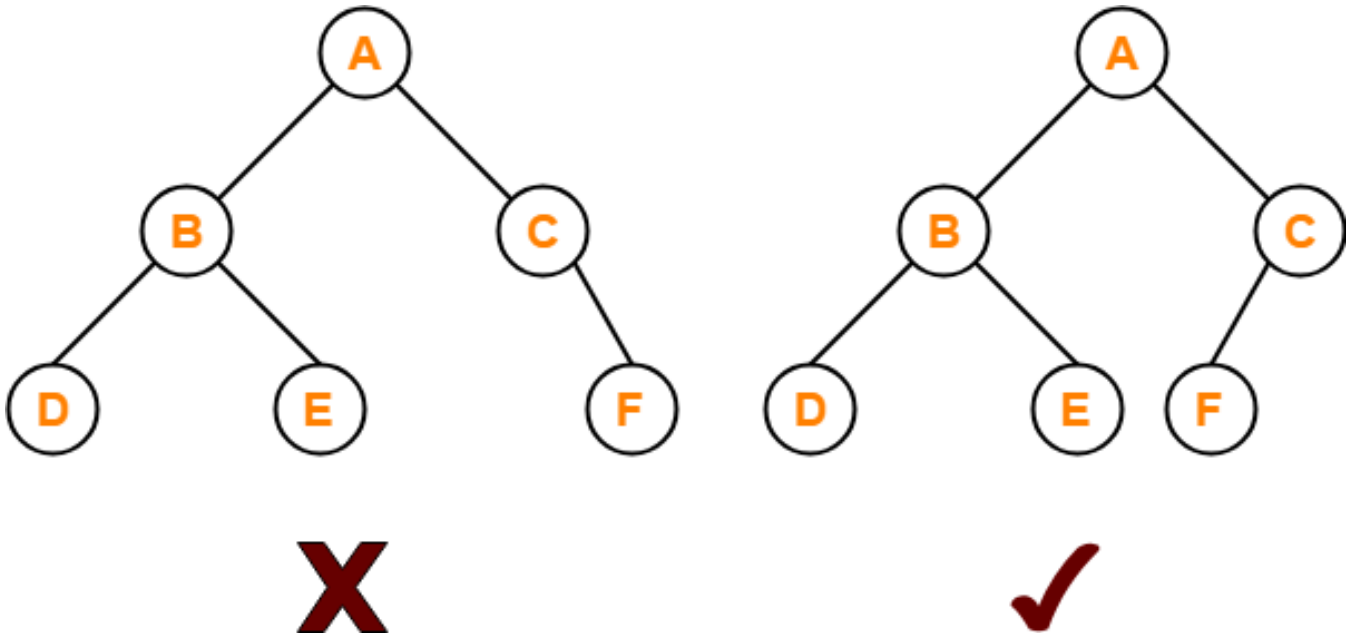
- **Full / Strictly Binary Tree**
  - A binary tree in which every node has either 0 or 2 children is called as a **Full binary tree**.

Types of Binary Trees



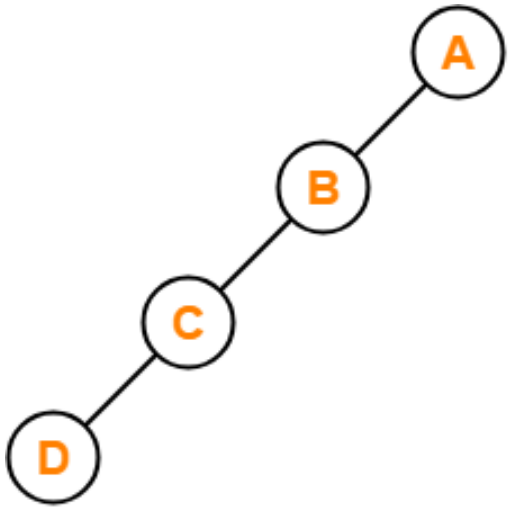
- **Complete / Perfect Binary Tree**
  - Every internal node has exactly 2 children.
  - All the leaf nodes are at the same level.

