Non-Linear Data Structures

“Making room for real-life concepts”

Prerequisite: None

Md. Saidul Hoque Anik
onix.hoque.mist@gmail.com

Types of Data Structures in C++

Raw Data Structures	Abstract Data Types
Concrete representation of data	Conceptual model defined by its behavior from the point of view of a user

• Primitive Data Types
  • int
  • float, double
  • char
• Array
• Struct
• Class
	• Vector, Deque
	• Set, Map
	• Stack, Queue, Priority Queue
	• Single Linked List
	• Doubly Linked List
	• Tree
	• Graph

Types of Data Structures in C++

Raw Data Structures	Abstract Data Types
Concrete representation of data	Conceptual model defined by its behavior from the point of view of a user
 	  

Types of Data Structures in C++

Raw Data Structures	Abstract Data Types
Concrete representation of data	Conceptual model defined by its behavior from the point of view of a user

• Primitive Data Types
    • int
    • float, double
    • char
• Array
• Struct
• Class
	• Vector, Deque
	• Set, Map
	• Stack, Queue, Priority Queue
	• Single Linked List
	• Doubly Linked List
	• Tree
	• Graph

Classification of ADT

Abstract Data Types

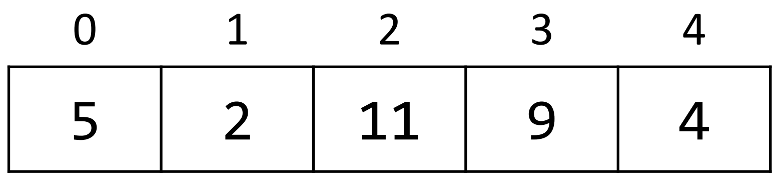
Linear

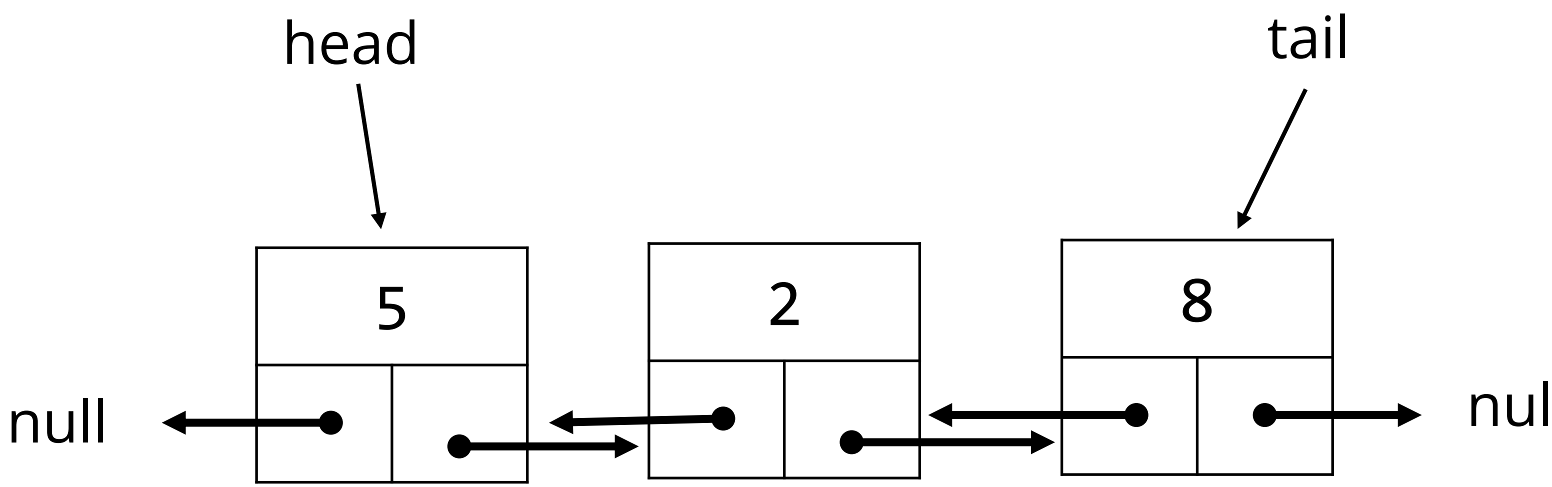
- Unique first element
- Unique last element
- Specific one next element
- Specific one previous element

Non-Linear

- First element may or may not be fixed
- No specific last element
- No specific next element
- Previous element may or may not be fixed

