Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Trees – Part I

Data Structures

Fernando Marson
fernando.marson@universidadeeuropeia.pt

Trees

- Lists, queues, and stacks are linear data structures where each node is connected to at most two others: the previous and the next.

- These structures are not suitable for representing hierarchical data, such as folders in a directory, hyperlinks in HTML pages, organizational structures, and so on.

- Trees are fundamental structures to effectively represent several kinds of information, especially when dealing with hierarchical relationships and complex data organization.

- They provide a clear and efficient way to organize data, allowing easy navigation and retrieval of information.

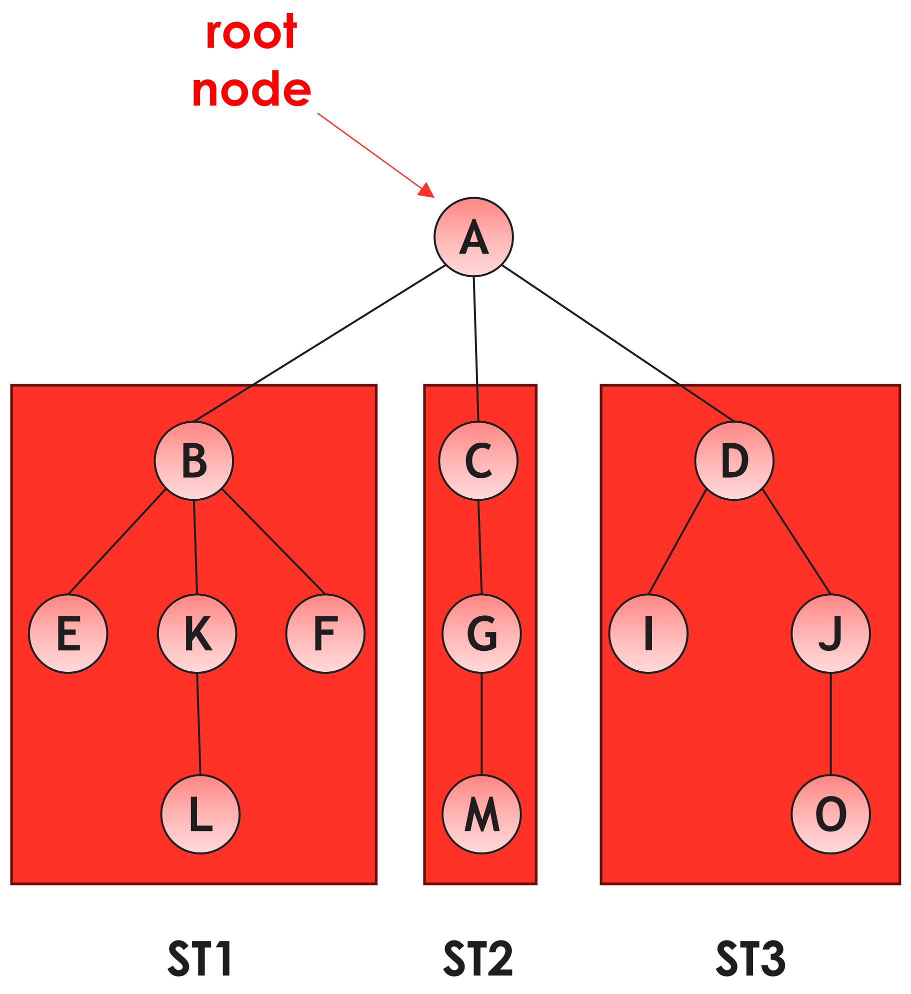
© Copyright European University. All rights reserved