Types of Binary Trees

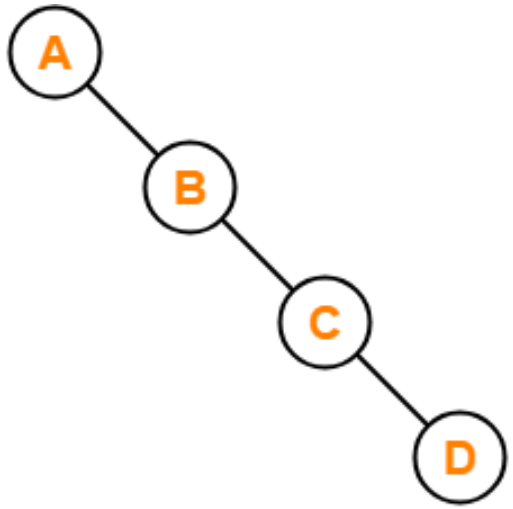


- **Almost Complete Binary Tree**
  - All the levels are completely filled except possibly the last level.
  - The last level must be strictly filled from left to right.

Types of Binary Trees



Left Skewed Binary Tree



Right Skewed Binary Tree

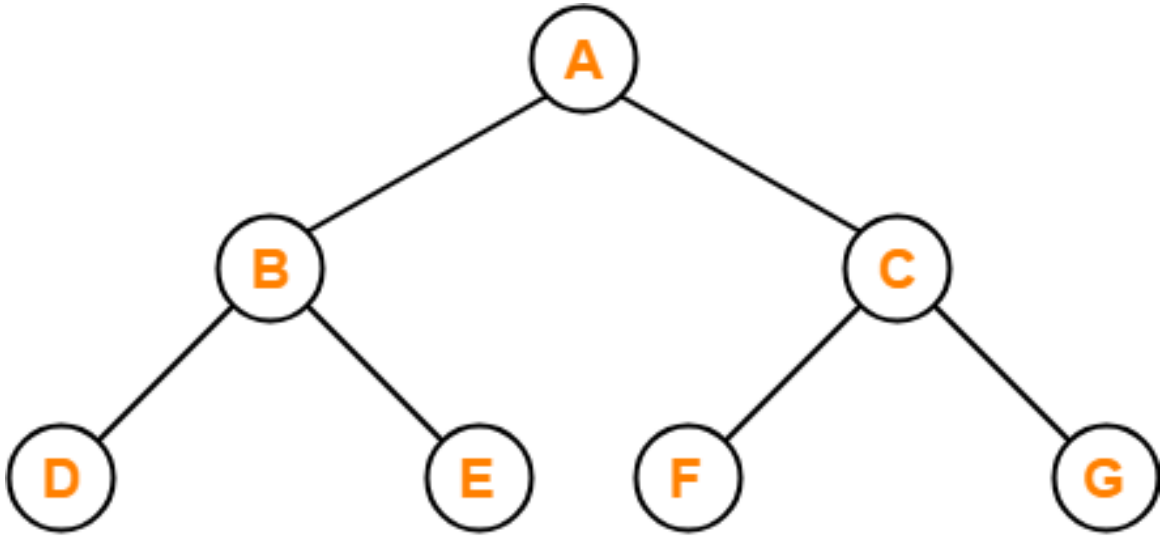
- **Skewed Binary Tree**
  - All the nodes except one node has one and only one child.
  - The remaining node has no child.

Tree Traversal

Tree Traversal Techniques

- **Depth First Traversal**
  - **Preorder Traversal**
  - **Inorder Traversal**
  - **Postorder Traversal**
- **Breadth First Traversal**