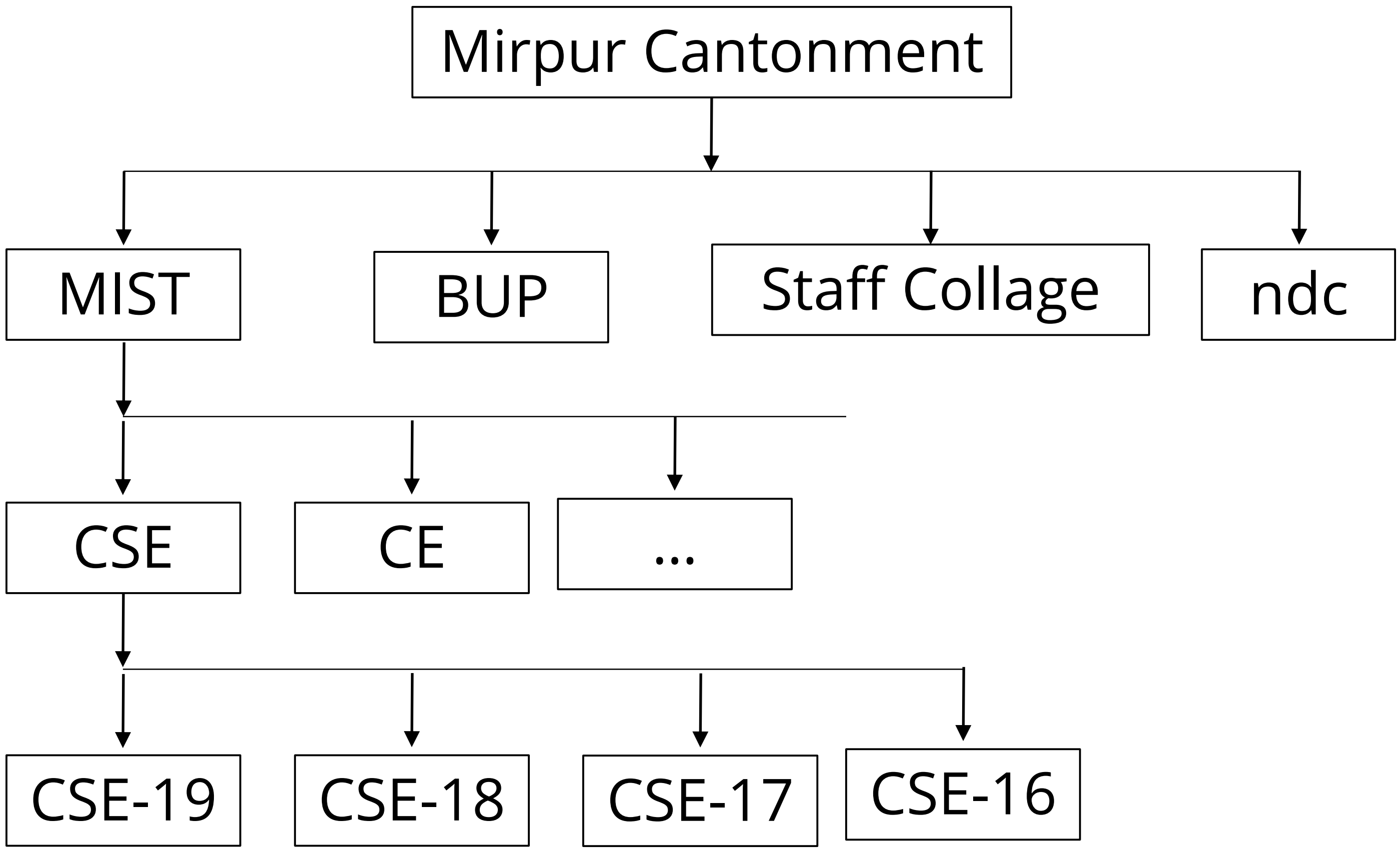
Linear Data Structures



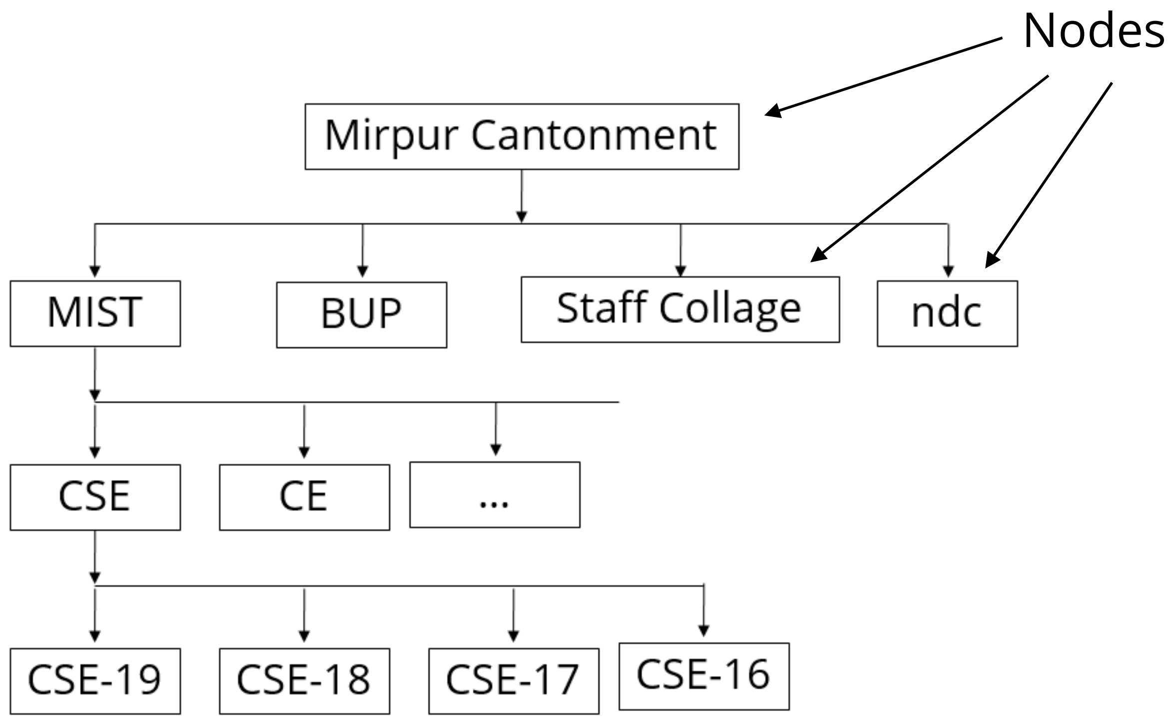


Limitation of Linear Data Structures

The following information cannot be adjusted into a linear data structure

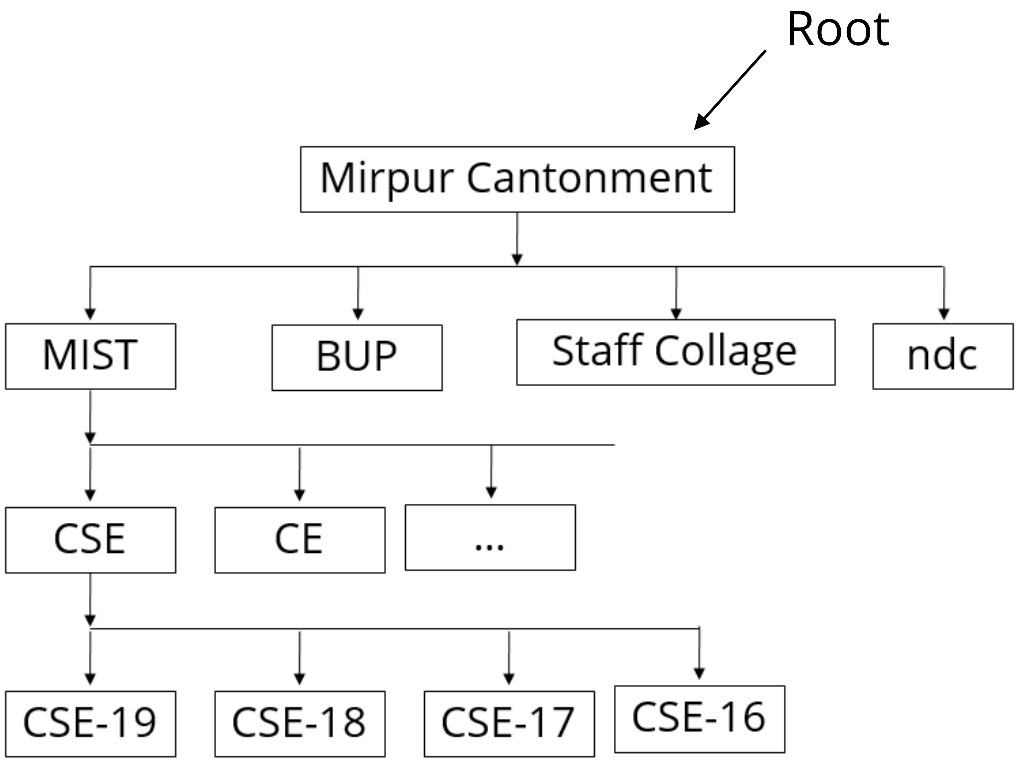


Tree



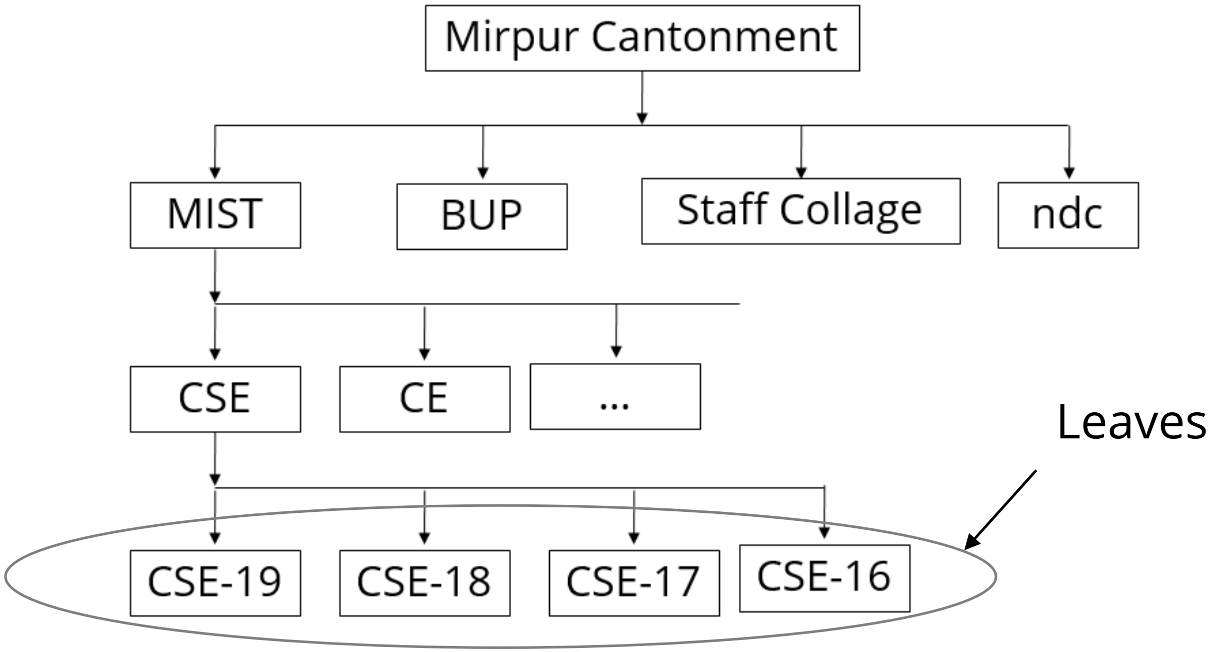
- First element may or may not be fixed
- No specific last element
- No specific next element
- Previous element may or may not be fixed

Tree



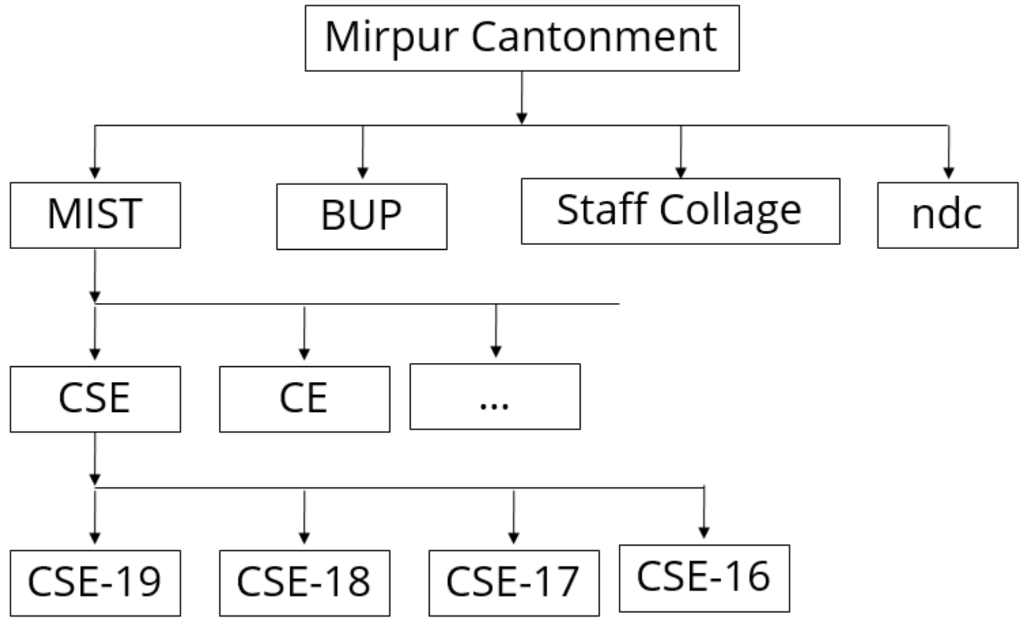
- First element fixed
- No specific last element
- No specific next element
- Previous element may or may not be fixed

Tree



- First element fixed
- No specific last element
- No specific next element
- Previous element may or may not be fixed

Tree



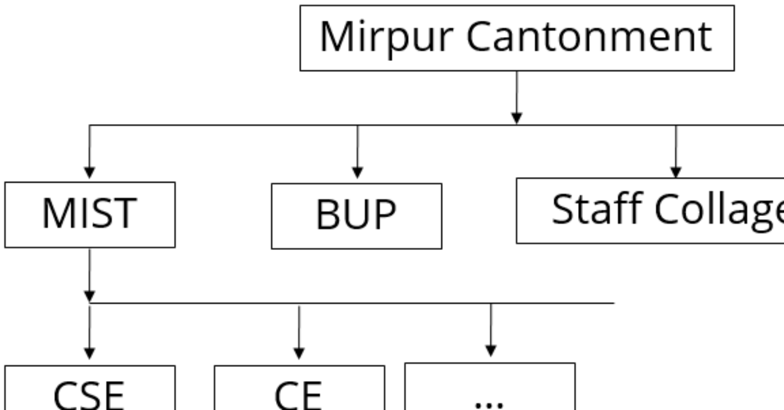
- First element fixed
- No specific last element
- No specific next element
- Previous element is unique, called parent

Tree

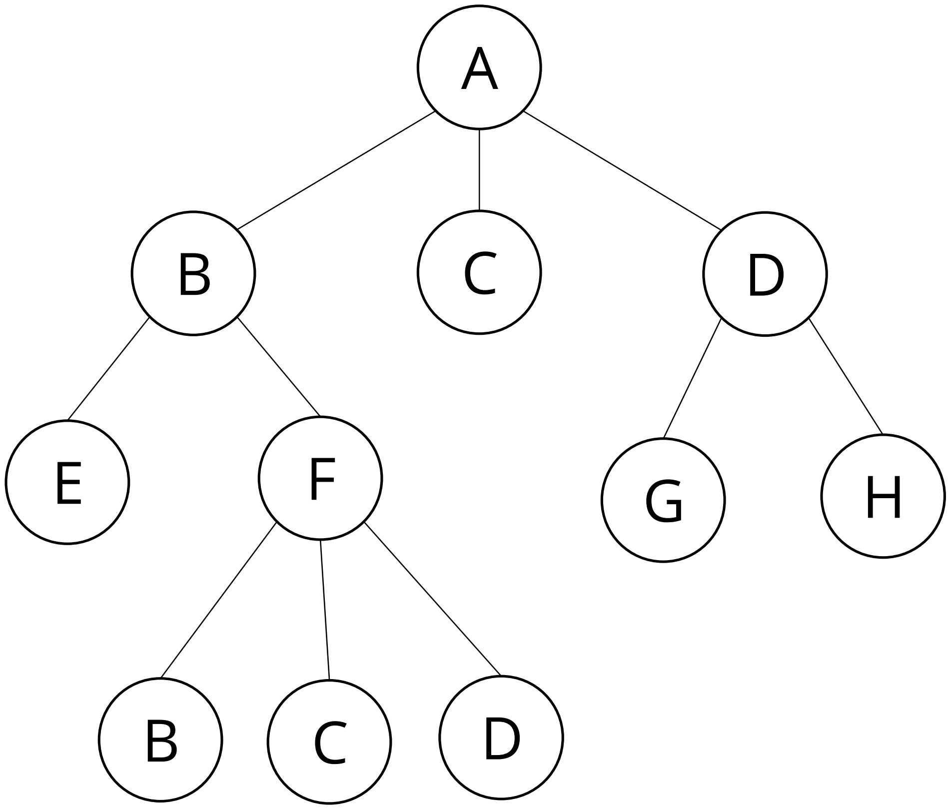
Formal Definition of a Tree T

T is a set of nodes with parent-child relationship where-

- T has a special node r (called root), with no parent node
- Each node v of T except r has a unique parent node u.
- v is called the child of u.