Tree

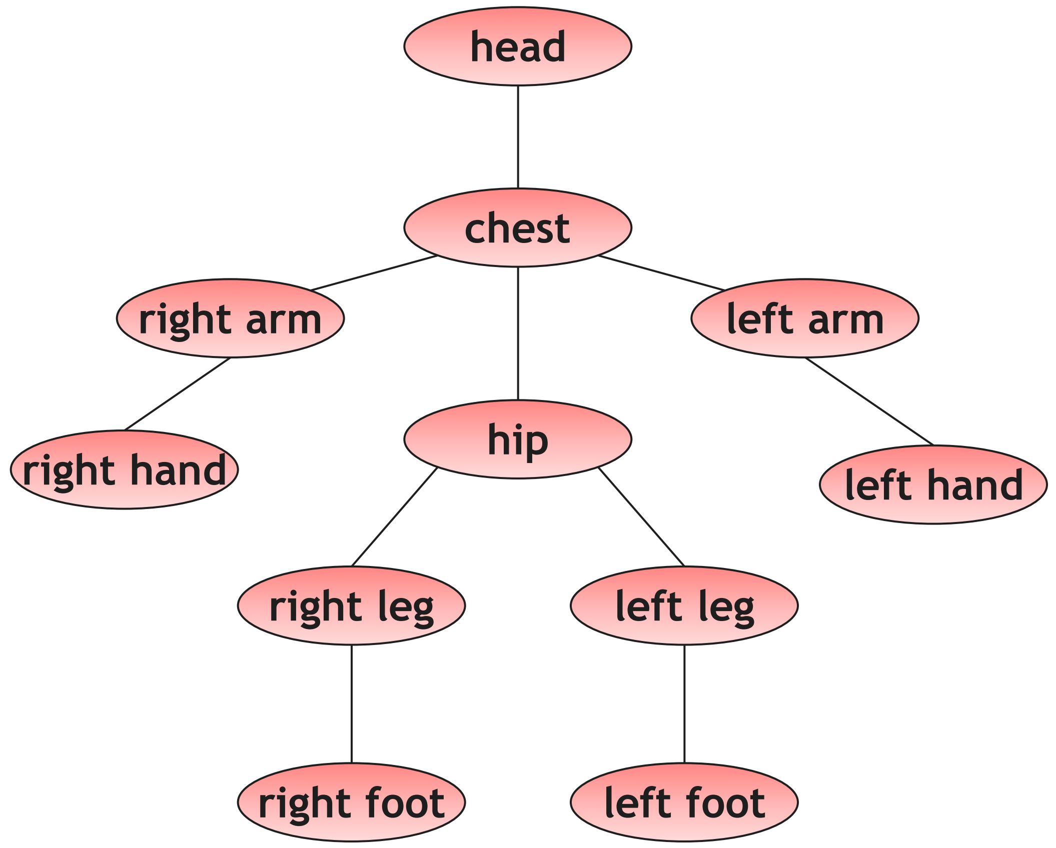
https://www.youtube.com/watch?v=RASniJbiWaE

Trees

- In a generic way, a tree is a collection of nodes structured hierarchically and allows for the establishment or representation of a hierarchy among the data stored in the tree.



