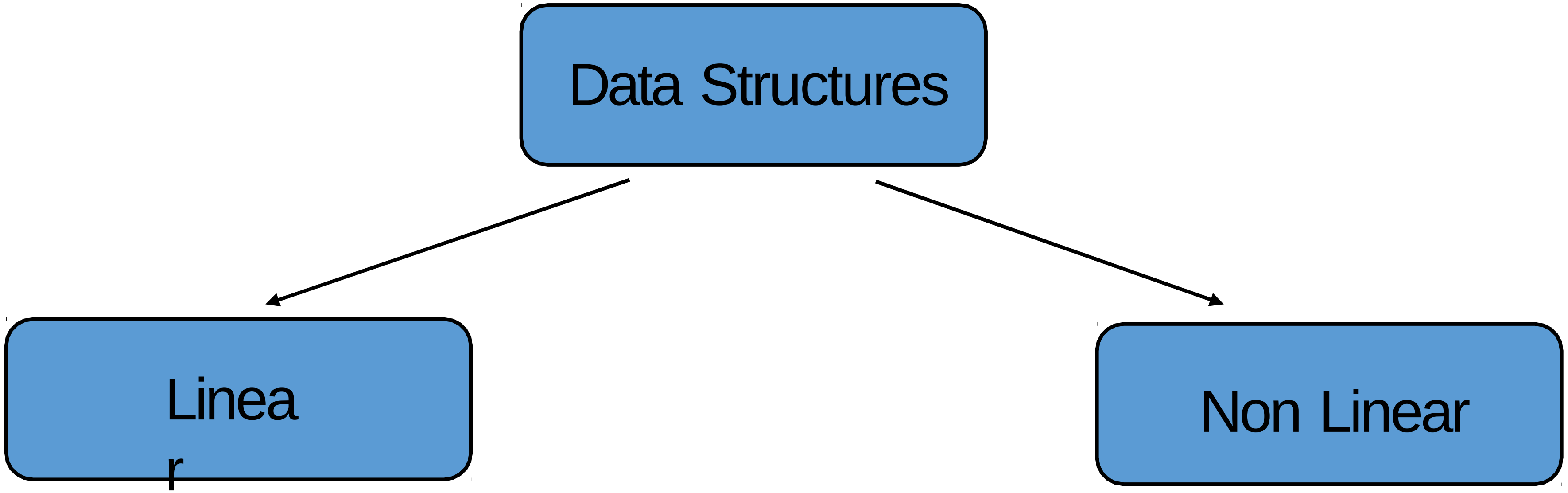
TREES

TYPES OF DATA STRUCTURE



- Array
- Linked List
- Stack
- Queue

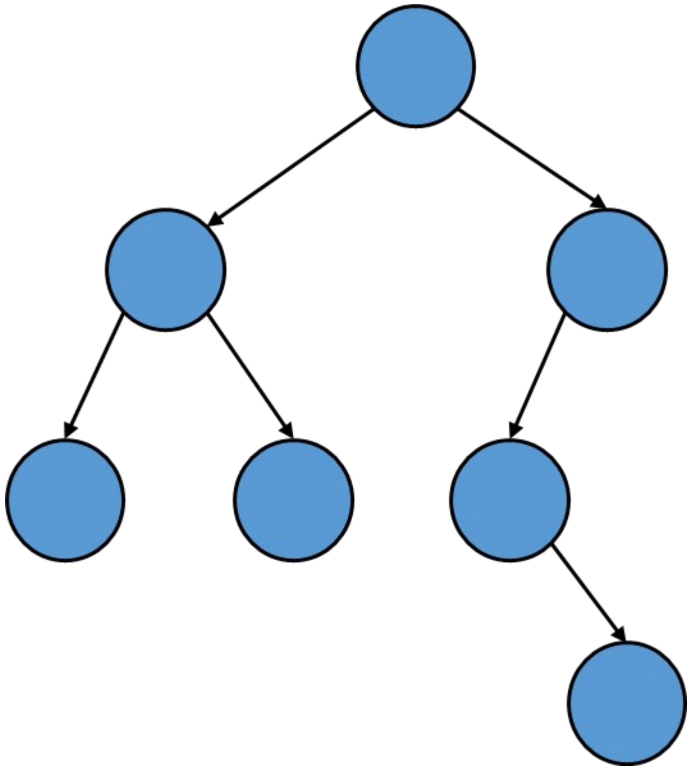
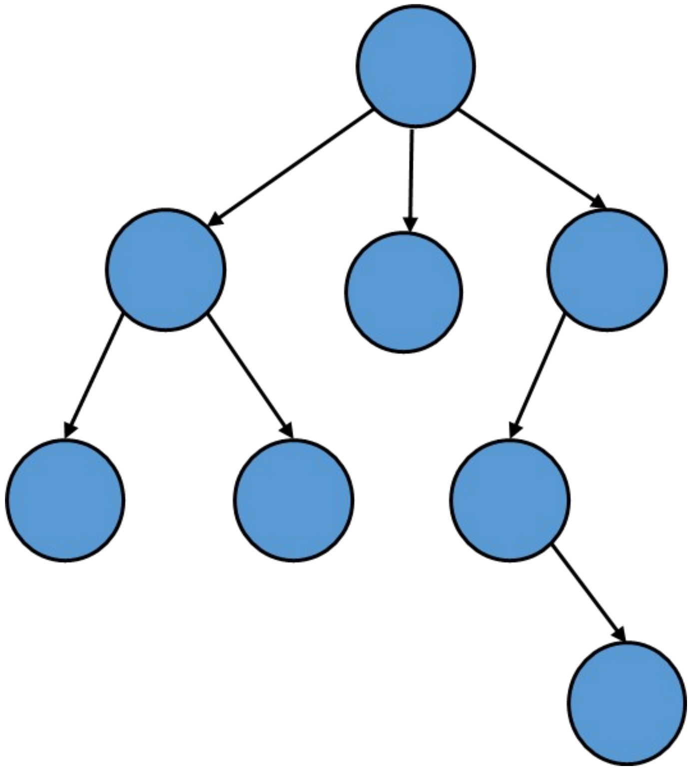
- Trees
- Graph

Tree

- A tree is a data structure made up of nodes or vertices and edges without having any cycle.

N-ary tree:

is a tree in which nodes can have **at most N** children.

TYPES OF NODES

A tree would have **three** types of nodes.

1. **Root Node**
2. **Leaf Node/External node**
3. **Intermediate Node/Internal node**

Root node: is the top most node in a tree.

Leaf node: is a node with no children.

Intermediate node: are nodes which have both incoming and outgoing edges.