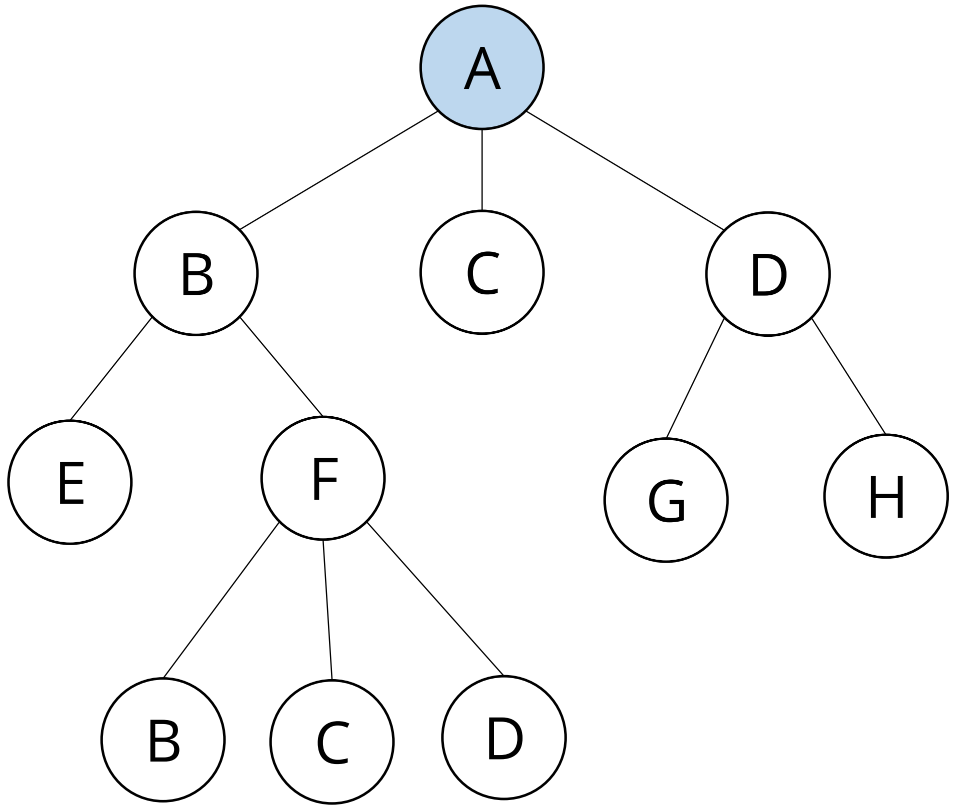


Tree Terminology



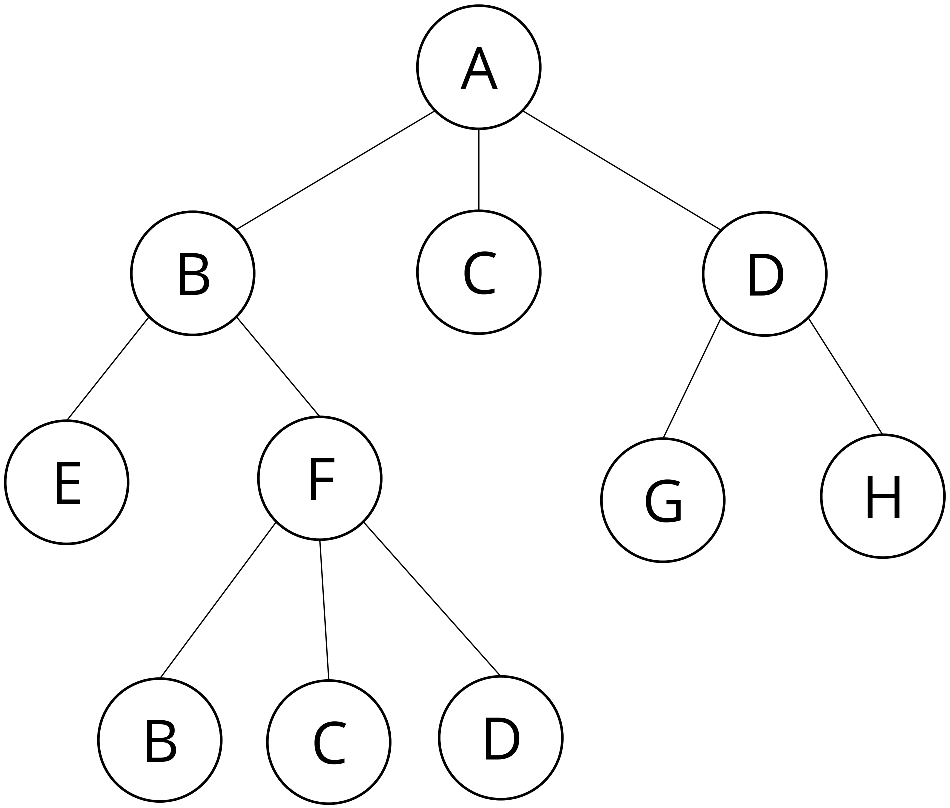
Root

Node without parent



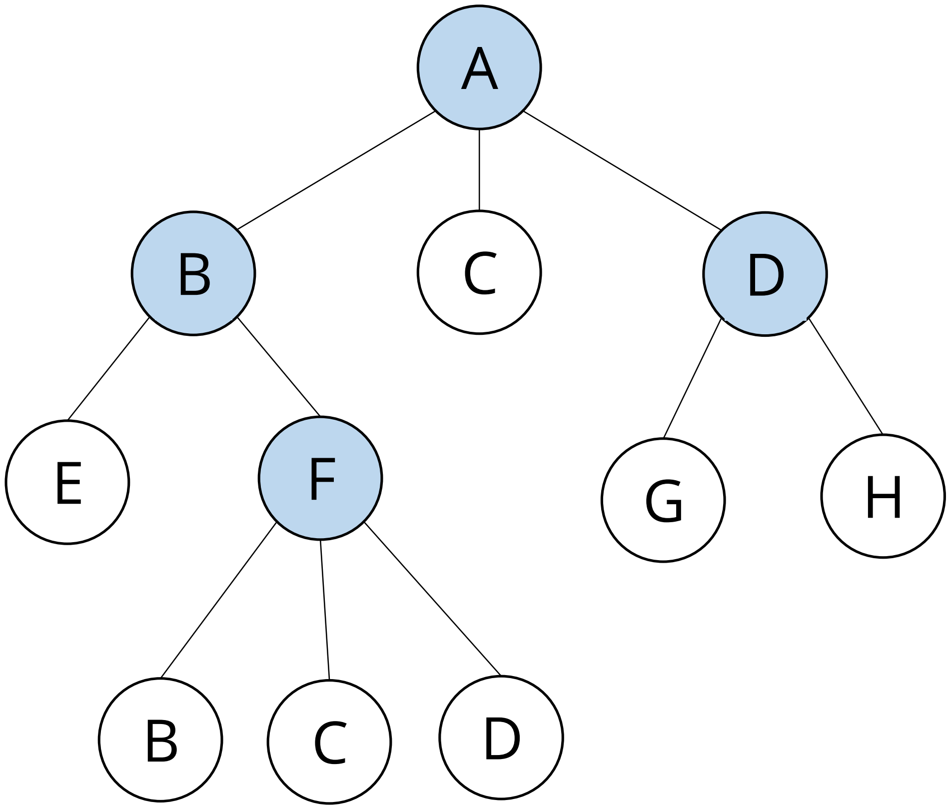
Internal Node

Node with at least one child



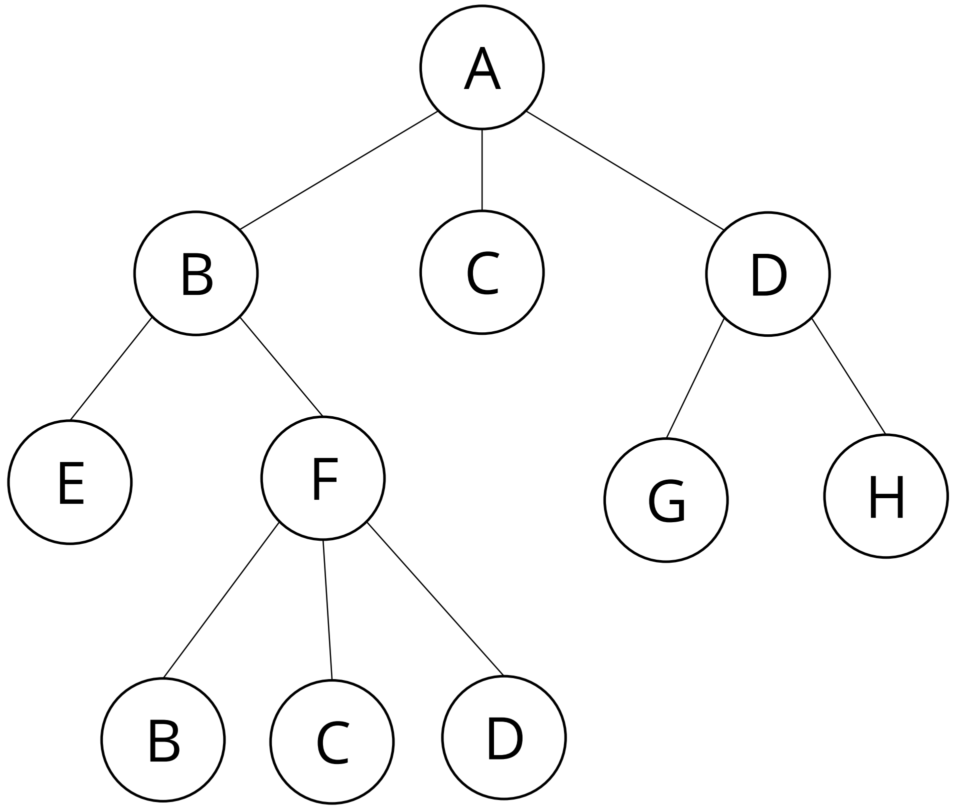
Internal Node

Node with at least one child



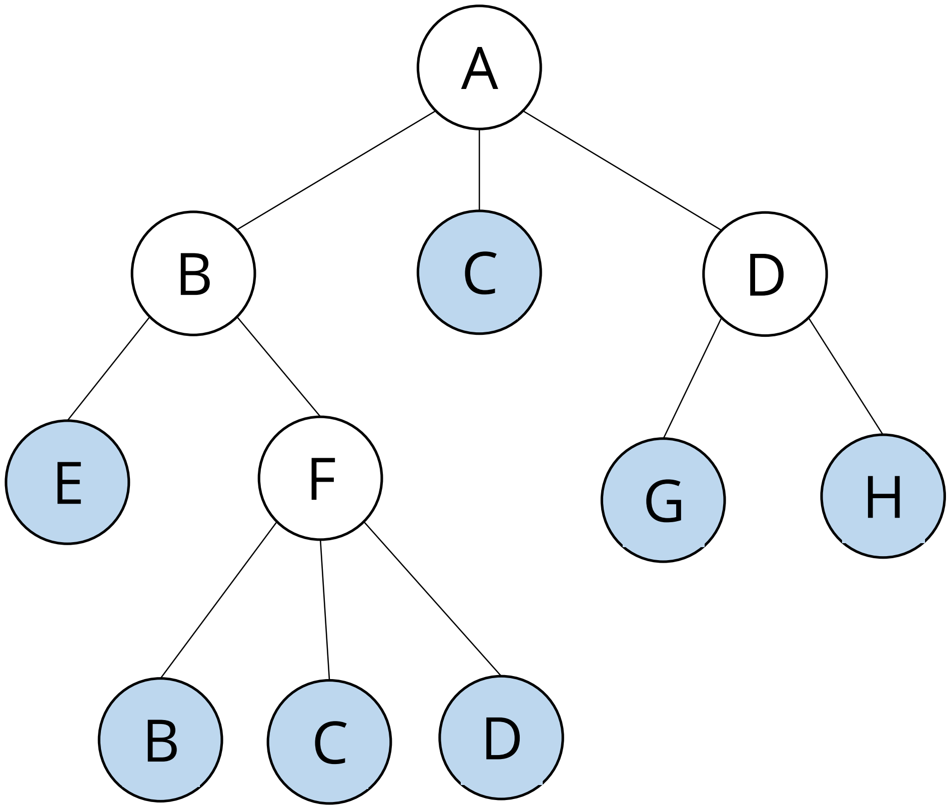
External Node (Leaf)

Node without children



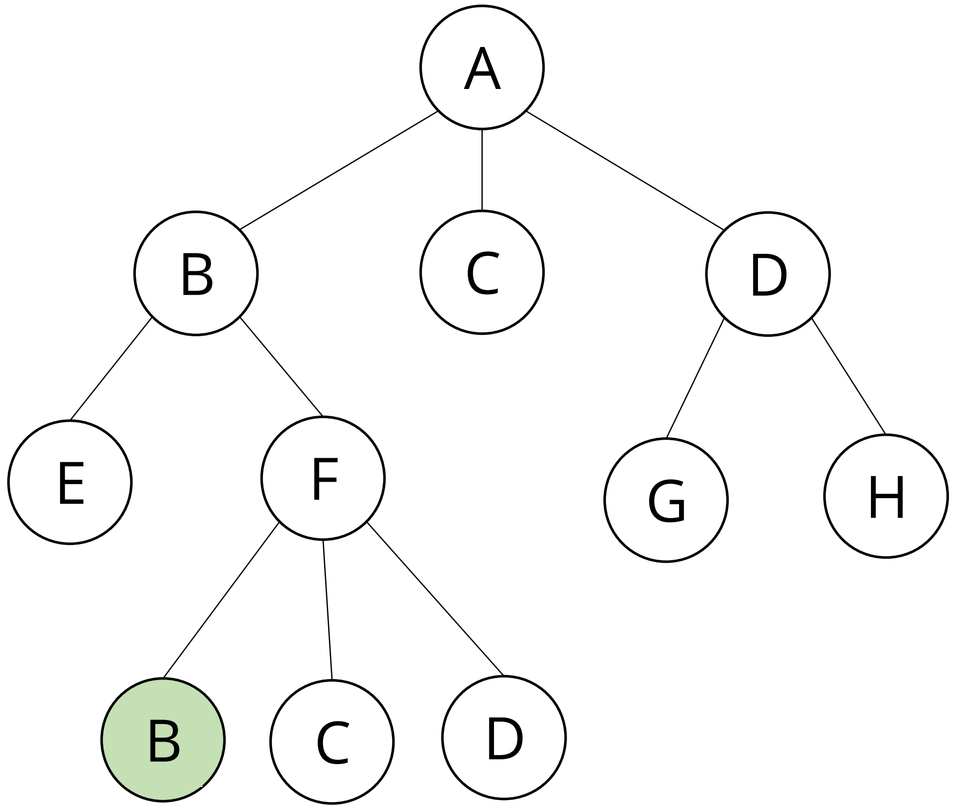
External Node (Leaf)