Trees



© Copyright European University. All rights reserved

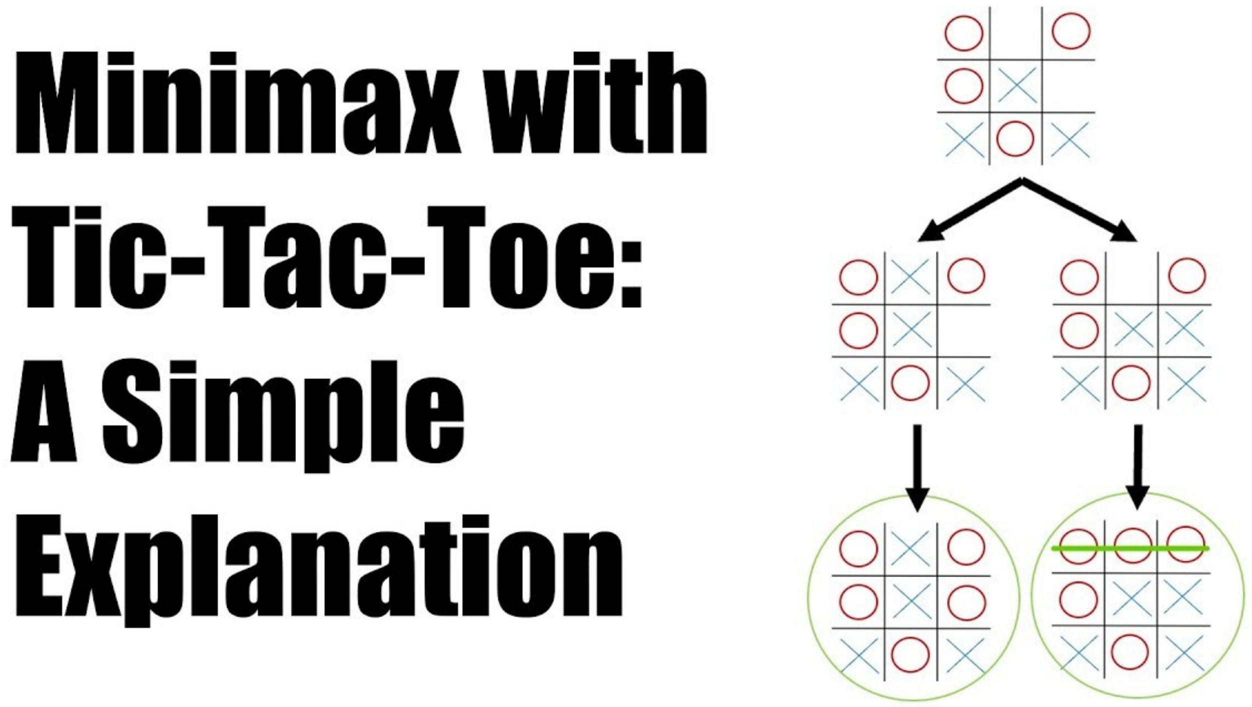
Trees – Computer Science

- **Hierarchical Data Representation**: Trees represent hierarchical data like file systems, organizational structures, and XML/JSON data.
- **Searching and Sorting**: Binary search trees facilitate efficient searching and sorting operations.
- **Databases**: B-trees and their variants are used in databases for indexing and to support quick data retrieval.
- **Artificial Intelligence**: Decision trees are used in machine learning for classification and regression tasks.

© Copyright European University. All rights reserved

Trees – Games

- **Game Trees**: Represent possible moves in a game (like chess or tic-tac-toe) to evaluate strategies and outcomes.