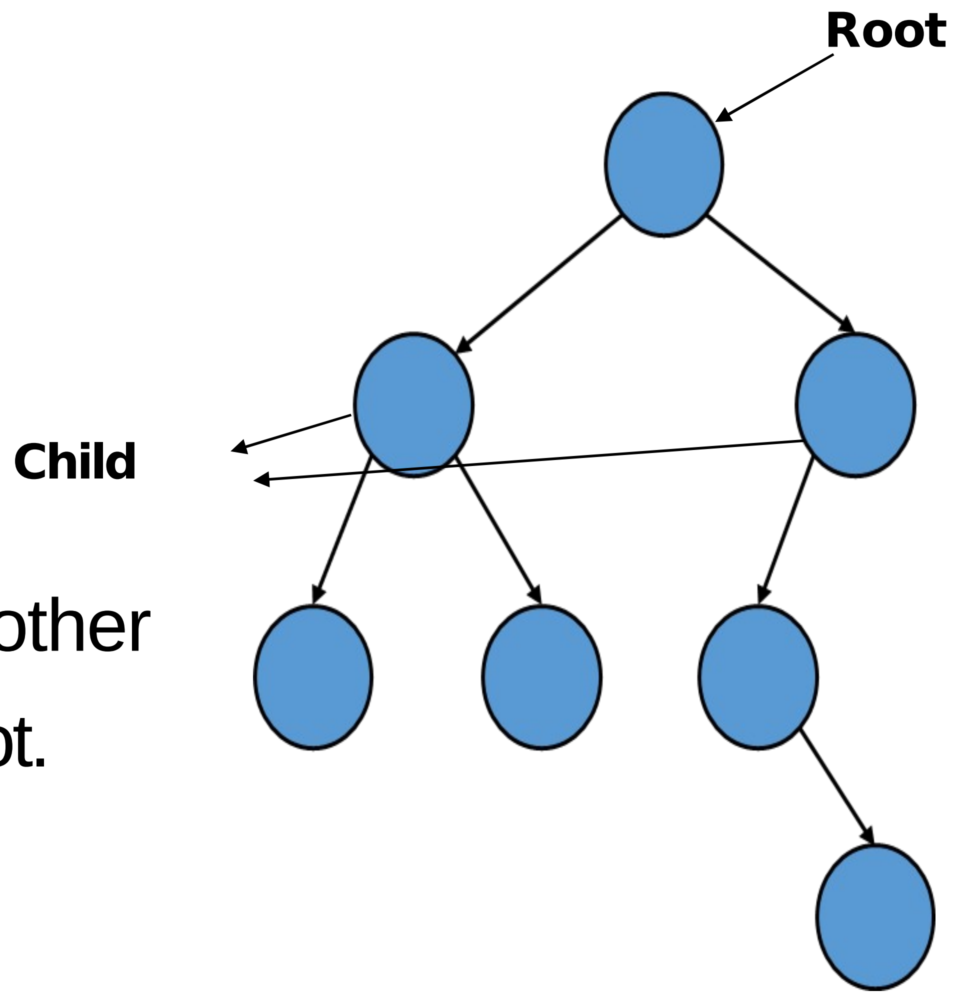
TERMINOLOGIES

- **Root**

  The top node in a tree.

- **Child**

  A node directly connected to another node when moving away from the Root.



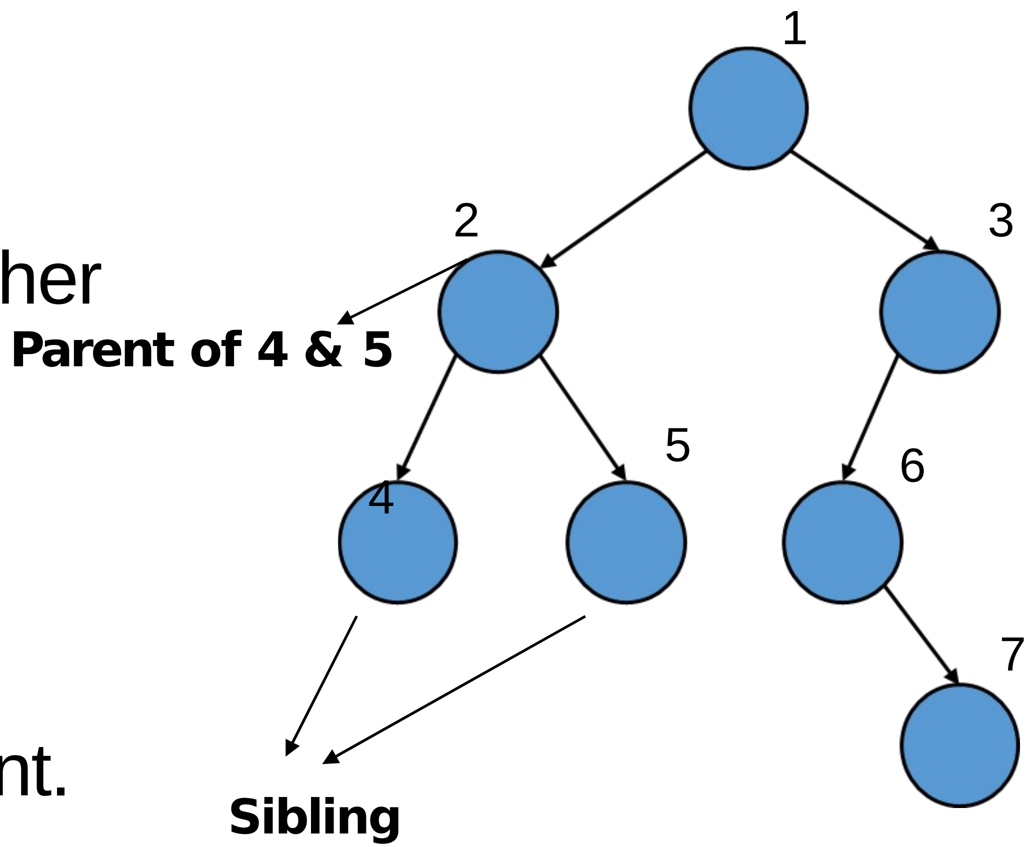
TERMINOLOGIES

- **Parent**

    A node directly connected to another node when moving towards the Root.

- **Siblings**

    A group of nodes with the same parent.



TERMINOLOGIES

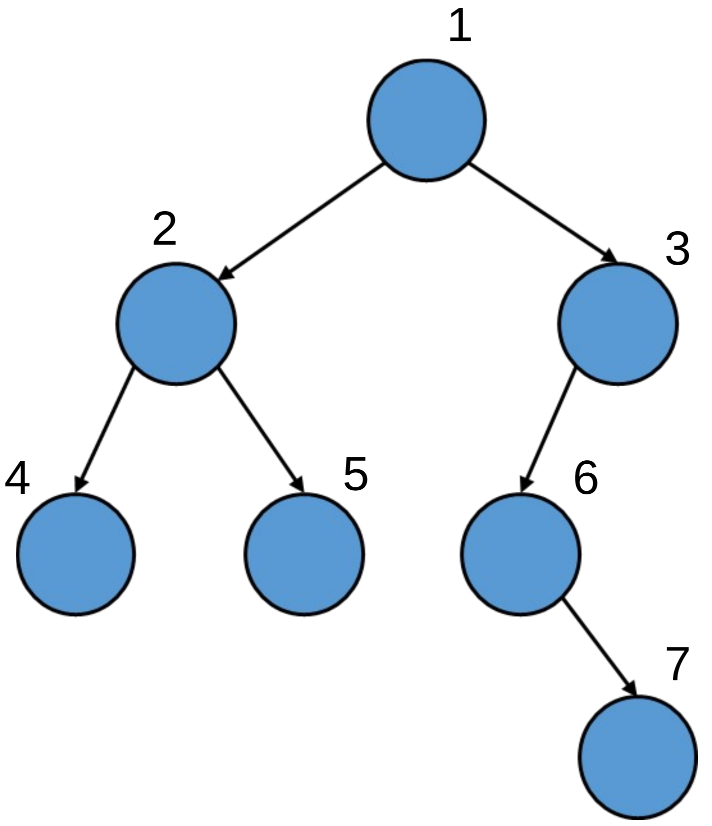
- **Descendant**

  A node reachable by repeated proceeding from parent to child.

- **Ancestor**

  A node reachable by repeated proceeding from child to parent.