Node without children



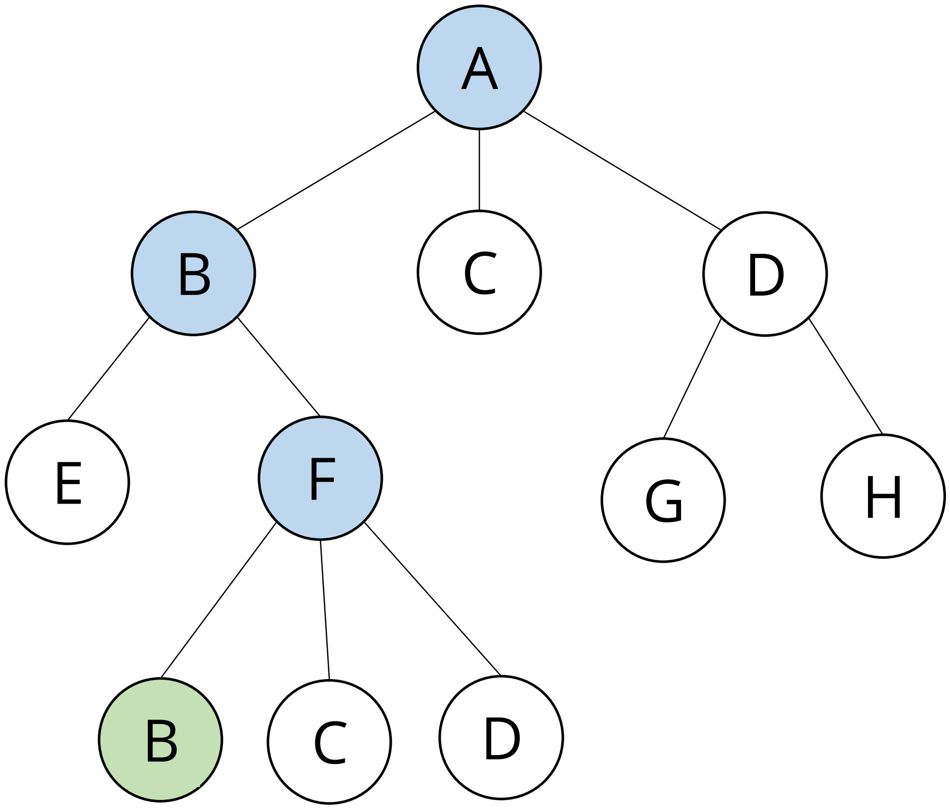
Ancestor of a node

Parent, grand-parent, grand-grandparent etc



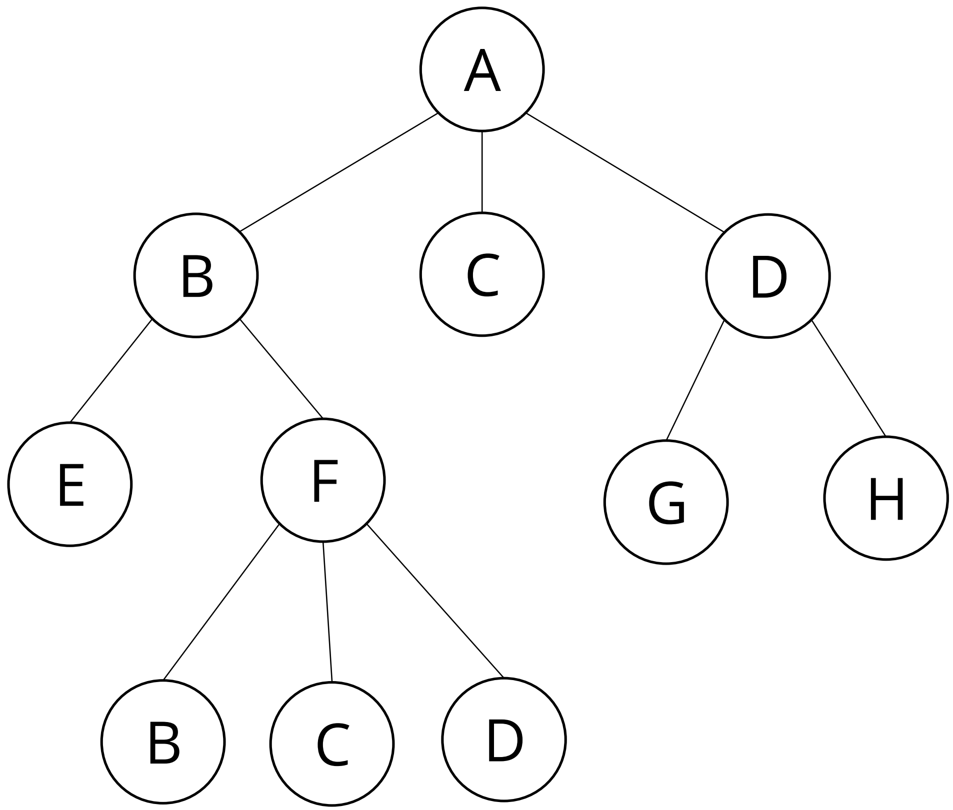
Ancestor of a node

Parent, grand-parent, grand-grandparent etc



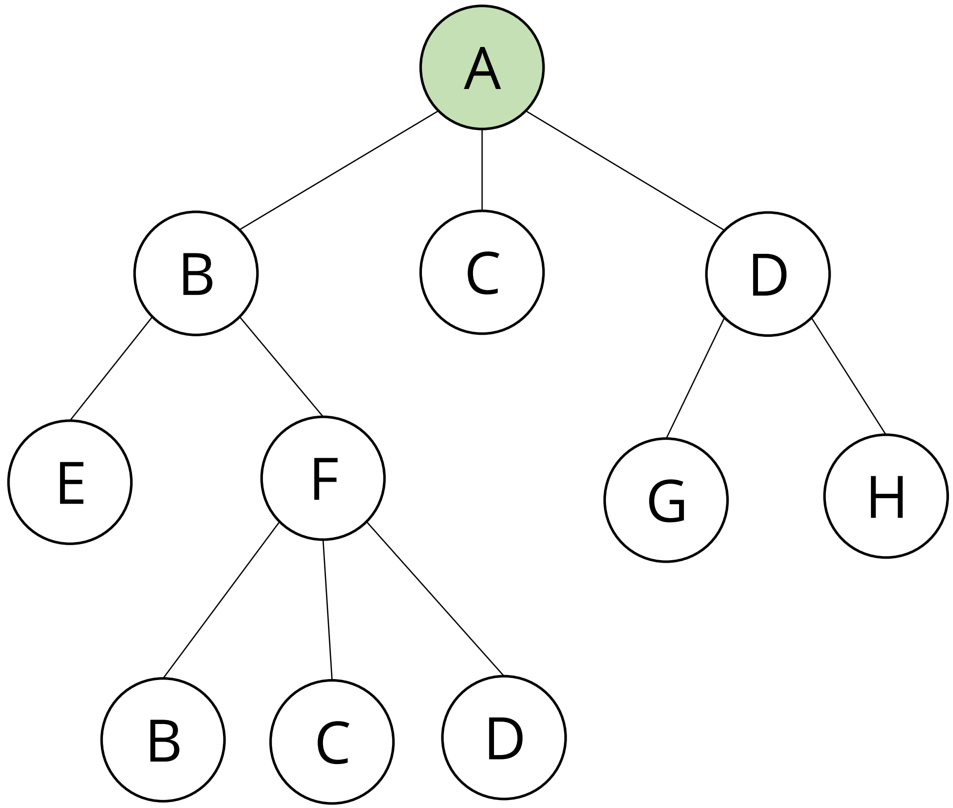
Descendant of a Node

Child, grandchild, grand-grandchild etc



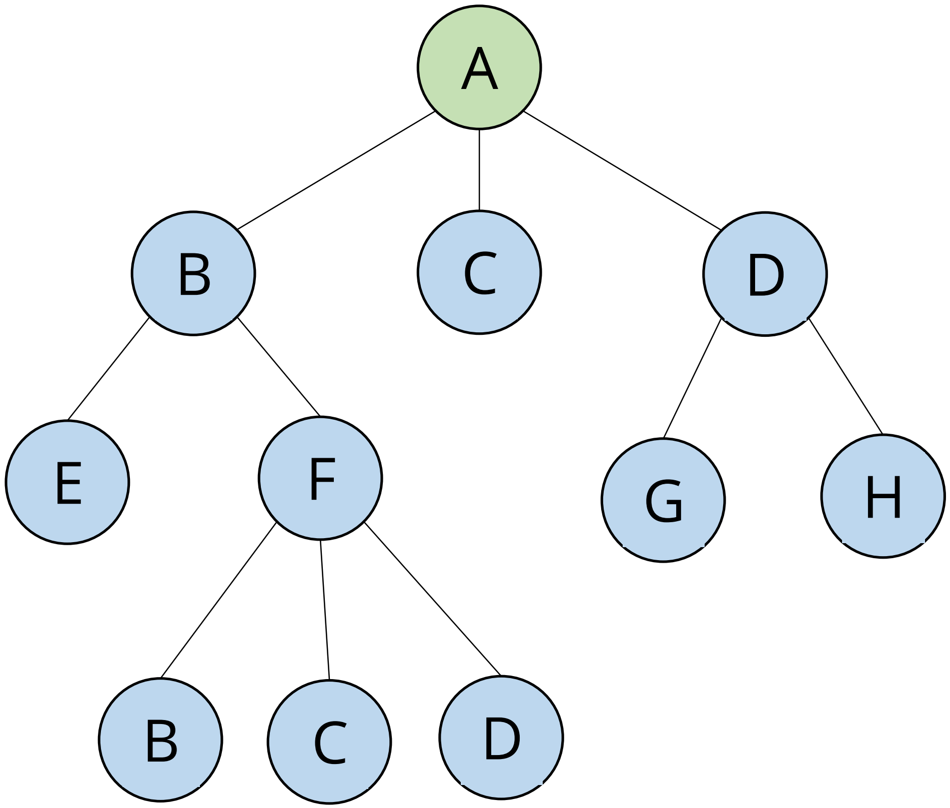
Descendant of a Node

Child, grandchild, grand-grandchild etc



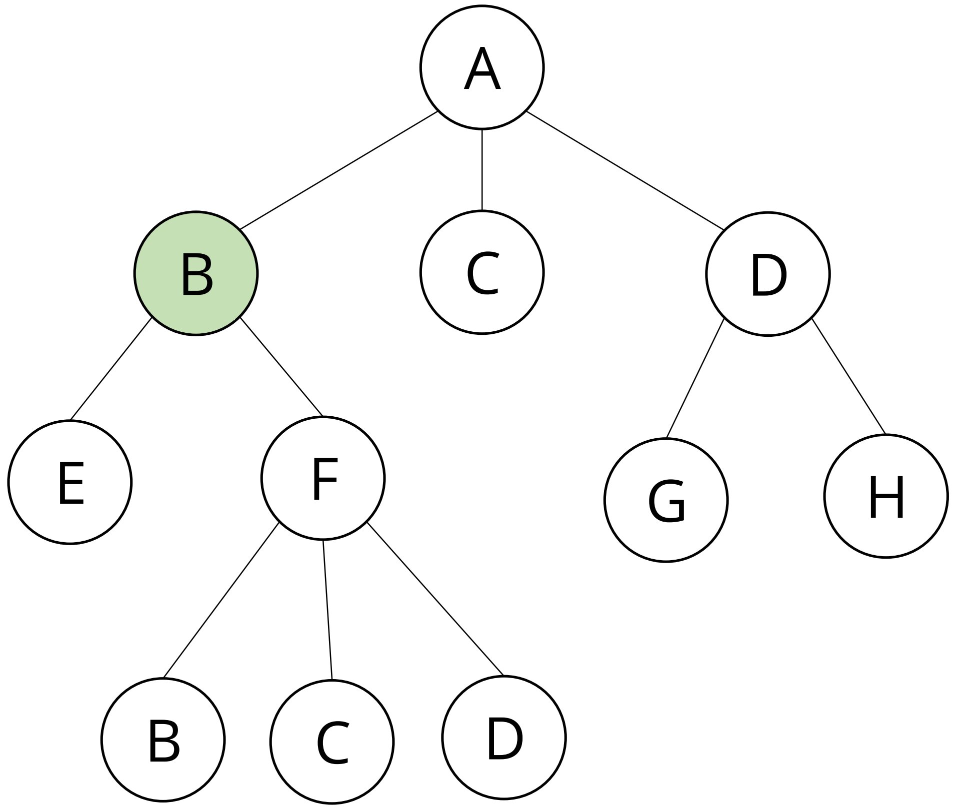
Descendant of a Node

Child, grandchild, grand-grandchild etc



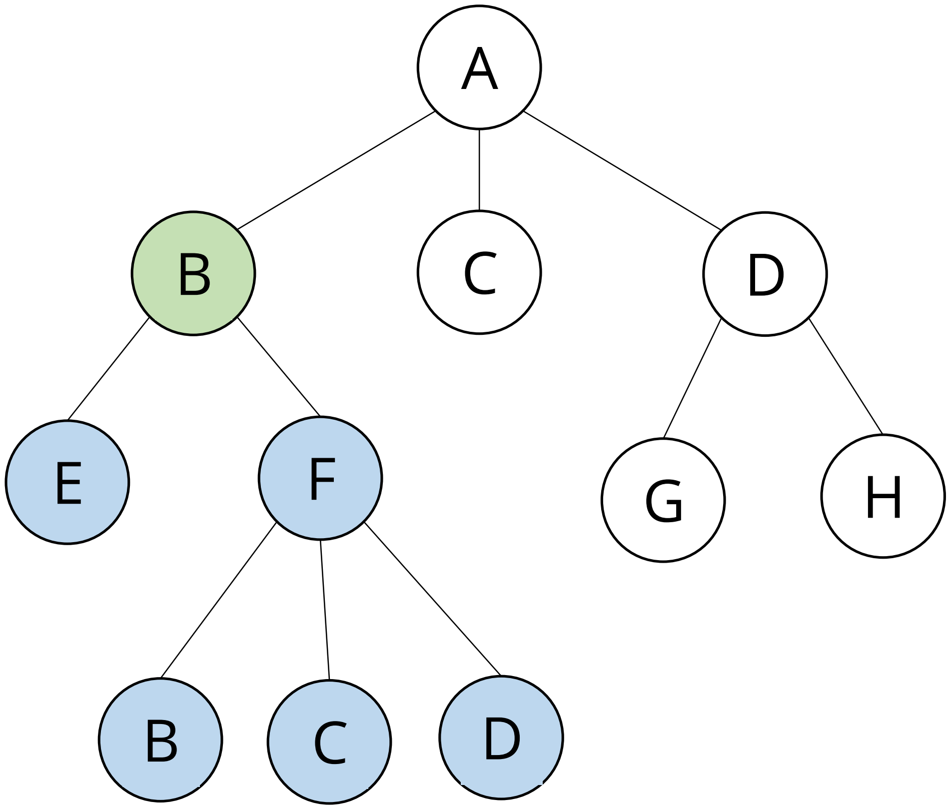