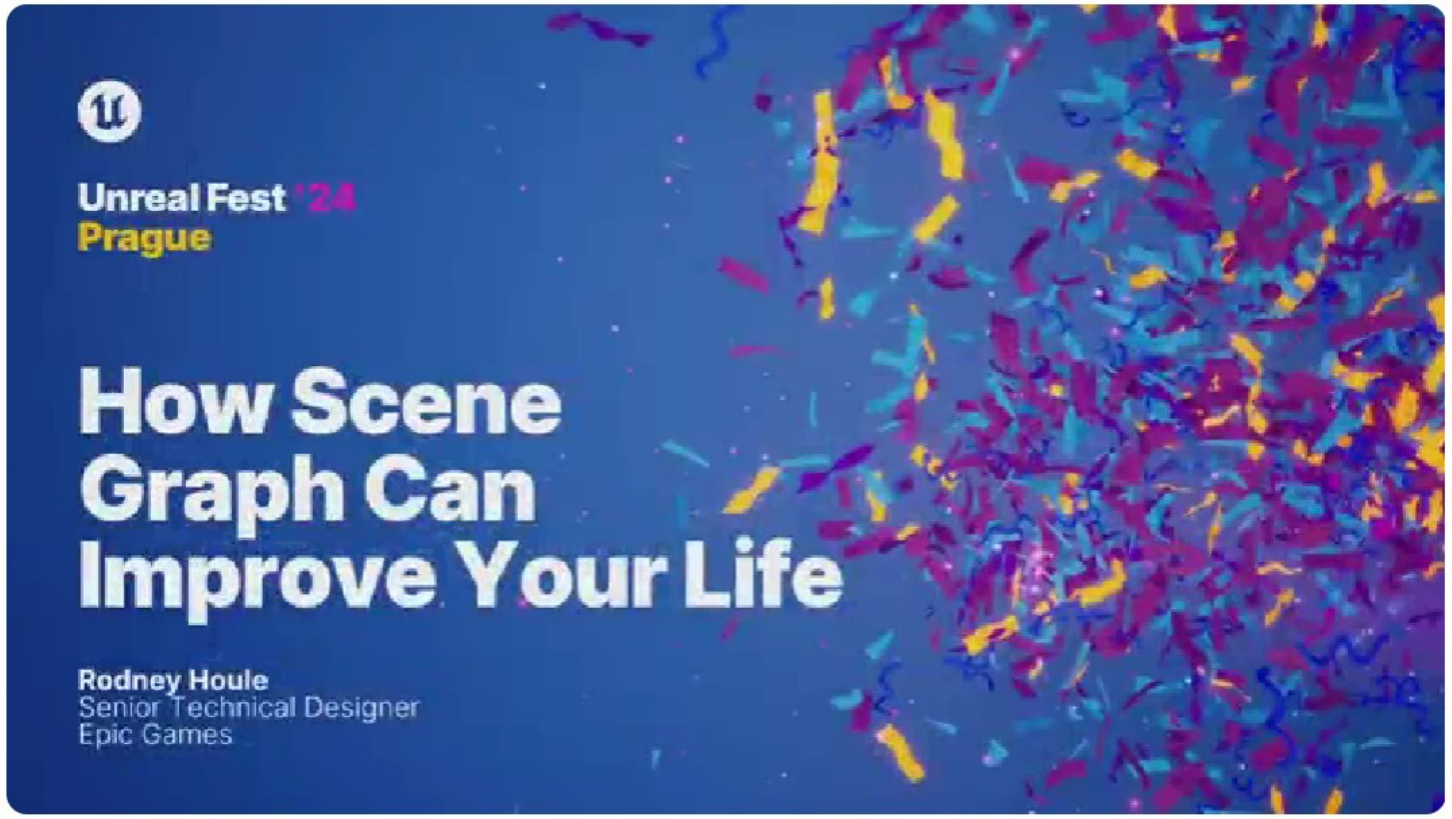


https://www.youtube.com/watch?v=l-hh51ncgDI

© Copyright European University. All rights reserved

Trees – Games

- **Scene Graphs**: In graphics engines, trees organize the spatial relationships of objects in a scene.


Unreal Fest '24
Prague
How Scene Graph Can Improve Your Life
Rodney Houle
Senior Technical Designer
Epic Games

https://youtu.be/p2aMVyRr7QY

© Copyright European University. All rights reserved

Trees – Games

- **Behavior Trees**: Used in AI for modeling the behavior of non-player characters (NPCs) through a hierarchical structure of tasks and conditions.

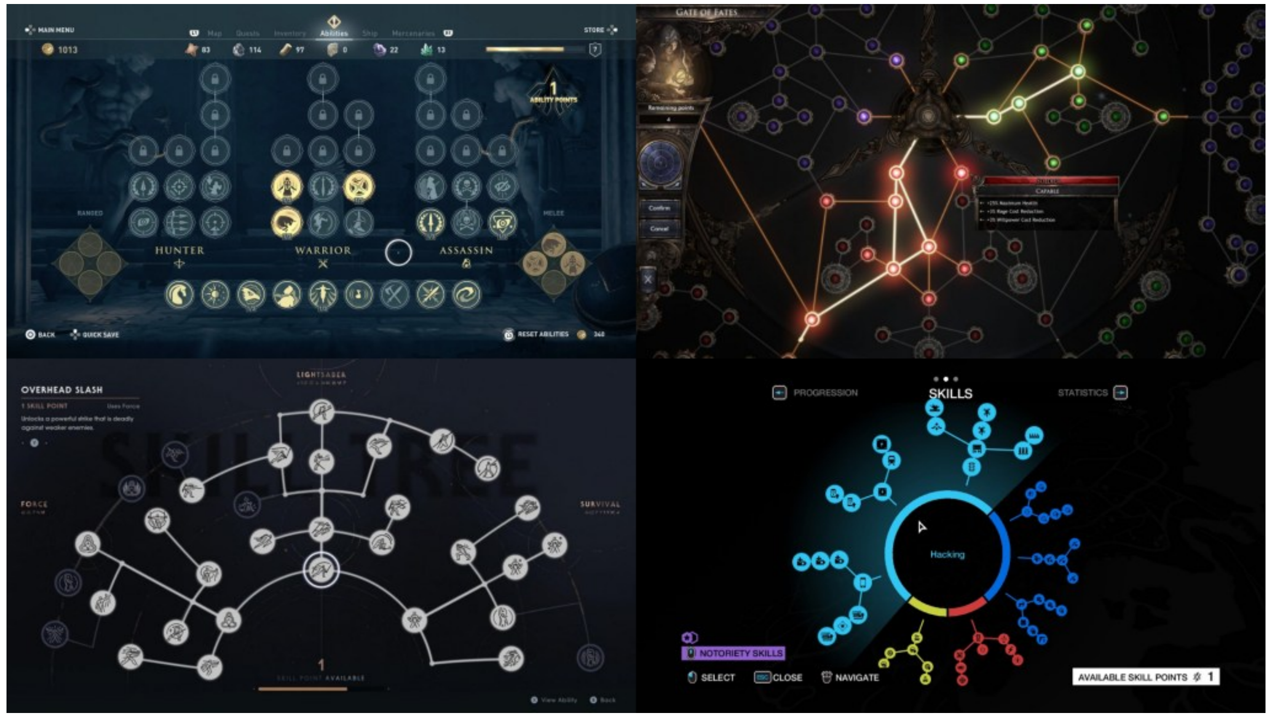
https://youtu.be/Qw5QX6gQC9c

© Copyright European University. All rights reserved

Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Trees – Games

- **Skill trees**: Used to allow players to unlock abilities or skills in a structured manner, often representing player progression and choices.