Example:



Node 1 is the *ancestor* of 5
Node 7 is the *descendant* of 3

TERMINOLOGIES

- **Cousins**

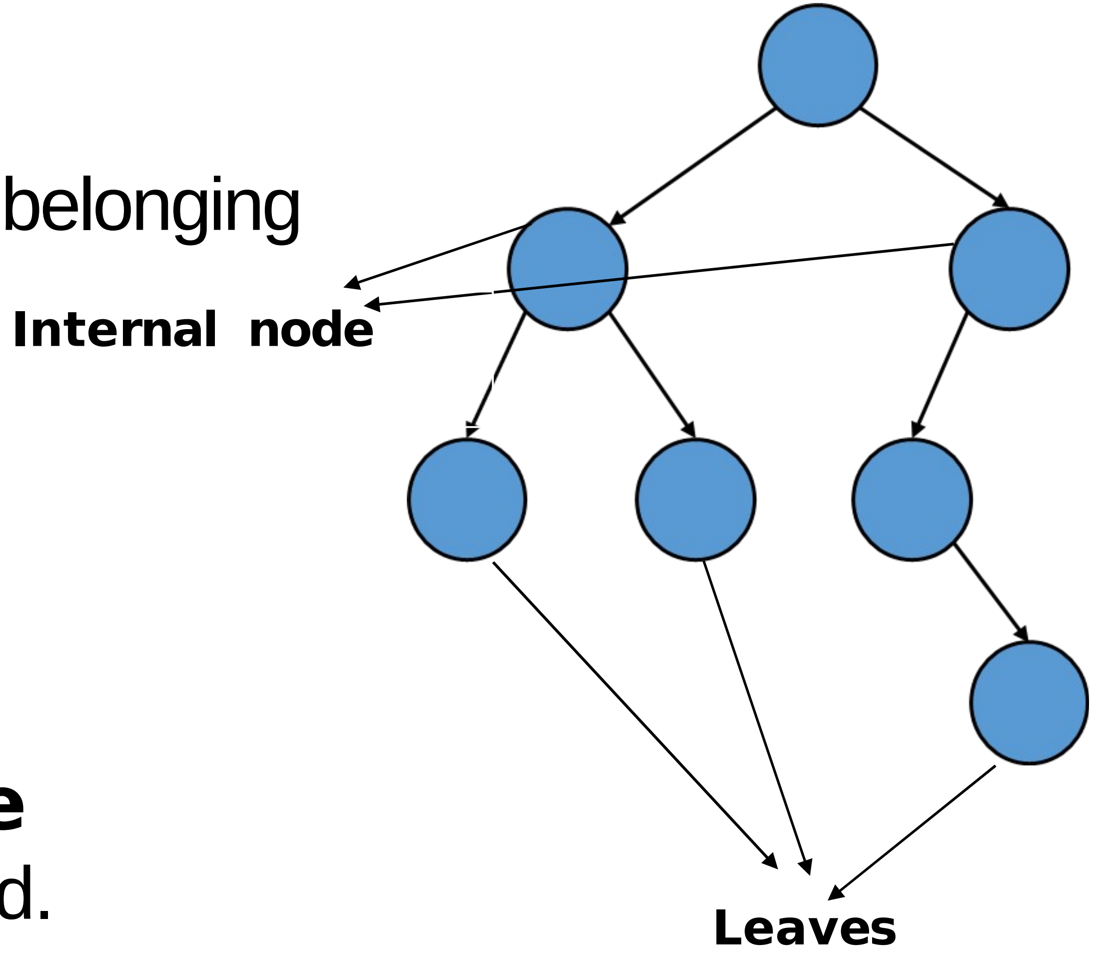
  Nodes at the same level but belonging to different parent.

- **Leaf/External node**

  A node with no children.

- **Intermediate/Internal node**

  A node with at least one child.



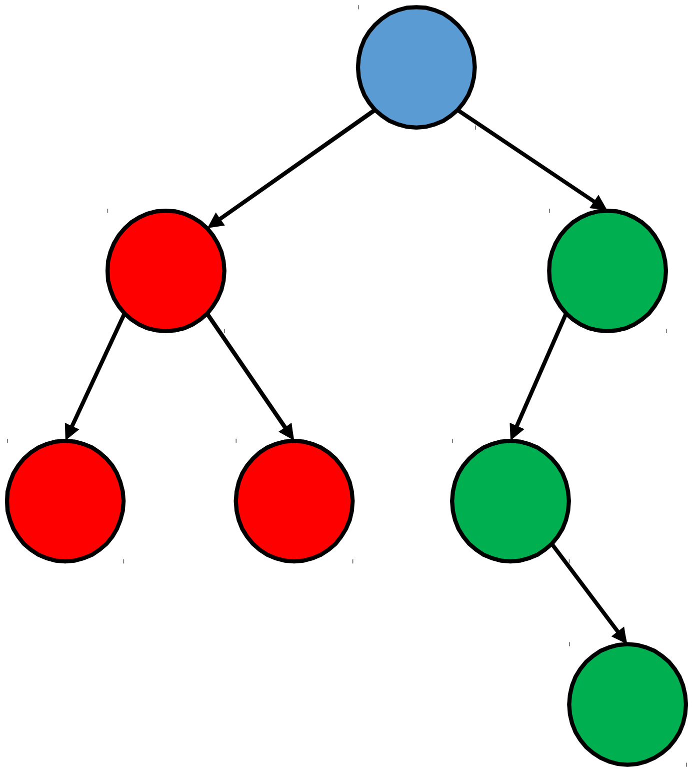
TERMINOLOGIES

- **Left subtree**

  All the nodes towards left side of a node are called left subtree.

- **Right subtree**

  All the nodes towards right side of a node are called right subtree.



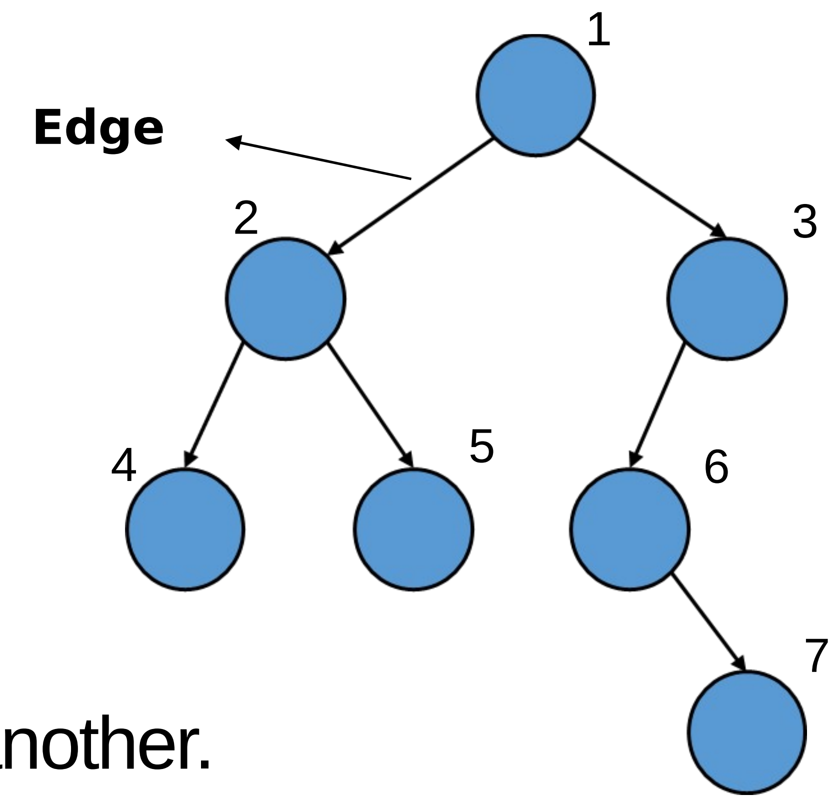
TERMINOLOGIES

- **Degree**

  The number of sub trees of a node.

- **Edge**

  The connection between one node and another.