Descendant of a Node

Child, grandchild, grand-grandchild etc



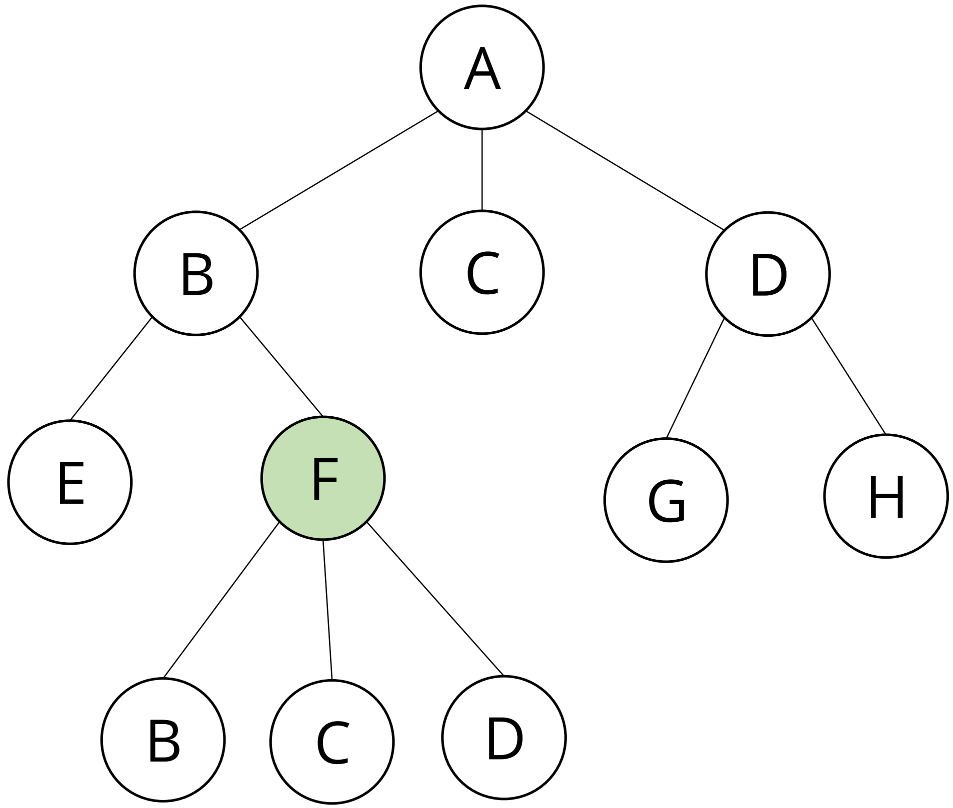
Descendant of a Node

Child, grandchild, grand-grandchild etc



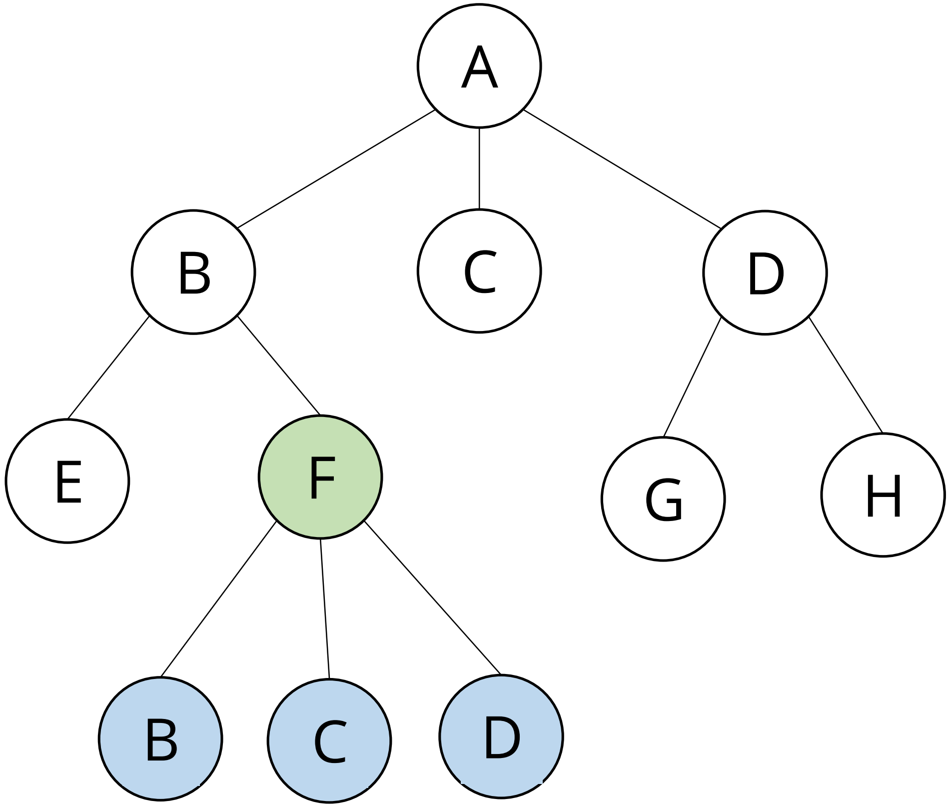
Descendant of a Node

Child, grandchild, grand-grandchild etc



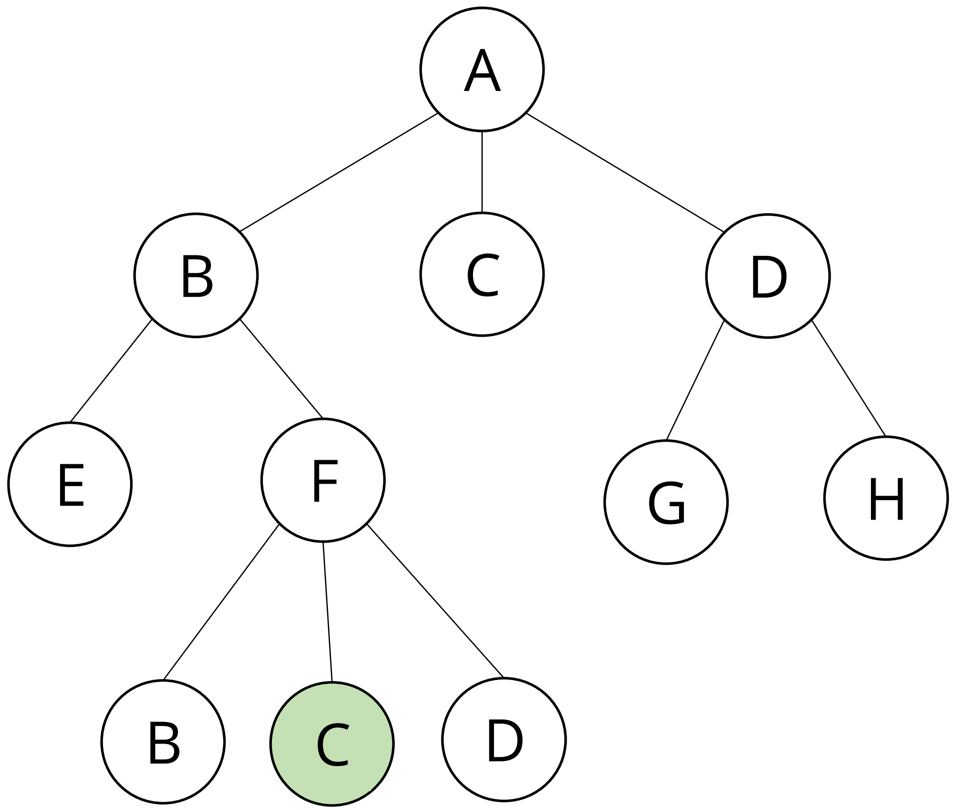
Descendant of a Node

Child, grandchild, grand-grandchild etc



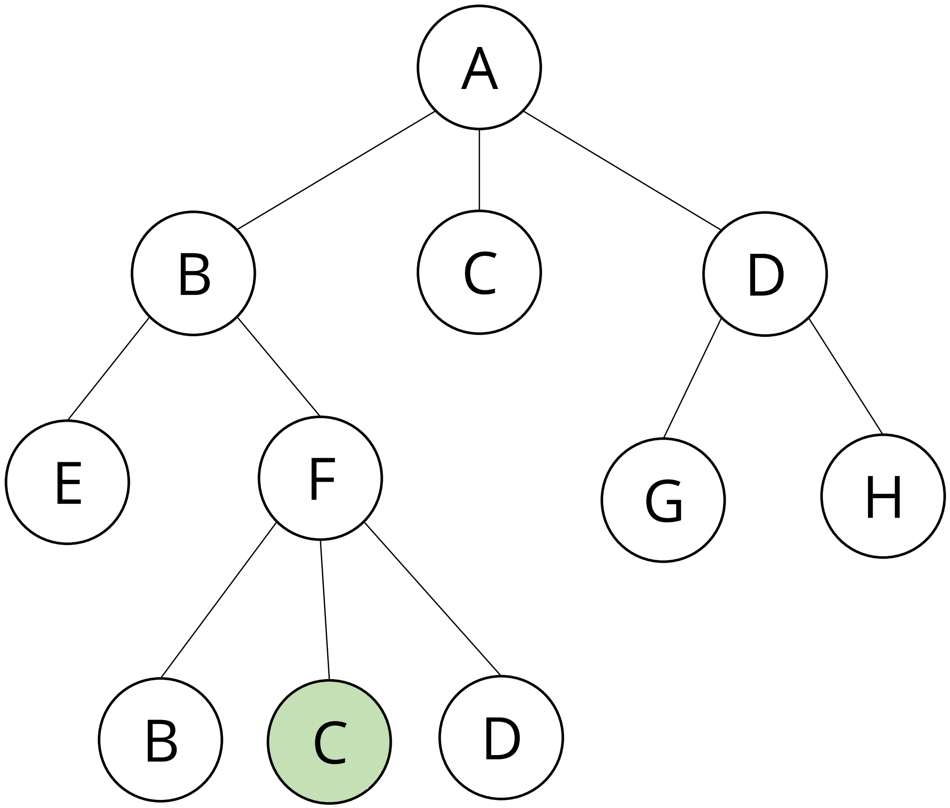
Depth of a node

Number of ancestors excluding the node itself



Depth of a node

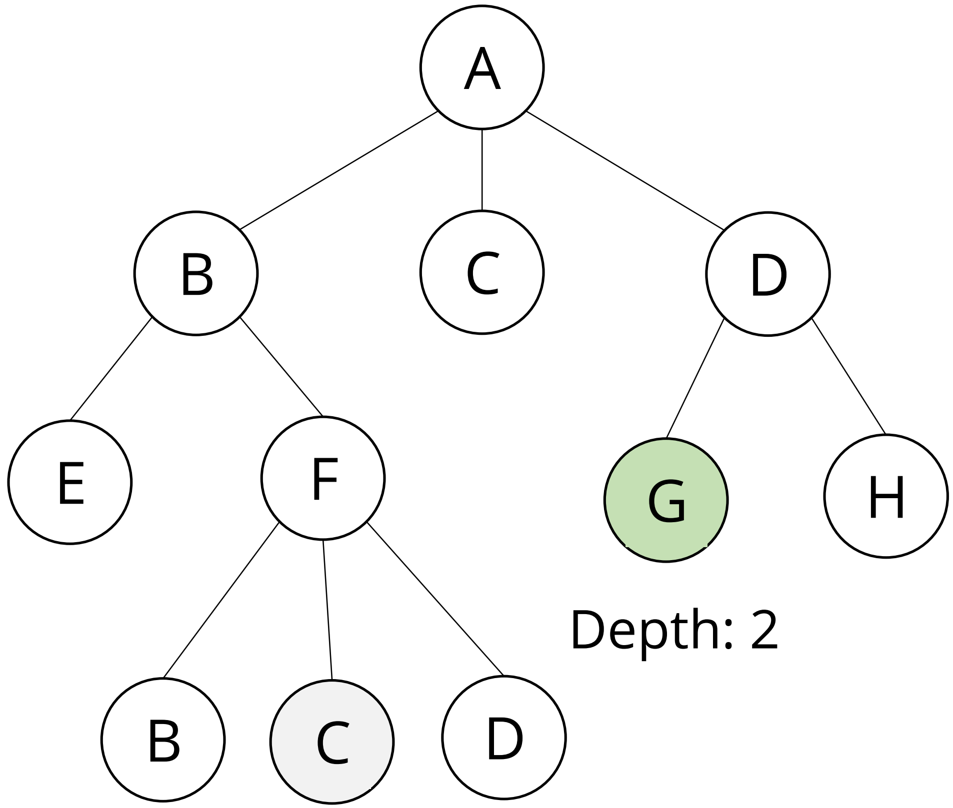
Number of ancestors excluding the node itself