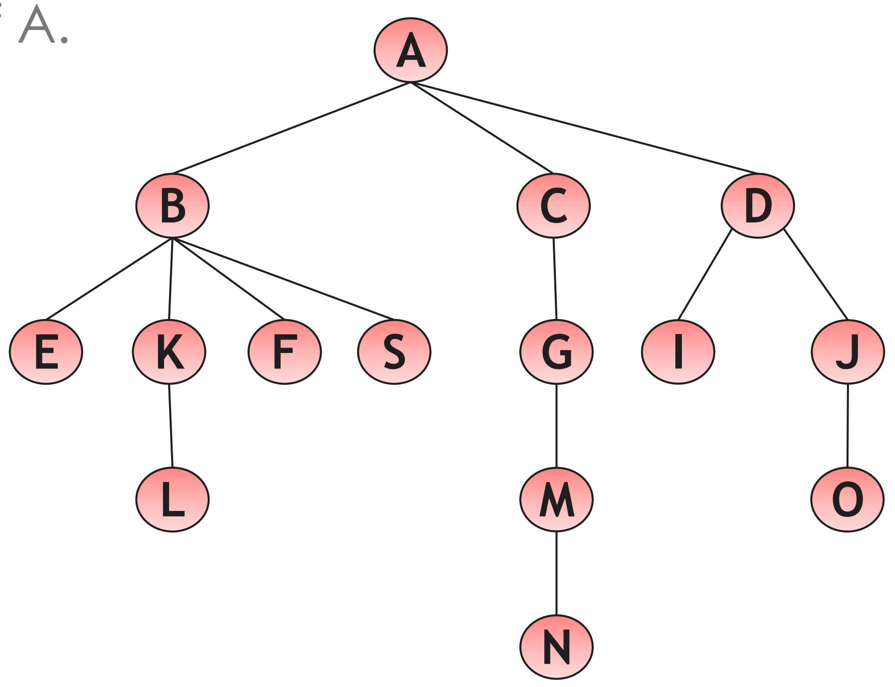


https://gdkeys.com/keys-to-meaningful-skill-trees

© Copyright European University. All rights reserved

Trees – Concepts

- **Root**
  - The node from which branches (subtrees) derive; the parent of all subtrees. In the example, node A is the root node.
- **Child**
  - A node that is directly below another node in the hierarchy. For example, node A has node B directly below it, then B is a child of A.
- **Parent**
  - A node that is directly above another node.
    In the example, node A is the parent of nodes B, C and D.
- **Siblings**
  - Nodes that share the same parent. Nodes B, C and D are both children of node A, they are considered siblings.



© Copyright European University. All rights reserved

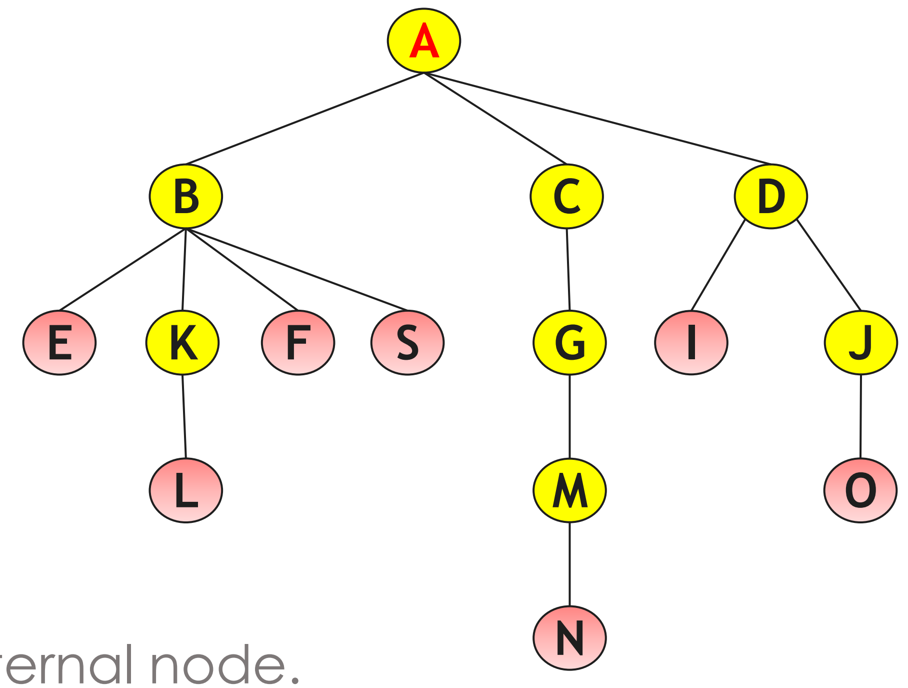
Trees – Concepts

- **Degree**
  - The number of subtrees (children) of a node.
  - The degree of node A is 3, as it has three children: B, C, and D.
  - The degree of node C is 1.

