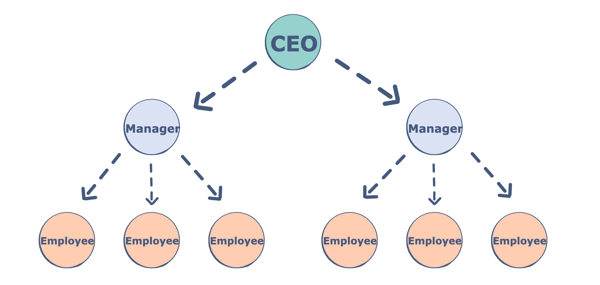
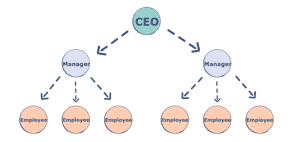
**What is a tree?**

[Data structures](https://www.educative.io/blog/learn-how-to-code-beginners-guide) are used to store and organize data. We can use algorithms to manipulate and use our data structures. Different types of data are organized more efficiently by using different data structures.

**Trees** are non-linear data structures. They are often used to represent hierarchical data. For a real-world example, a hierarchical company structure uses a tree to organize.





**Components of a Tree**

Trees are a collection of nodes (vertices), and they are linked with edges (pointers), representing the hierarchical connections between the nodes. A node contains data of any type, but all the nodes must be of the same data type. Trees are similar to graphs, but a cycle cannot exist in a tree. What are the different components of a tree?

**Root:** The root of a tree is a node that has no incoming link (i.e. no parent node). Think of this as a starting point of your tree.

**Children:** The child of a tree is a node with one incoming link from a node above it (i.e. a parent node). If two children nodes share the same parent, they are called siblings.

**Parent:** The parent node has an outgoing link connecting it to one or more child nodes.

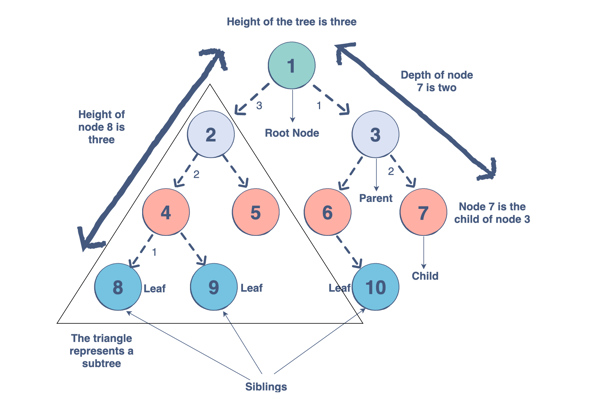
**Leaf:** A leaf has a parent node but has no outgoing link to a child node. Think of this as an endpoint of your tree.

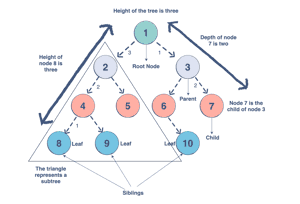
**Subtree:** A subtree is a smaller tree held within a larger tree. The root of that tree can be any node from the bigger tree.

**Depth:** The depth of a node is the number of edges between that node and the root. Think of this as how many steps there are between your node and the tree’s starting point.

**Height:** The height of a node is the number of edges in the longest path from a node to a leaf node. Think of this as how many steps there are between your node and the tree’s endpoint. The height of a tree is the height of its root node.

**Degree:** The degree of a node refers to the number of sub-trees.





**Why do we use trees?**

Trees can be applied to many things. The hierarchical structure gives a tree unique properties for storing, manipulating, and accessing data. Trees form some of the most basic organization of computers. We can use a tree for the following:

* **Storage as hierarchy.** Storing information that naturally occurs in a hierarchy. File systems on a computer and PDF use tree structures.
* **Searching.** Storing information that we want to search quickly. Trees are easier to search than a Linked List. Some types of trees (like AVL and Red-Black trees) are designed for fast searching.
* [**Inheritance**](https://www.educative.io/blog/java-inheritance-tutorial)**.** Trees are used for inheritance, XML parser, machine learning, and DNS, amongst many other things.
* **Indexing.** Advanced types of trees, like B-Trees and B+ Trees, can be used for indexing a database.
* **Networking.** Trees are ideal for things like social networking and computer chess games.
* **Shortest path.** A Spanning Tree can be used to find the shortest paths in routers for networking.
* and much more