Depth: 3

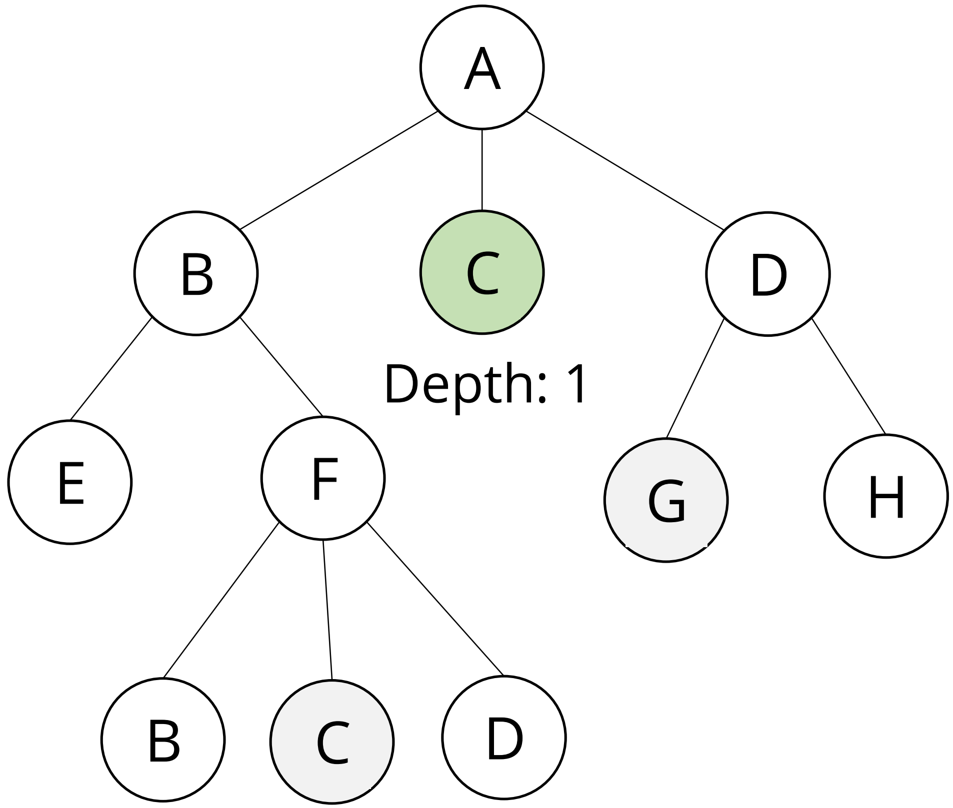
Depth of a node

Number of ancestors excluding the node itself



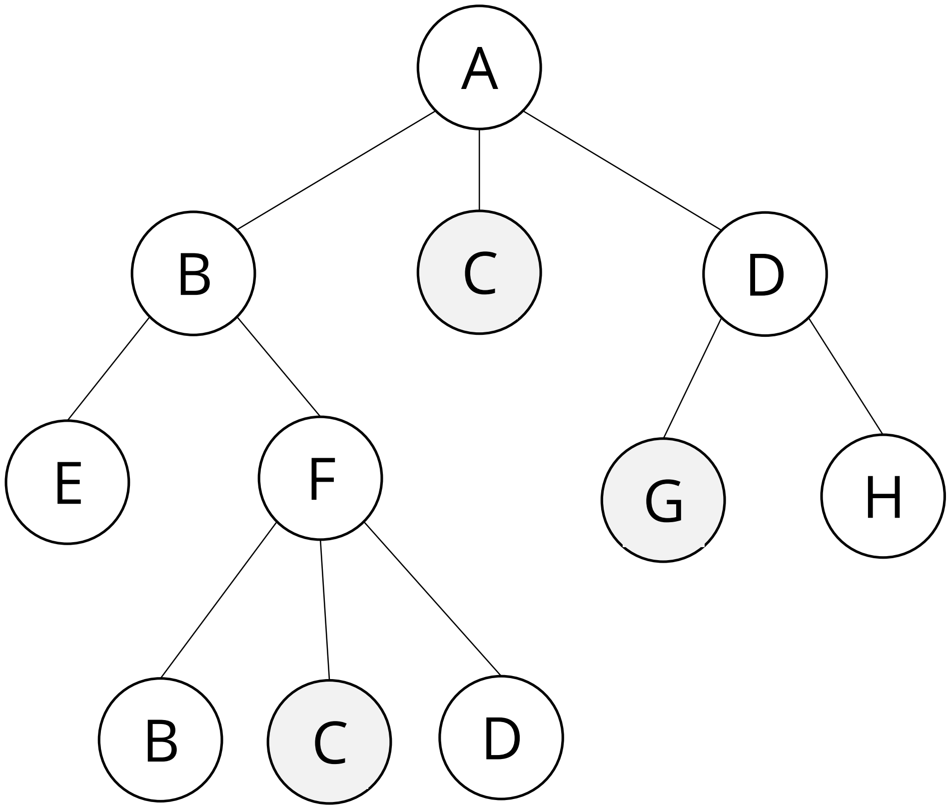
Depth of a node

Number of ancestors excluding the node itself



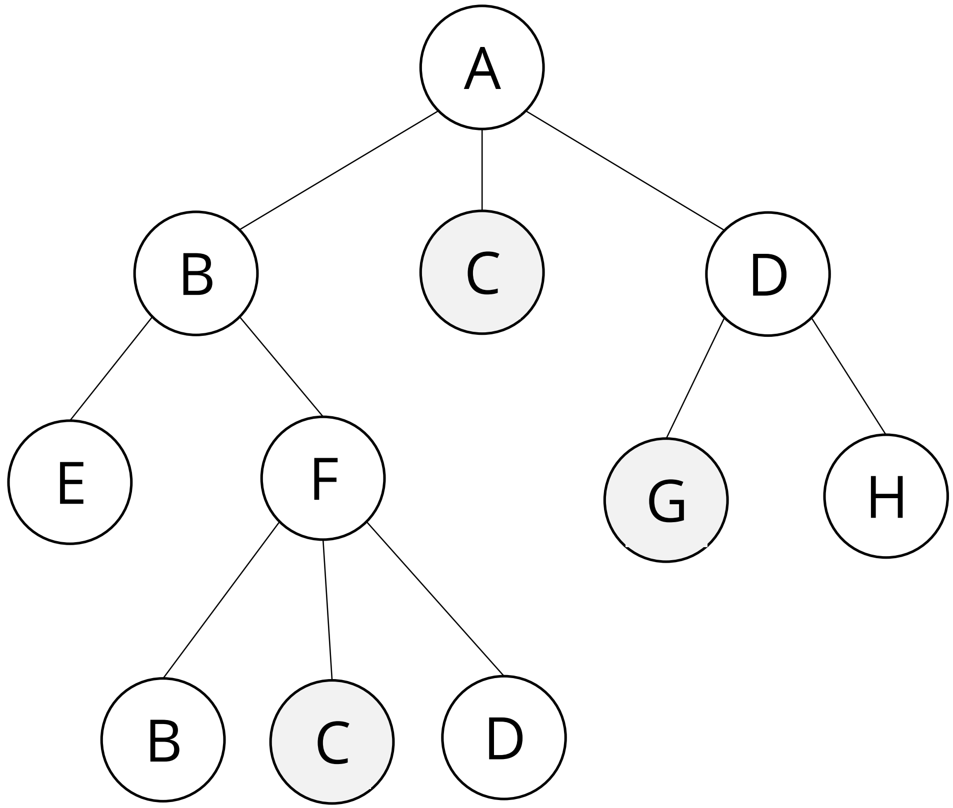
Height of a Tree

Max depth among all the nodes



Height of a Tree

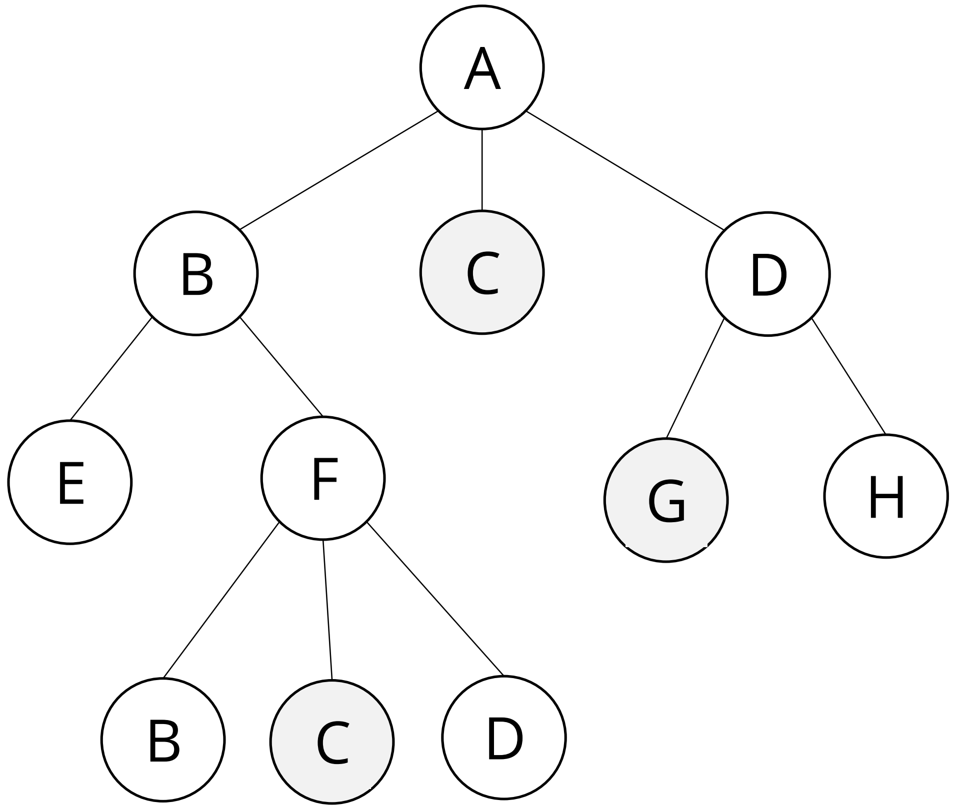
Max depth among all the nodes



Height : 3

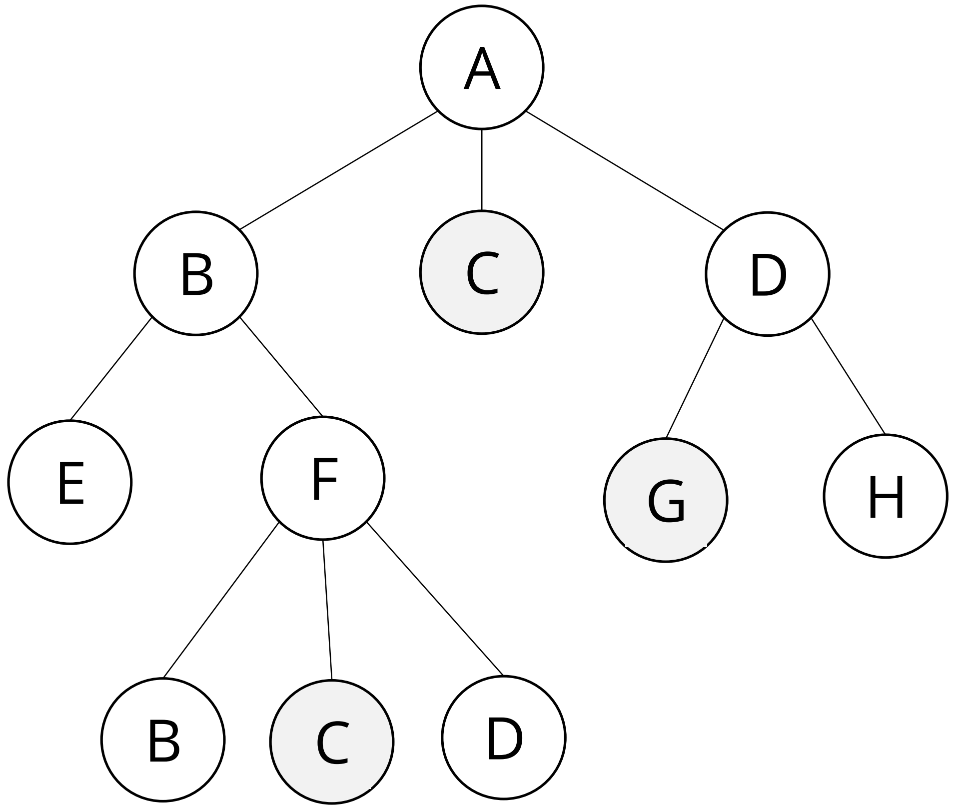
Siblings

Two nodes are siblings if their parents are same



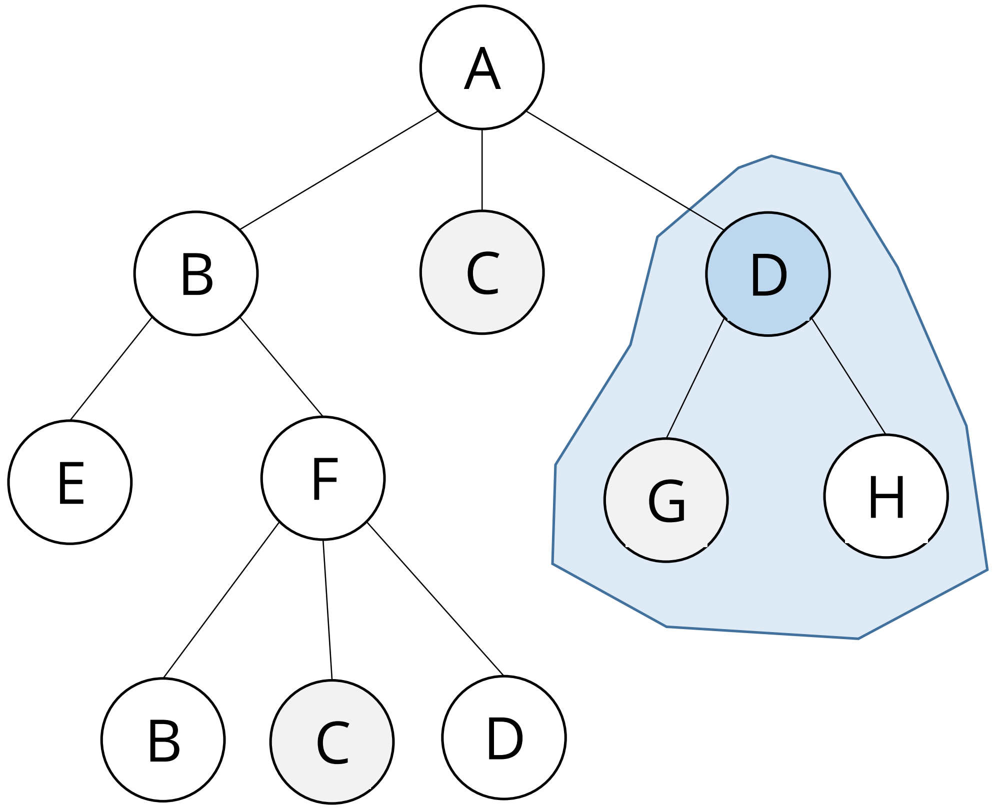
Subtree

Tree consisting of a node and its descendants



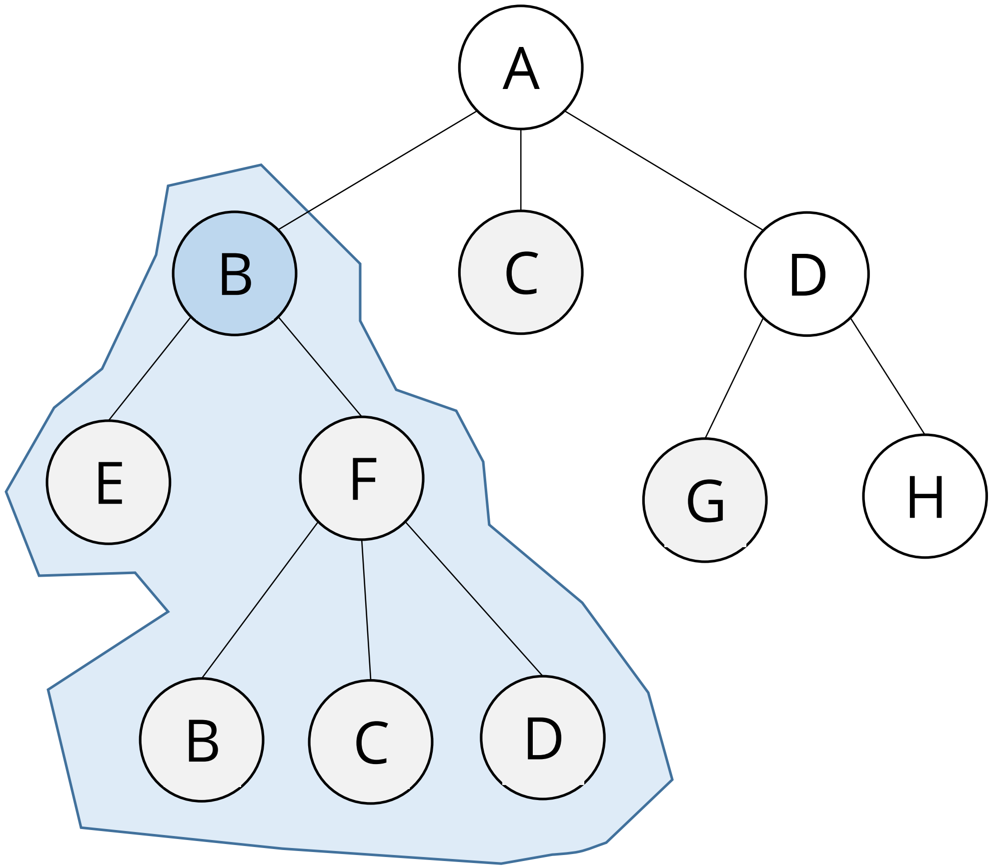
Subtree

Tree consisting of a node and its descendants



Subtree

Tree consisting of a node and its descendants