- **Degree of the Tree**
  - The maximum degree among all nodes in the tree.

- **Leaf node**
  - A node with no subtrees, i.e., with a degree of zero.

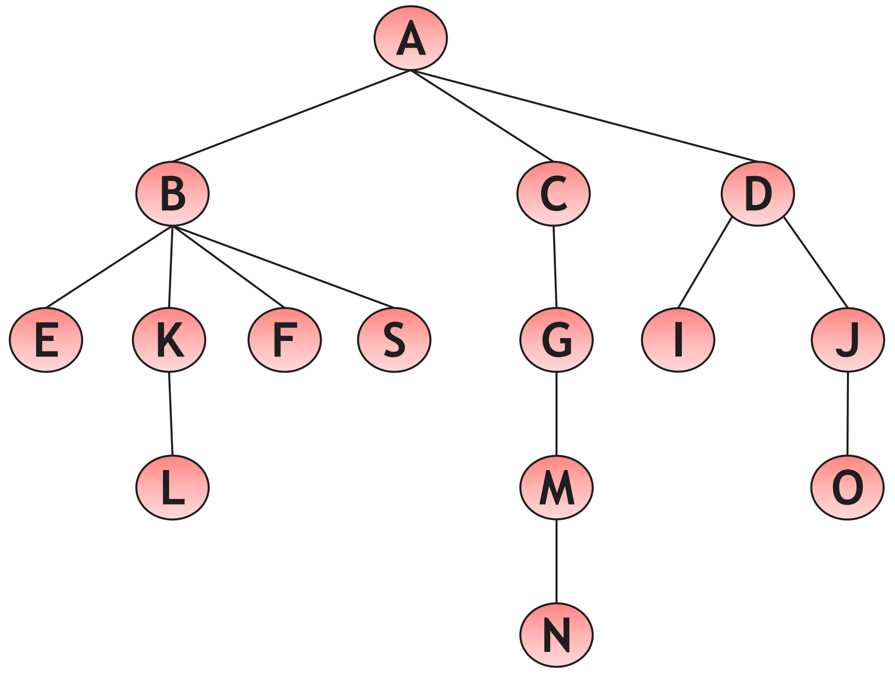
- **Internal node (non-leaf nodes or inner nodes)**
  - Are nodes that have at least one child. A root also can be an internal node.



© Copyright European University. All rights reserved

Trees – Concepts

- **Path**
  - A sequence of distinct nodes such that there are always consecutive nodes that have the relationship **is a child of** or **is a parent of**.
- **Length of the Path**
  - The number of nodes in the path minus one.
- **Level**
  - The length of the path from the root **R** to a node **N**.
  - The root is at level zero.



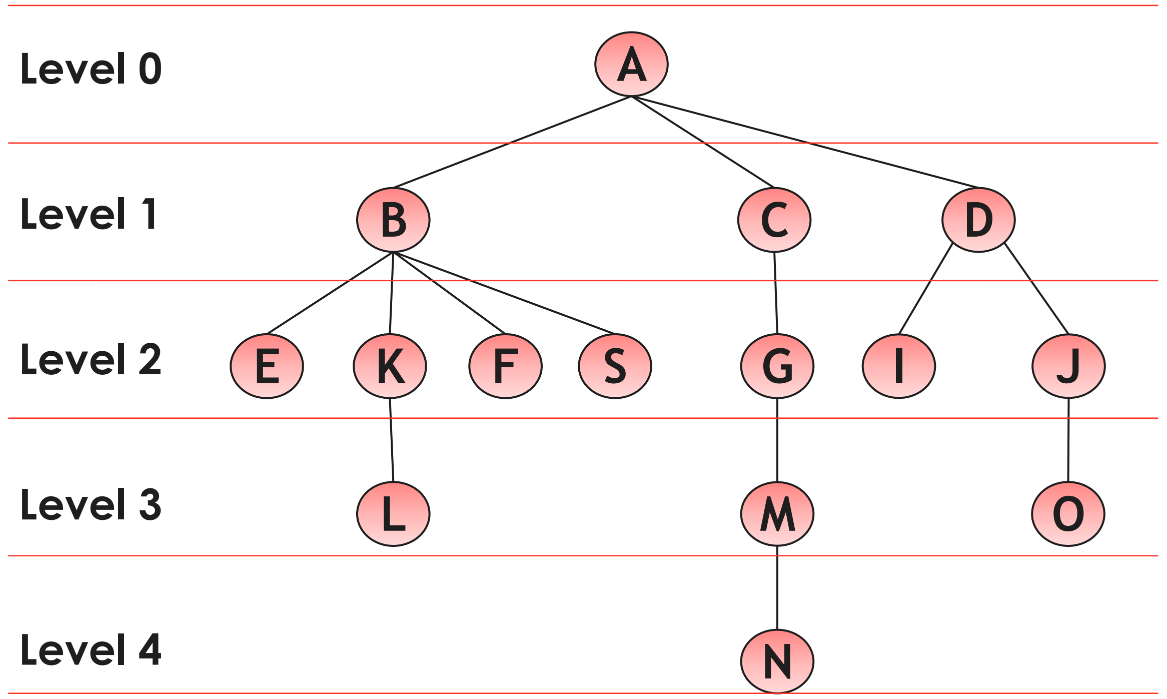
© Copyright European University. All rights reserved