Degree of node 1 is 2
Degree of node 3 is 1
Degree of node 6 is 1

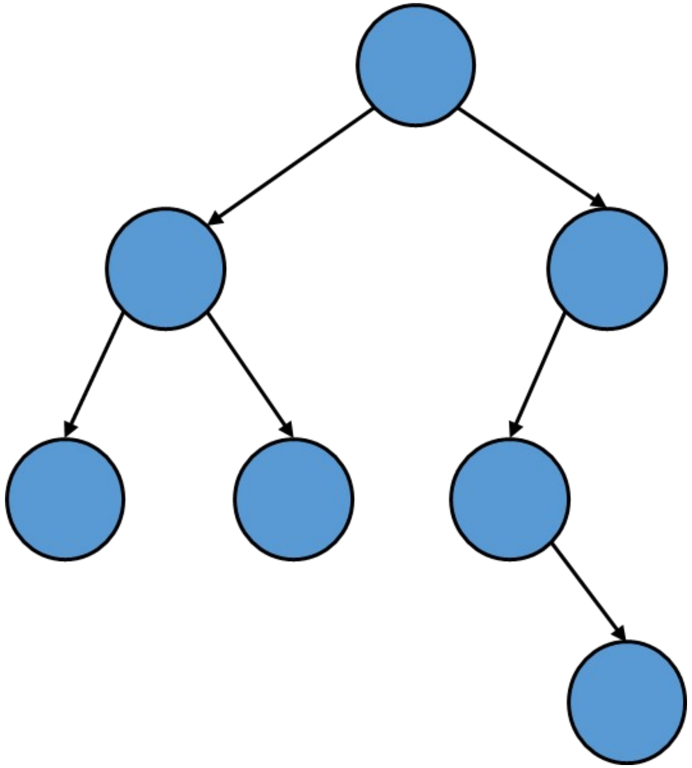
TERMINOLOGIES

- **Level**

  A sequence of nodes and edges connecting a node with a descendant.

- **Forest**

  A forest is a set of $n \geq 0$ disjoint trees.



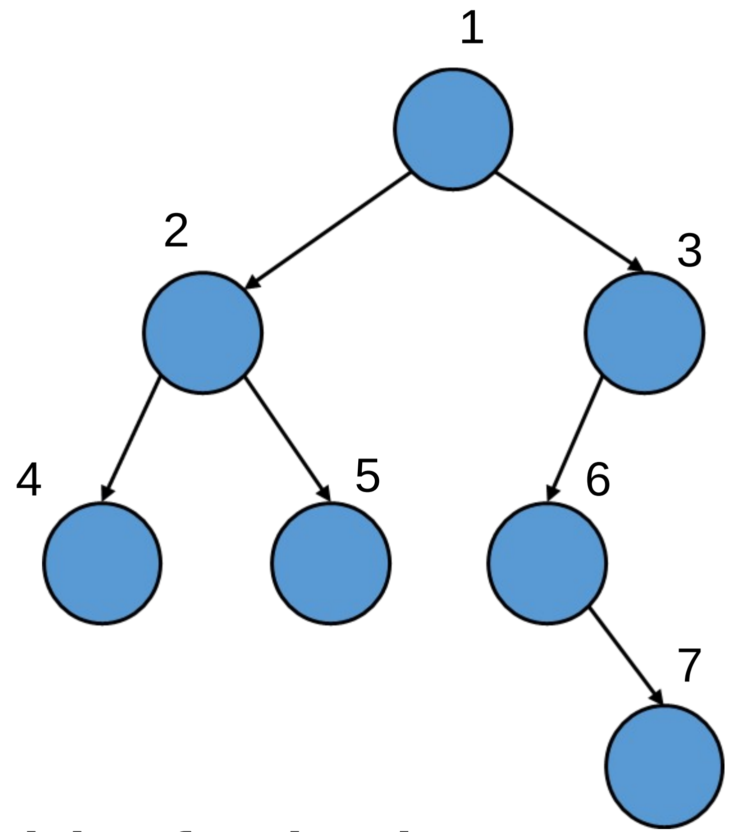
TERMINOLOGIES

- **Height of a node**

  The height of a node is the number of edges on the longest path between that node and a leaf.

- **Height of a tree**

  Maximum level of the tree.



Height of node 2 is 1
Height of node 3 is 2

Height of the tree is 3

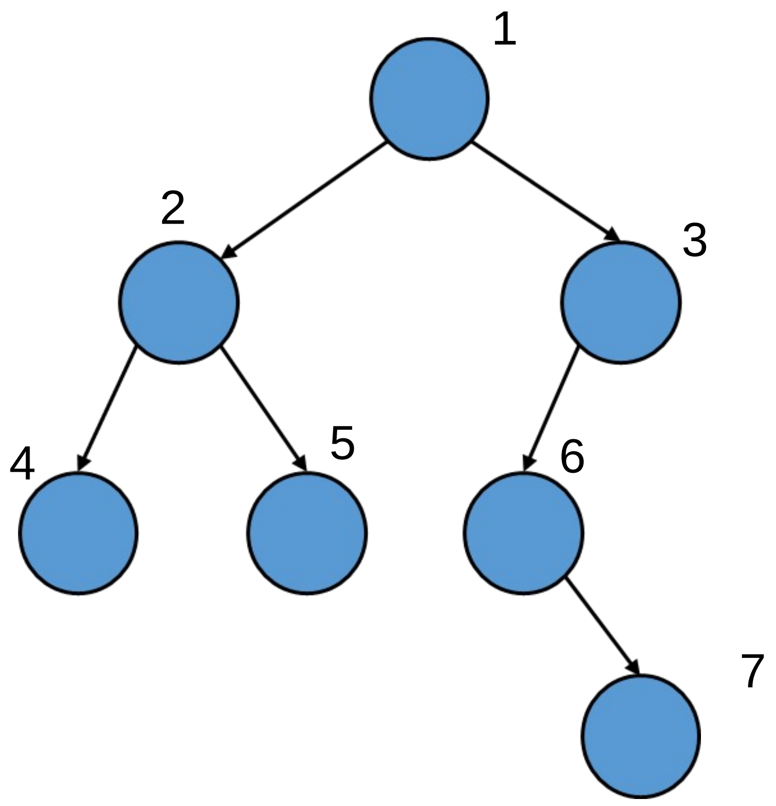
TERMINOLOGIES

- **Depth of a node**

  The depth of a node is the number of edges from the tree's root node to a particular node

- **Depth of a tree**

  Maximum level of the tree.



Depth of node:
Depth of node 6 is 2.
Dept of node 2 is 1.

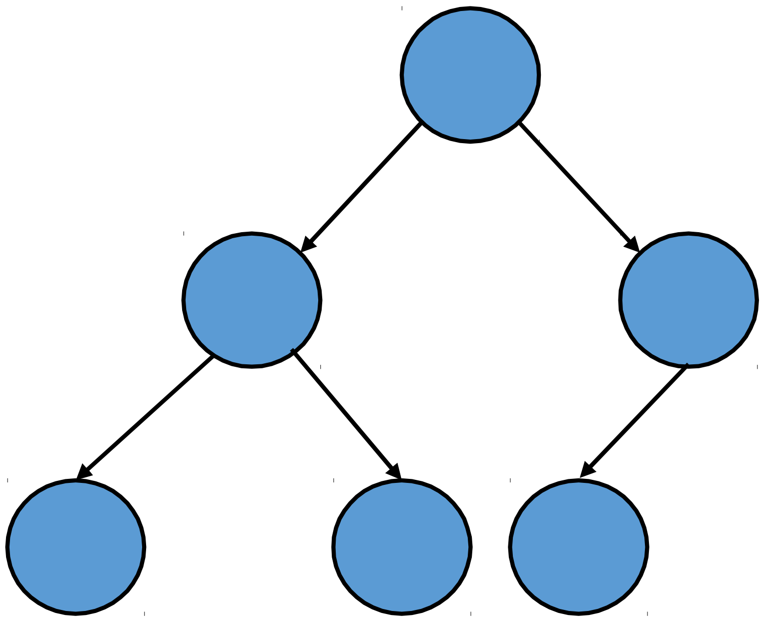
Depth of the tree:
Depth of the tree is 3

TYPES OF TREES

- Binary tree
- Quad tree
- Oct tree
- Binary Search tree
- AVL tree
- Red Black tree
- Splay tree
- Trie
- Huffman tree
- Heap tree
- B tree
- B+ tree

Binary Tree

Is a tree where each node has at most 2 children.