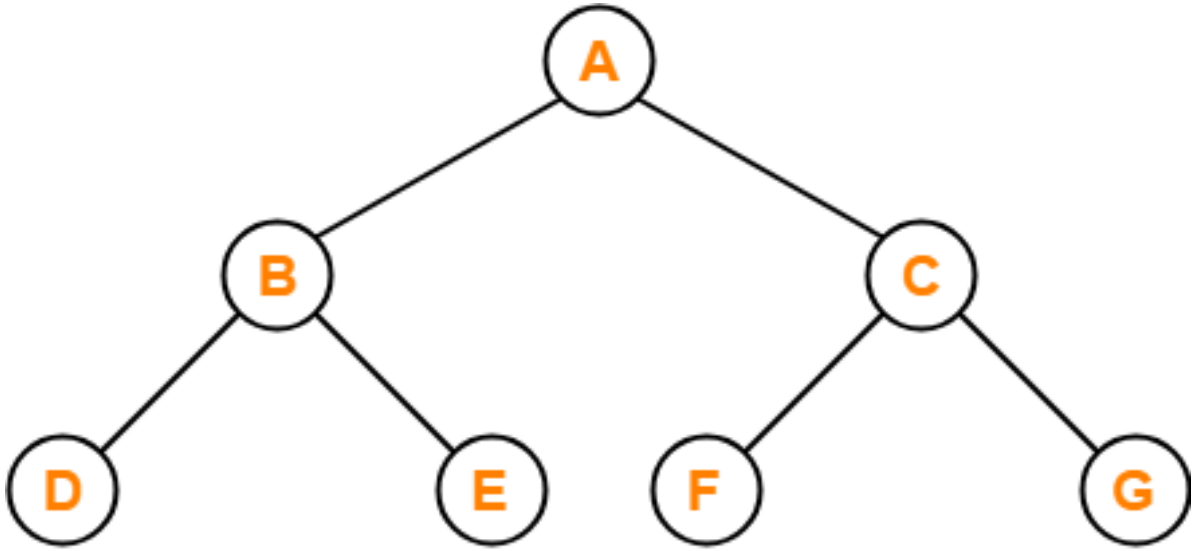
Preorder Traversal



Preorder Traversal : A , B , D , E , C , F , G

- Visit the root
- Traverse the left sub tree
- Traverse the right sub tree

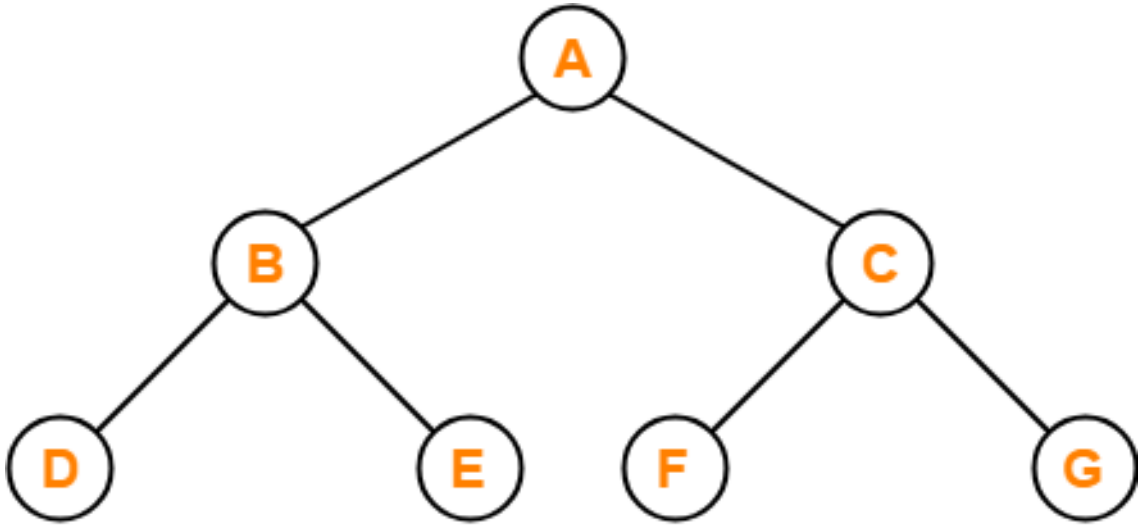
Inorder Traversal



Inorder Traversal : D , B , E , A , F , C , G

- Traverse the left sub tree
- Visit the root
- Traverse the right sub tree

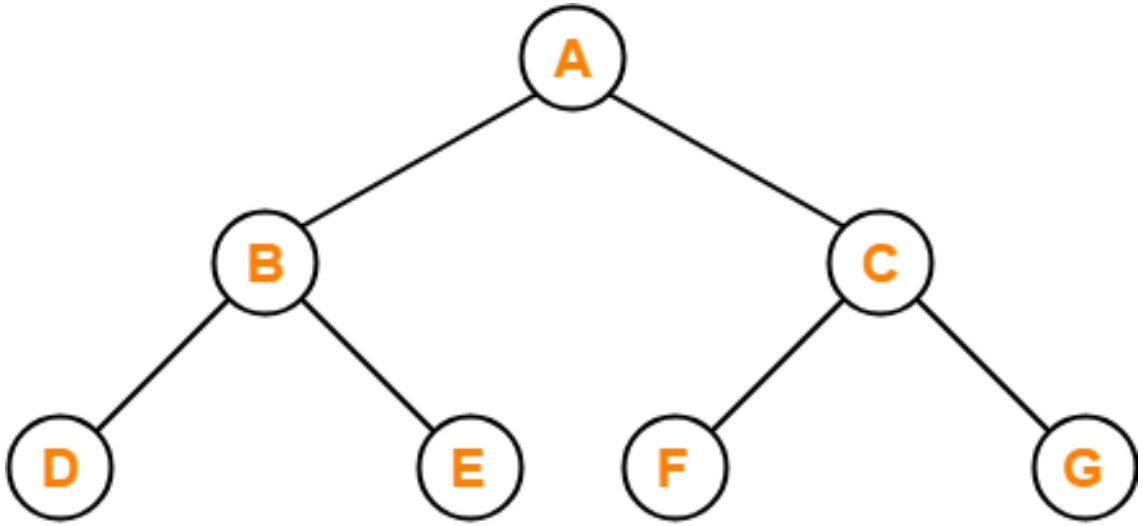