Trees – Concepts

- **Path**
  - A sequence of distinct nodes such that there are always consecutive nodes that have the relationship **is a child of** or **is a parent of**.

- **Length of the Path**
  - The number of nodes in the path minus one.

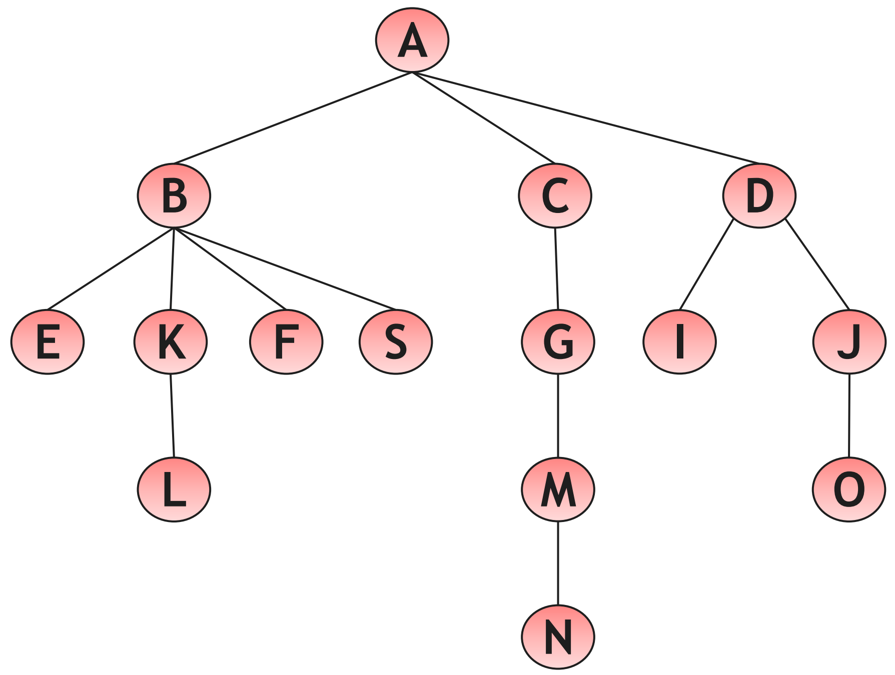
- **Level**
  - The length of the path from the root **R** to a node **N**.
  - The root is at level zero.