(a)

- N-ary tree:

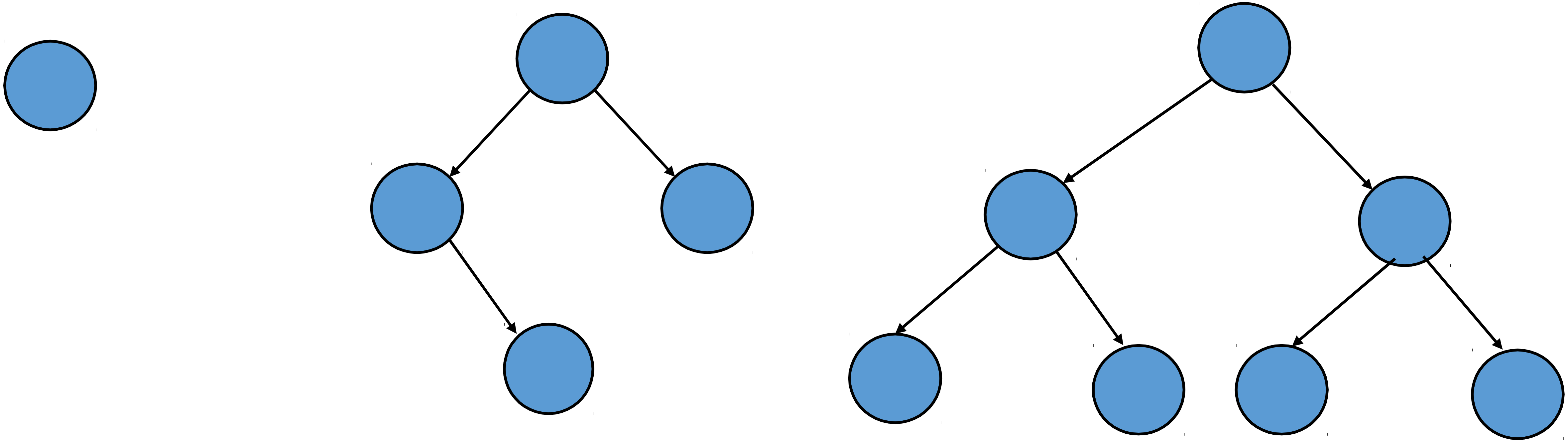
  is a tree in which nodes can have **at most N** children.

  Quad tree: is a tree in which nodes can have **at most 4** children.

  Oct tree: is a tree in which nodes can have **at most 8** children.

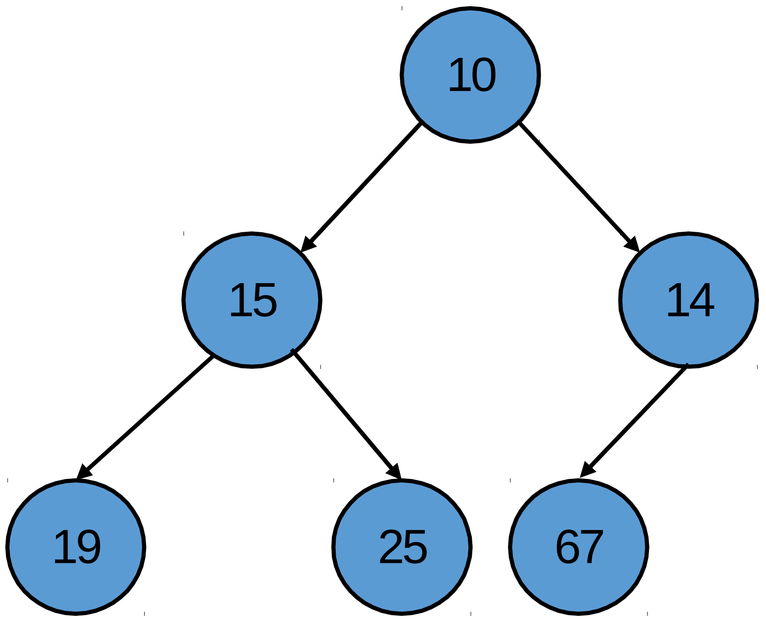
  Binary tree: is a tree in which nodes can have **at most 2** children.

Examples of Binary Tree

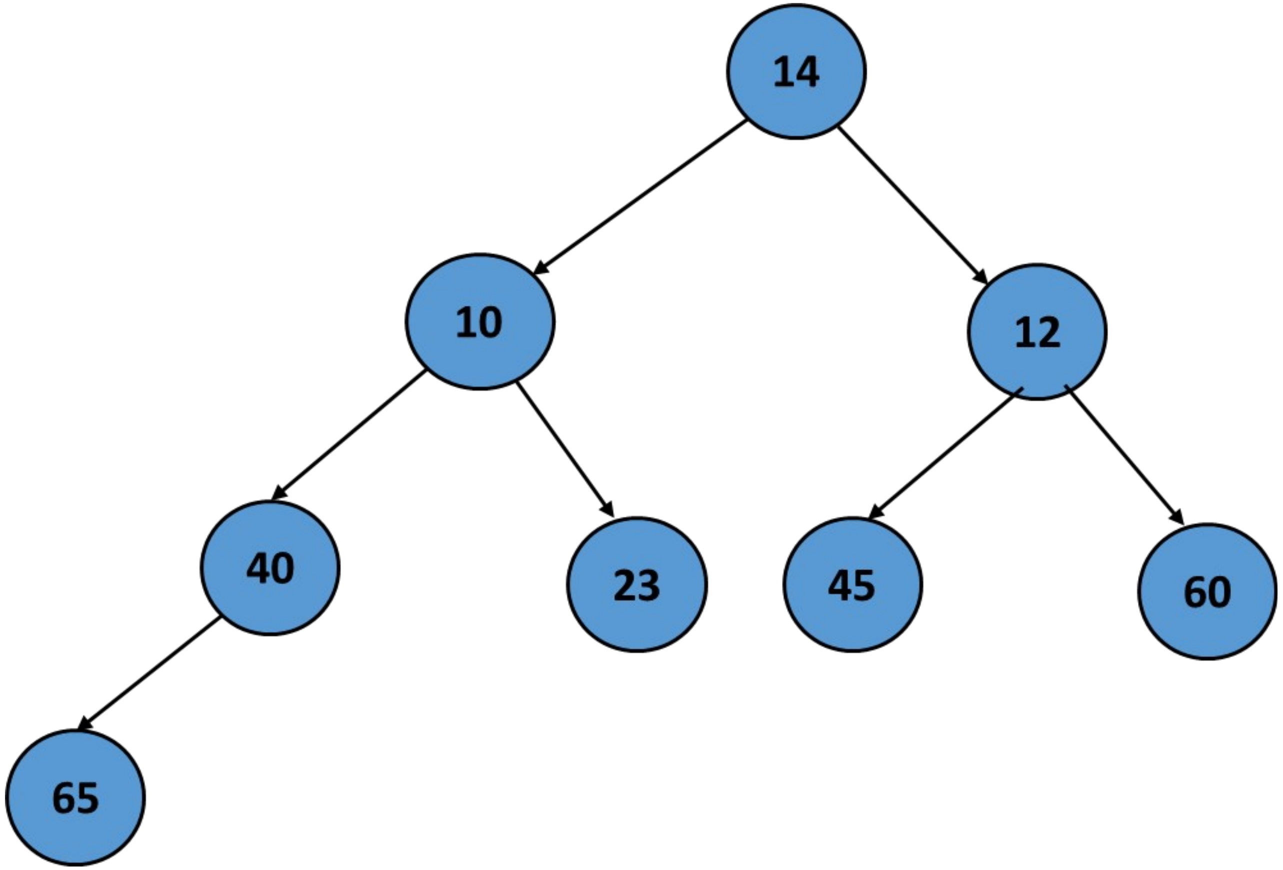


CONSTUCTION OF A BINARY TREE

1) **10, 15, 14, 19, 25, 67**



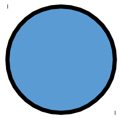
2) **14, 10, 12, 40, 23, 45, 60, 65**



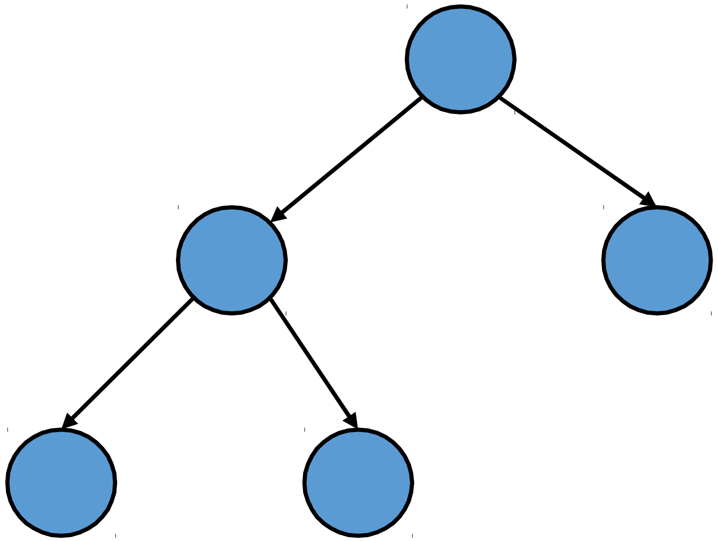
TYPES OF BINARY TREE

1. Fully Binary Tree/Strictly Binary Tree

is a binary tree in which each node has exactly zero or two children.