Postorder Traversal



Postorder Traversal : D , E , B , F , G , C , A

- Traverse the left sub tree
- Traverse the right sub tree
- Visit the root

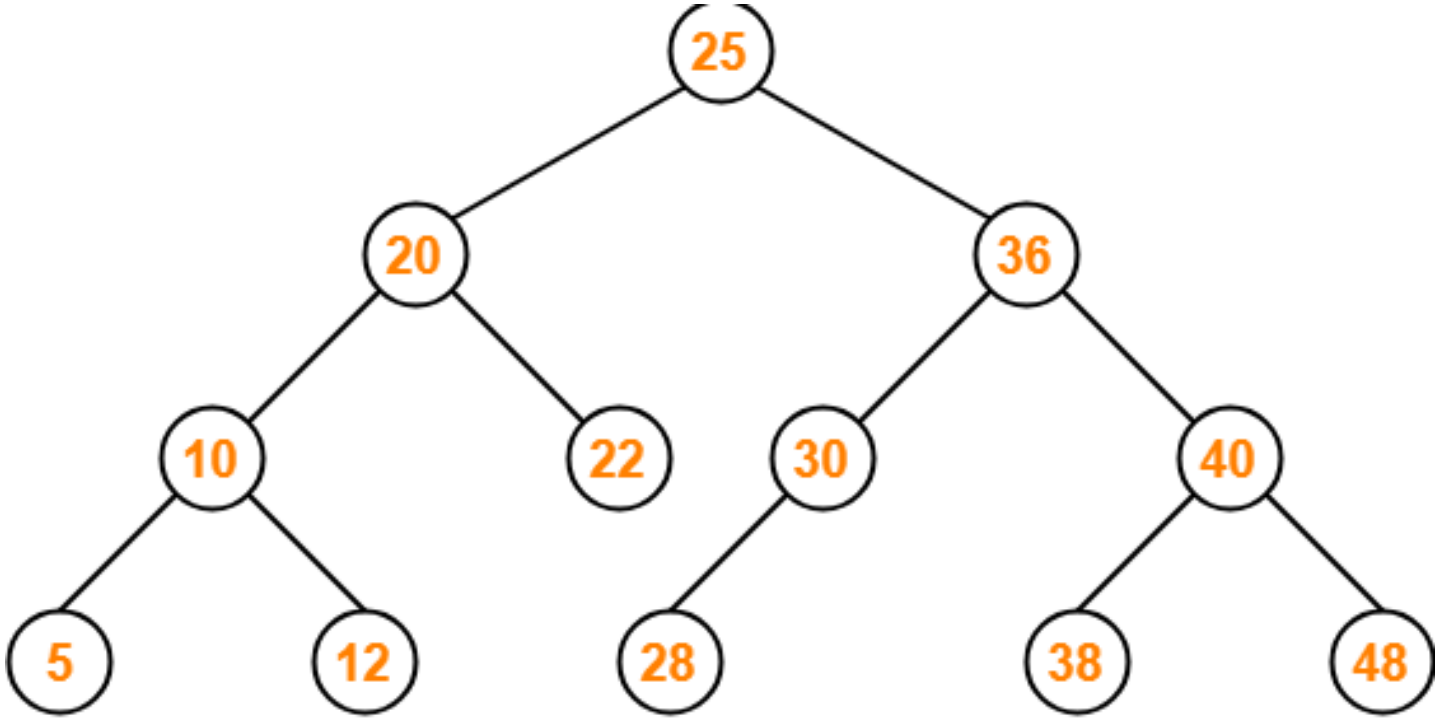
Breadth First Traversal



Level Order Traversal : A , B , C , D , E , F , G

- Breadth First Traversal of a tree prints all the nodes of a tree level by level.
- Breadth First Traversal is also called as **Level Order Traversal**