© Copyright European University. All rights reserved

Trees – Concepts

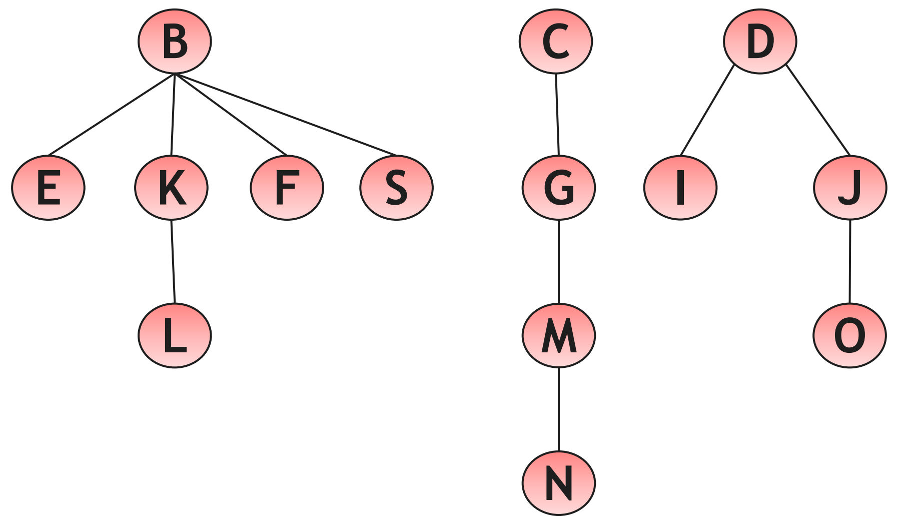
- **Height**
  - The height of the tree is the number of levels in the tree, not counting the root. This means that the height is measured from the first level below the root to the deepest level that contains leaves.
  - If the tree has **h** levels, the height would be **h−1**, where **h** is the total number of levels including the root.
  - In this example, the **height** is **4**.



© Copyright European University. All rights reserved

Trees – Concepts

- **Forest**
  - A forest is a collection of disjoint trees, meaning that it consists of multiple trees that do not share any nodes, without a root node linking them.



© Copyright European University. All rights reserved

Trees – Basic Operations

1) **Insertion of a new node**
   - As the root.
   - As a leaf.
   - In an intermediate position.

2) **Deletion of a specific node**

3) **Access to a node**
   a) Determines how to traverse the tree.
   b) Search for an item.
   c) Find the root of any node.

4) **Enumeration**
   - Of all items.
   - Of a subtree.

5) **Pruning**
   - Remove a subtree.

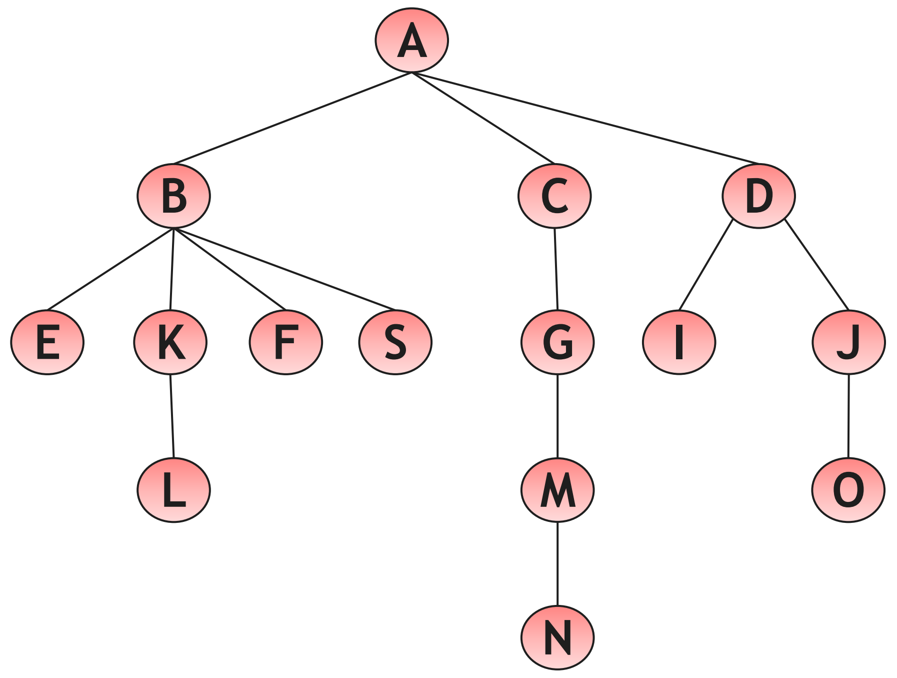
6) **Grafting**
   - Add a subtree.

7) **Destruction of the Tree**

© Copyright European University. All rights reserved

Trees

- Just as a linked list, a tree structure is based on nodes.
- Each node points to the subsequent nodes in the structure.
- Node A is the root of the tree;
- Node B is the root of nodes E, K, F and S;
- Node K is the root of node L;
- Disjoint, B, C, and D make up a forest;
- Joined by A, they form a tree;
- If there is at least one child, each node can become the root of a subtree.



Due to the **features** outlined above, it is possible to use **recursion** to implement trees, as **each subtree** can be treated as **a smaller tree** with the **same properties**.

© Copyright European University. All rights reserved