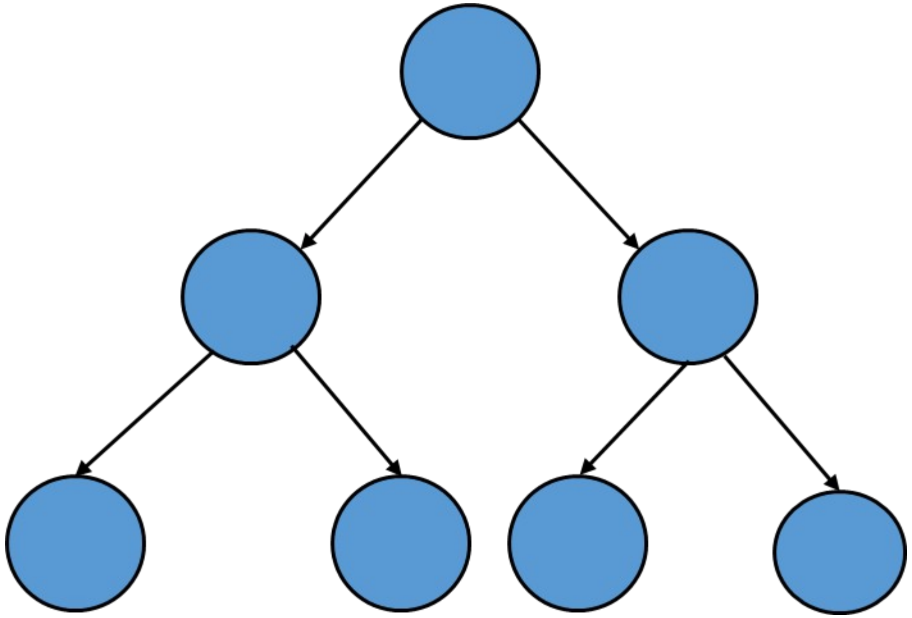
(a)



(b)

2. Complete Binary tree/Perfect Binary tree

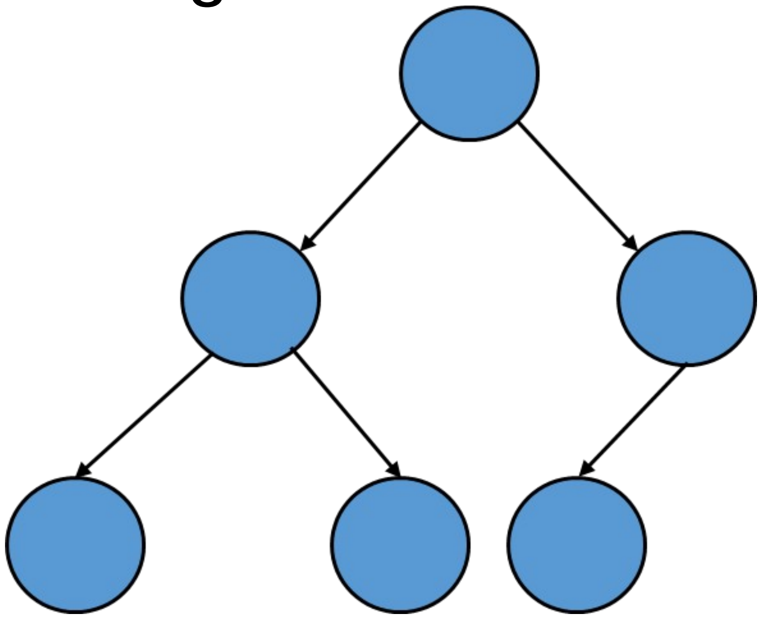
is a binary tree with the leaf nodes at the same level.
Nth level should have 2^n nodes.



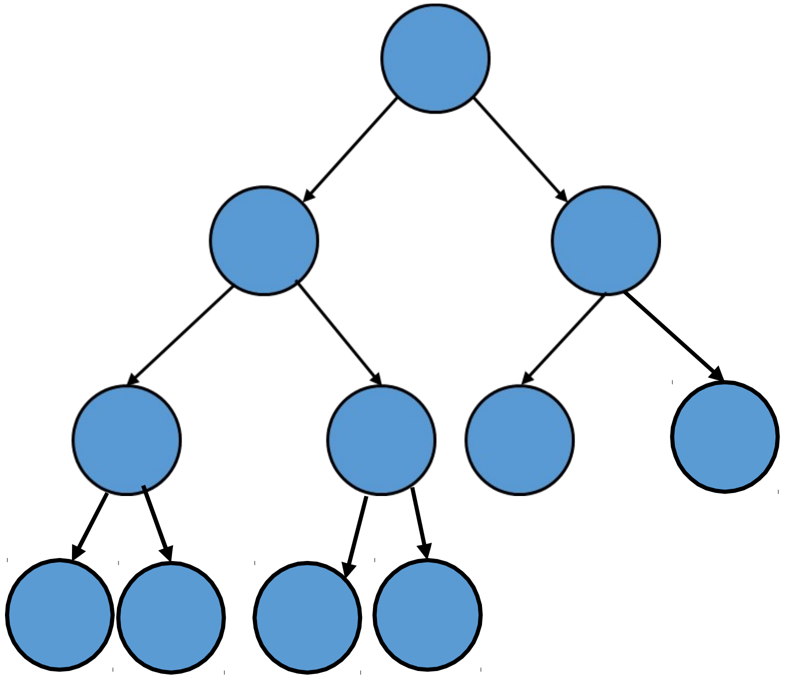
(a)

3. Almost Complete Binary Tree

- is a complete binary tree up to $(n-1)^{th}$ level and at the nth level it would have less than 2^n nodes.
- All the nodes at the n^{th} level must be compulsorily be filled in the left to right order.