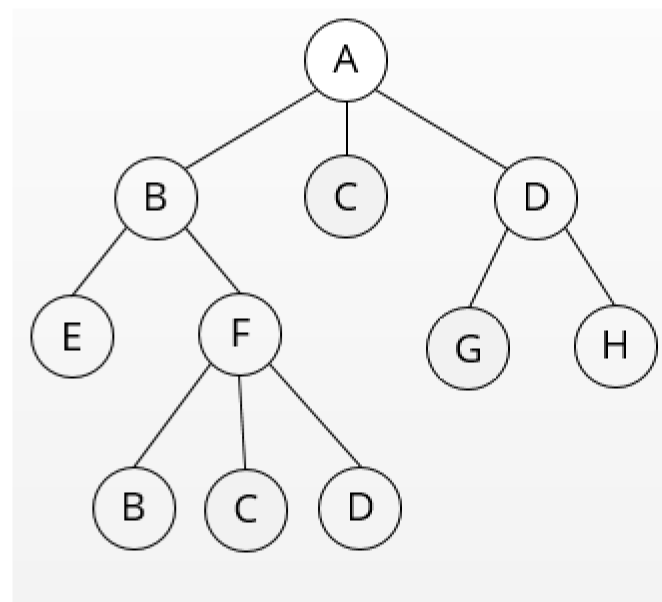


Depth calculation

Can be calculated recursively

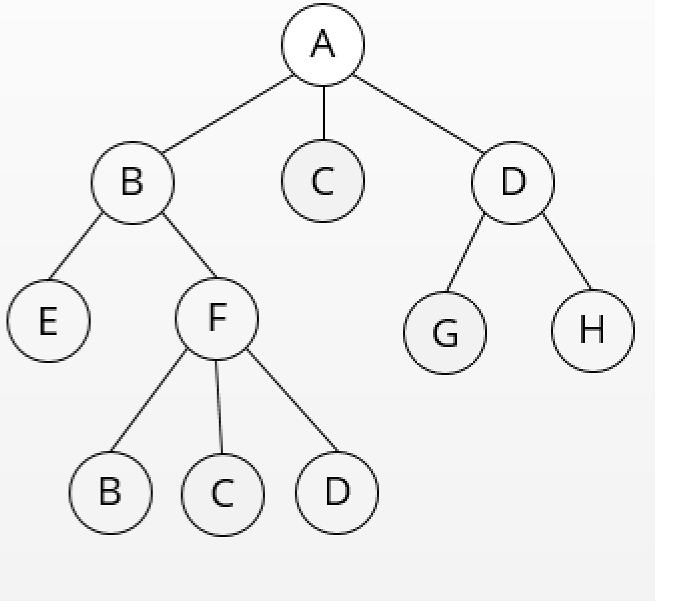
- If v is the root, then the depth is 0
- Otherwise, depth of v is 1 + depth(parent of v)

```
depth(v)
    if (isRoot(v)) then
            return 0
    else
            return 1 + depth(parent(v))
```



Height calculation

Can also be calculated recursively. How?

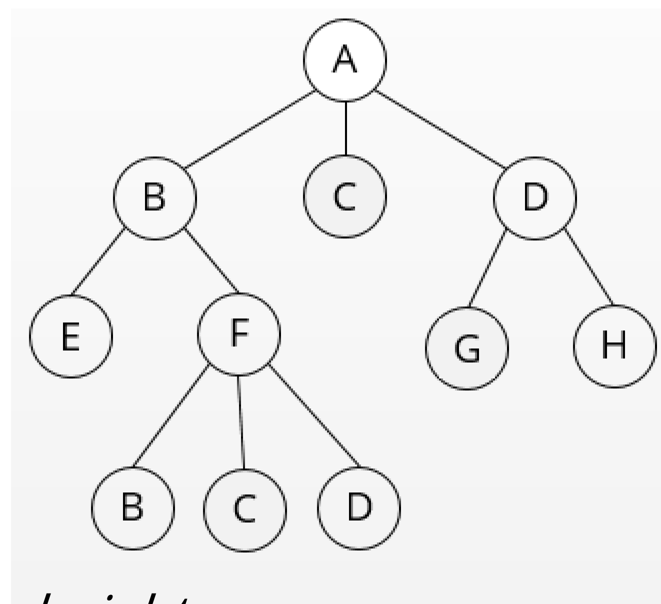


Height calculation

Can also be calculated recursively.

- If v is the leaf, then the height is 0
- Otherwise, depth of v is 1 + max(heights of children of v)

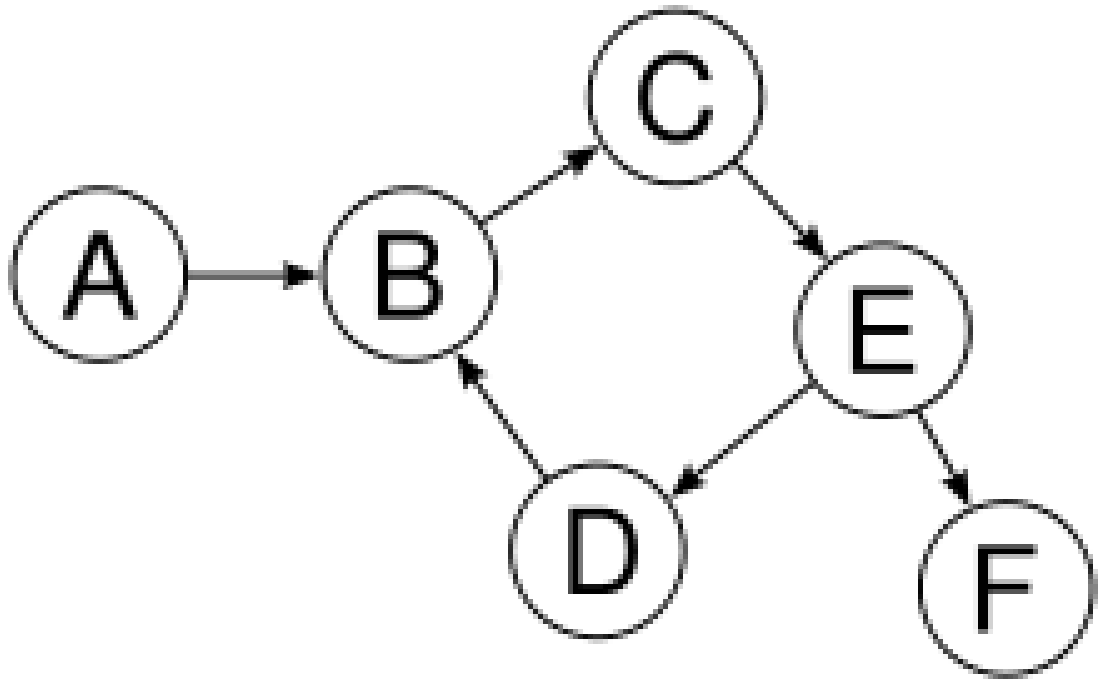
```
int height(int v)
    If (v is leaf node) return 0
    h = 0
    for each child of v as w:
        h = max(h, height(w))
    return 1 + h
```