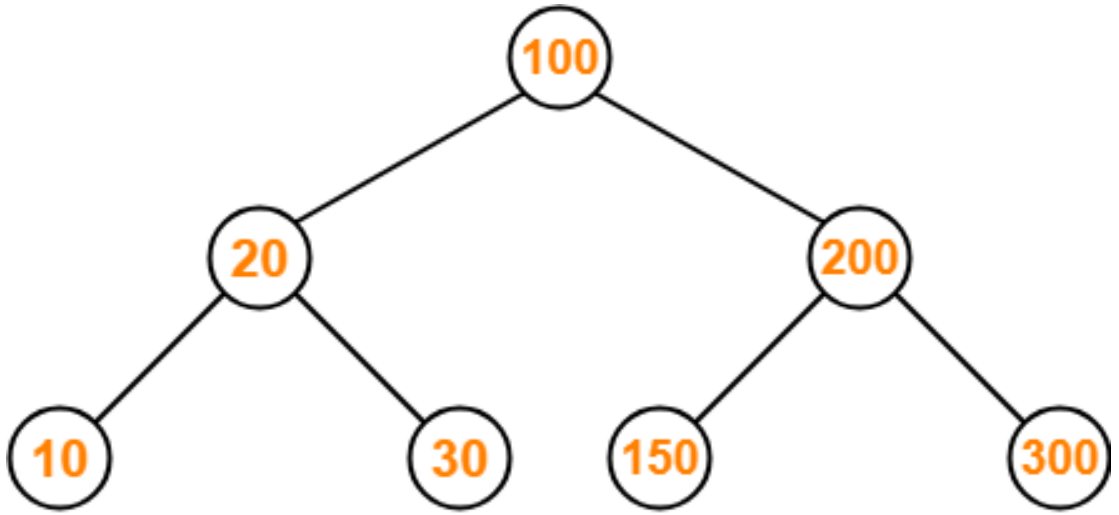
Binary Search Tree



Binary Search Tree

- Smaller values in its left sub tree
- Larger values in its right sub tree

BST Traversal



Binary Search Tree

- **Preorder Traversal-**

  100 , 20 , 10 , 30 , 200 , 150 , 300

- **Inorder Traversal-**

  10 , 20 , 30 , 100 , 150 , 200 , 300

- **Postorder Traversal-**

  10 , 30 , 20 , 150 , 300 , 200 , 100