(a)



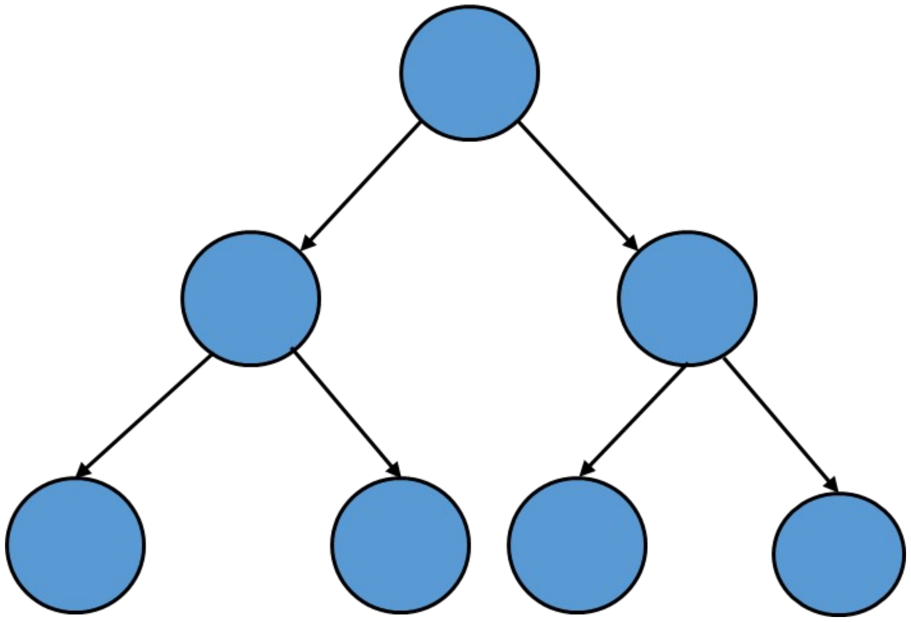
(b)

PROPERTIES OF A BINARY TREE

1. Maximum number of nodes at level 'l' of a binary tree is $2^l - 1$.

$2^0 + 2^1 + 2^2 + \ldots\ldots + 2^l$

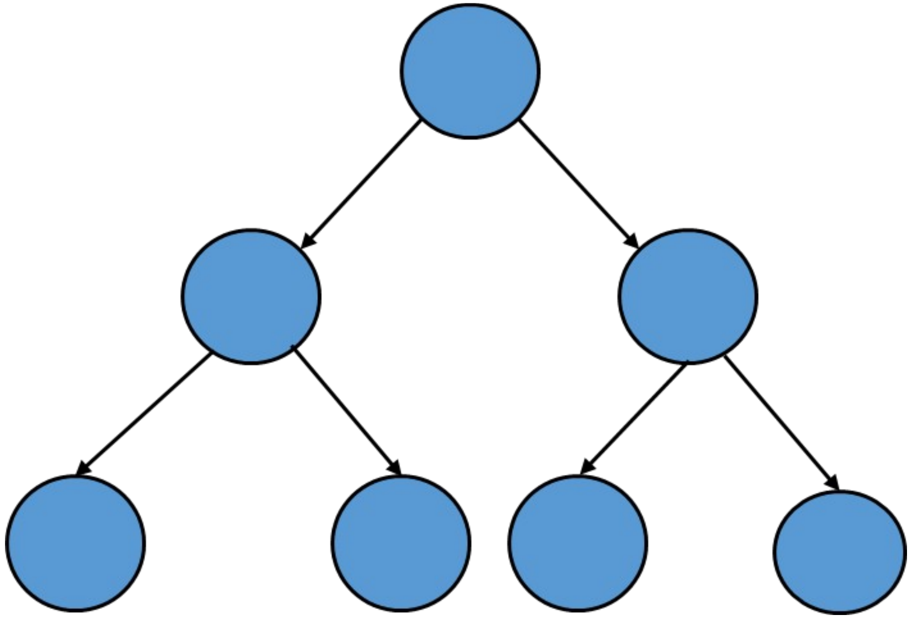
2^l



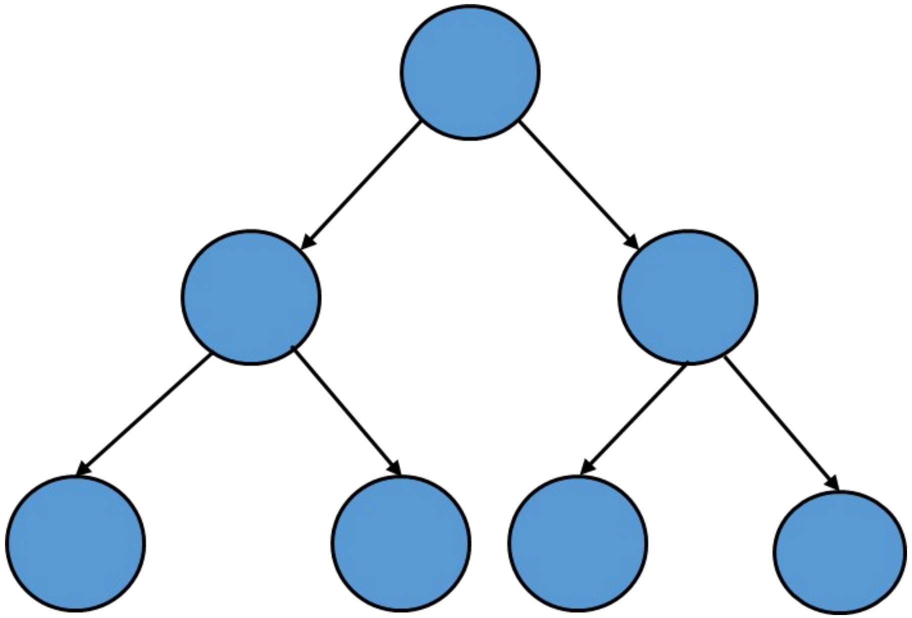
2. In a perfect binary tree, height of the tree is $\log_2(N+1) - 1$.

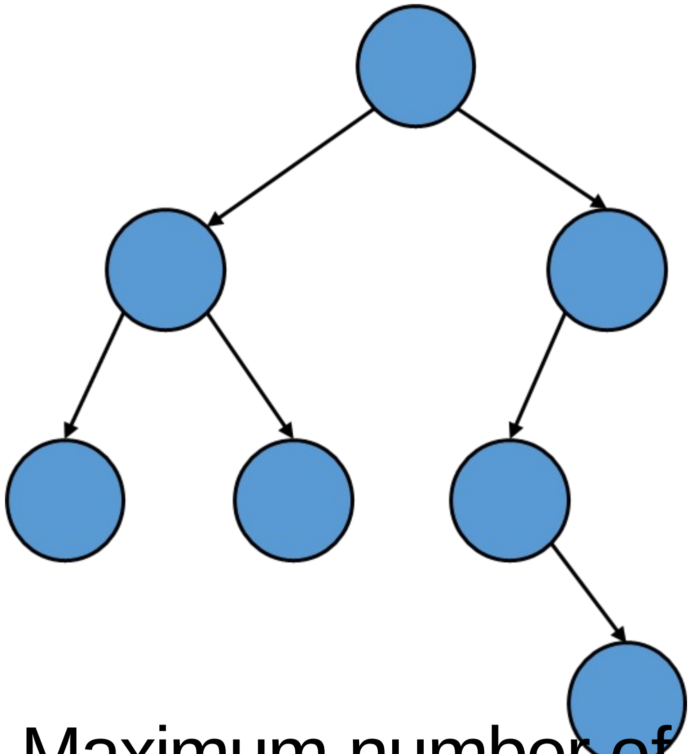
(or)

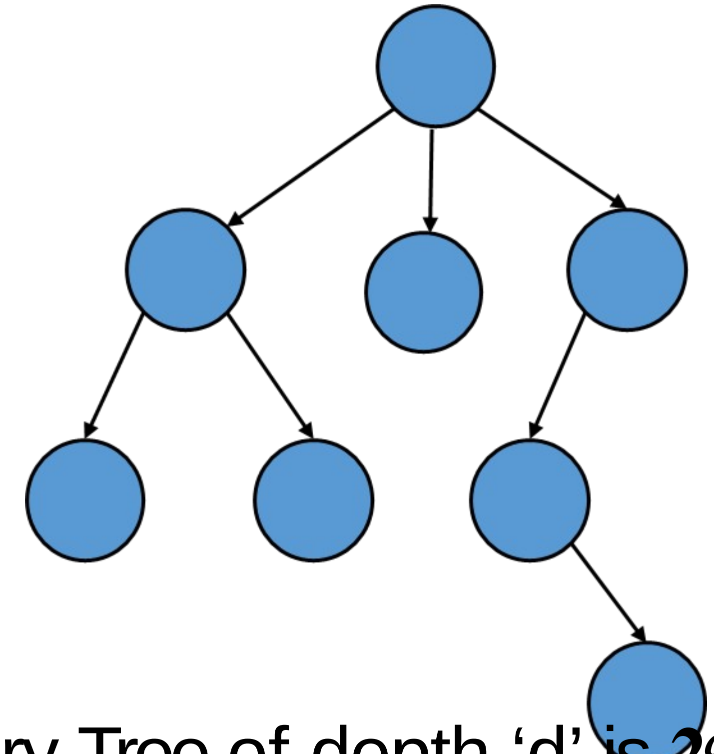
In a Binary Tree with N nodes, minimum possible height or minimum number of levels is $\log_2(N+1) - 1$.