Note: Max depth among all leaves is the height

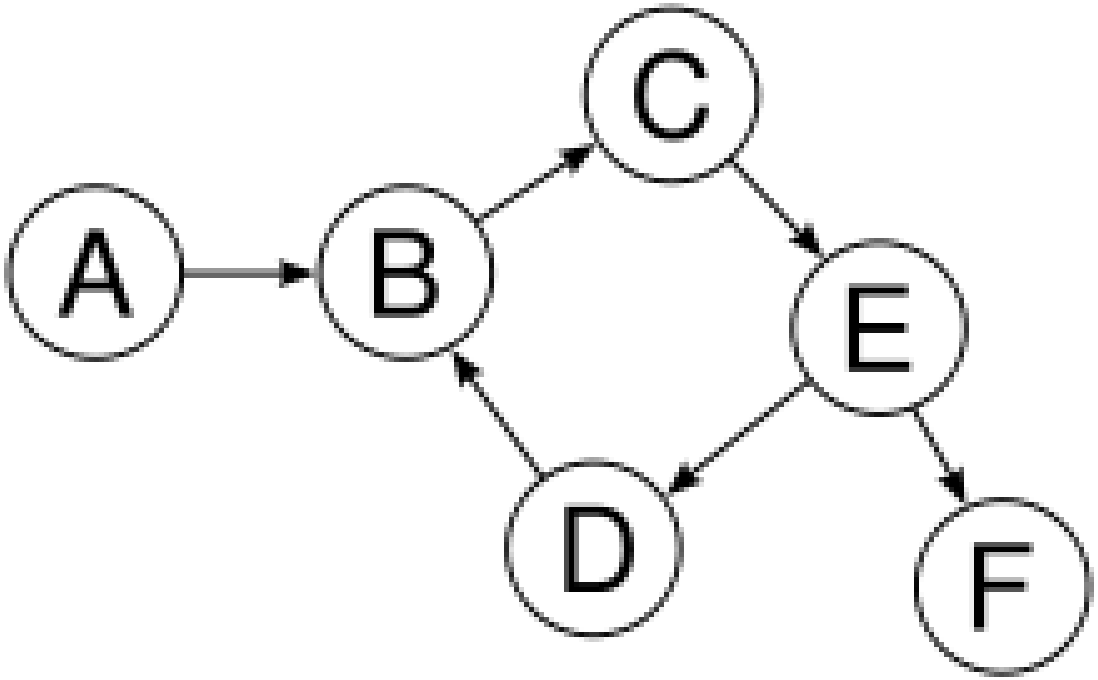
Exercise

Is it a tree?



Exercise

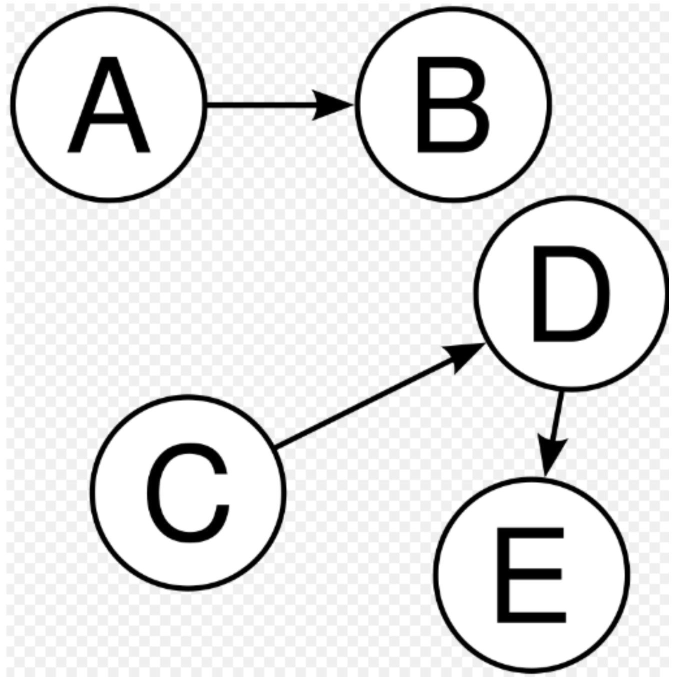
Is it a tree?



 No, because B has more than one parent

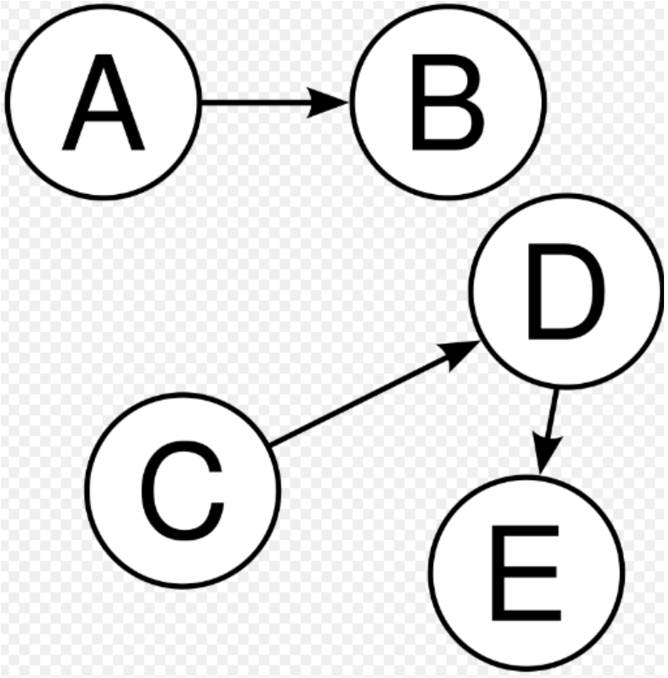
Exercise

Is it a tree?



Exercise

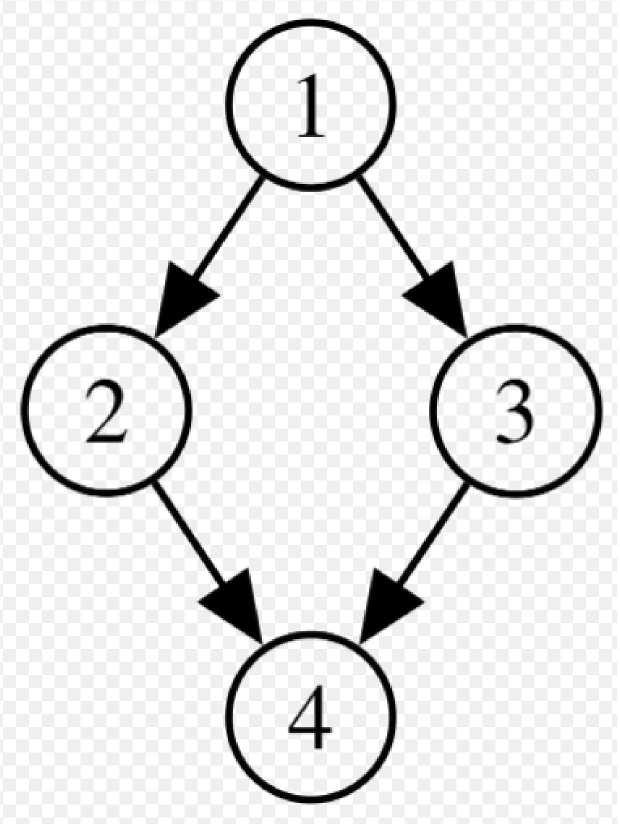
Is it a tree?



 No, because the tree has more than one root

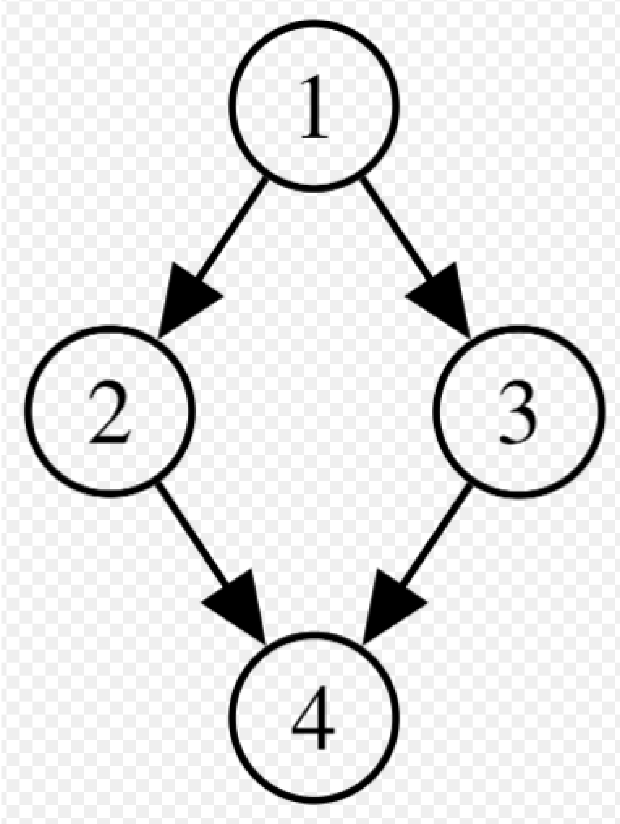
Exercise

Is it a tree?



Exercise

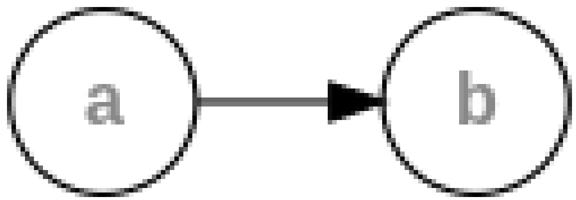
Is it a tree?



 No, because 4 has more than one parent

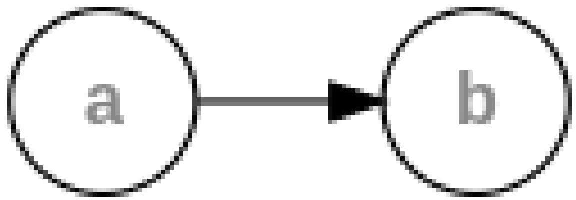
Exercise

Is it a tree?



Exercise

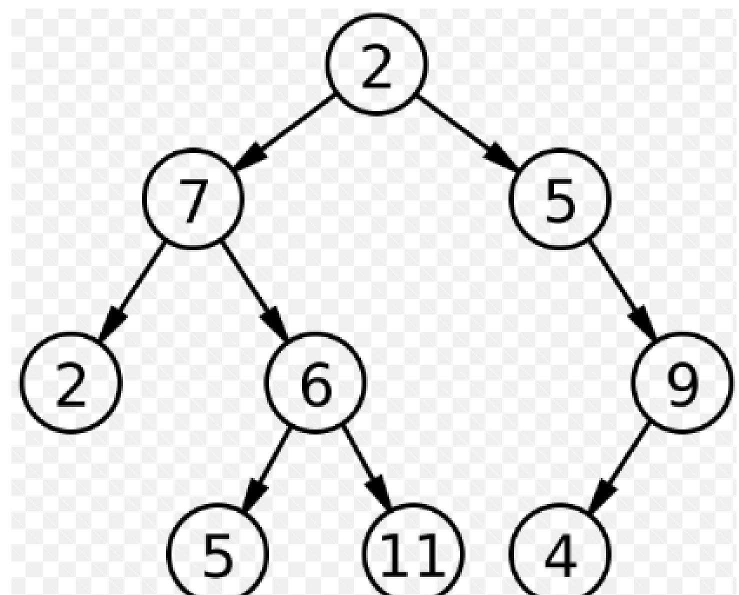
Is it a tree?



Yes. Linked List is trivially a tree

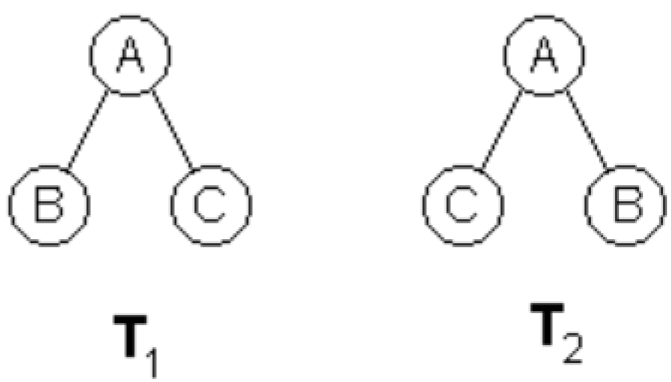
Binary Tree

- A binary tree is composed of zero or more nodes (At most 2)
- Each node contains:
  - A value (some sort of data item)
  - A reference or pointer to a left child (may be null), and
  - A reference or pointer to a right child (may be null)
- A binary tree may be empty (contain no nodes)
- If not empty, a binary tree has a root node
- Every node in the binary tree is reachable from the root node by a unique path
- A node with no left child and no right child is called a leaf
- In some binary trees, only the leaves contain a value



Ordered Tree

- A tree is ordered if there is a linear ordering defined for the children of the nodes
- A binary tree is an ordered tree in which every node has at most two children.
- If each node of a tree has either zero or two children, the tree is called a proper (strictly) binary tree.



If T_1 and T_2 are ordered trees then $T_1 \neq T_2$ else $T_1 = T_2$.

Ordered Tree

- A tree is ordered if there is a linear ordering defined for each child of each node.
- A binary tree is an ordered tree in which every node has at most two children.
- If each node of a tree has either zero or two children, the tree is called a proper (strictly) binary tree.

- The following two binary trees are different-