3. A perfect binary tree with ‘L’ levels has 2^l leaves.







4.Maximum number of nodes in a Binary Tree of depth ‘d’ is $2^{d+1}-1$.

5.Binary tree with ‘l’ leaves has atleast $\log_2 l$ levels.

6.Number of leaves is one greater than the number of non leaf nodes with two children.

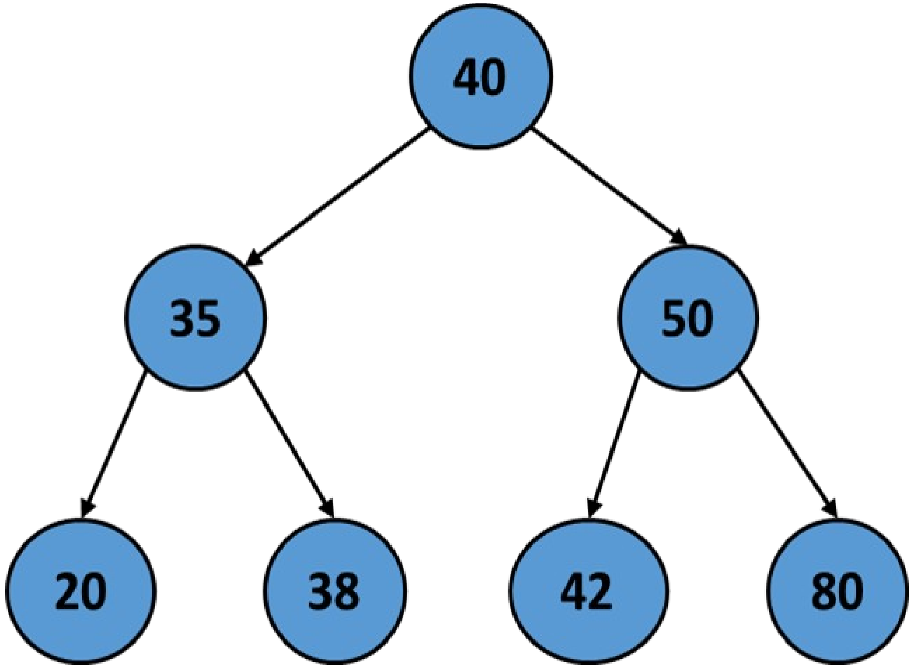
Binary Search Tree

is a binary tree data structure which has the following properties:

- The left subtree of a node contains only nodes with keys less than the node’s key.
-
- The right subtree of a node contains only nodes with keys greater than the node’s key.
-
- The left and right subtree each must also be a binary search tree.

left_child < parent < right_child

Example:



TYPES OF BINARY SEARCH TREE TRAVERSALS

1. **In-order traversal**
2. **Pre-order traversal**
3. **Post-order traversal**

ALGORITHM FOR IN-ORDER TRAVERSAL

1. Traverse the left subtree in in-order.
2. Visit the root node.
3. Traverse the right subtree in in-order.

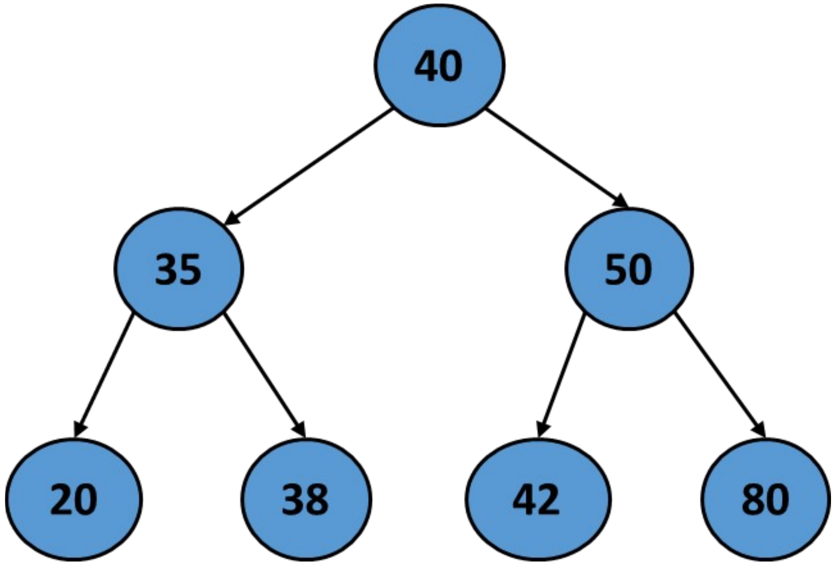
ALGORITHM FOR PRE-ORDER TRAVERSAL

1. Visit the root node.
2. Traverse the left subtree in pre-order.
3. Traverse the right subtree in pre-order.

ALGORITHM FOR POST-ORDER TRAVERSAL

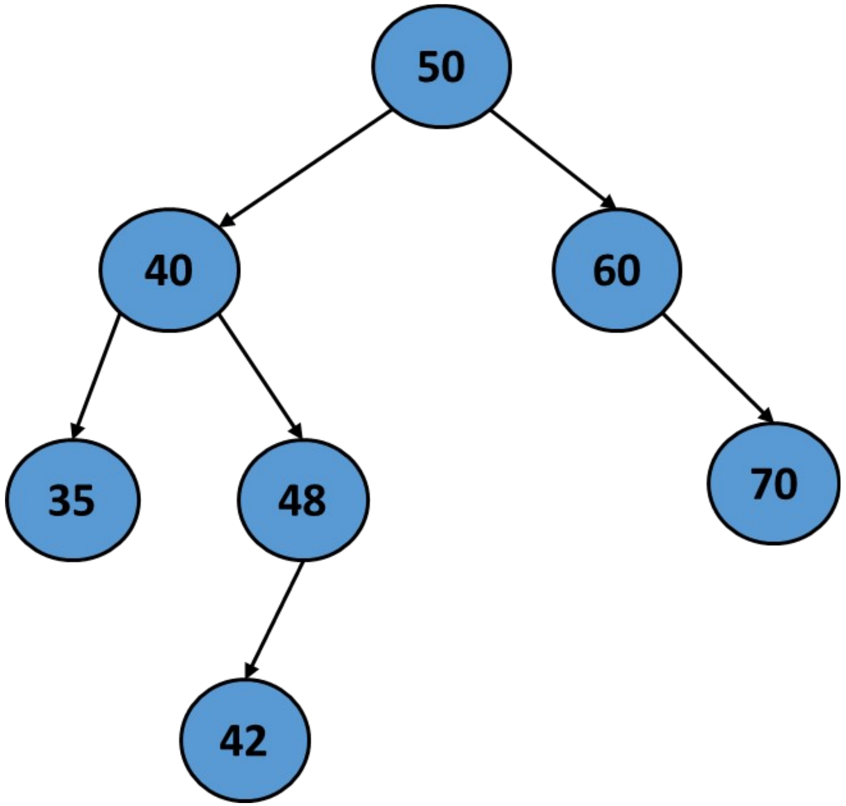
1. Traverse the left subtree in post-order.
2. Traverse the right subtree in post-order.
3. Visit the root node.

1. 40, 50, 35, 38, 80, 42, 20



1. **In-order: 20 35 38 40 42 50 80**
2. **Pre-order: 40 35 20 38 50 42 80**
3. **Post-order: 20 38 35 42 80 50 40**

2. 50, 60, 40, 48, 70, 35, 42



1. **In-order: 35 40 42 48 50 60 70**
2. **Pre-order: 50 40 35 48 42 60 70**
3. **Post-order: 35 42 48 40 70 60 50**