Today …

Trees
  definition
  properties
  traversals
Trees

List, Stacks, Queues are linear data structures
Trees

List, Stacks, Queues are linear data structures

Trees allow for hierarchical relationships

nodes have parent-child relation
There is a unique path from the root to each node in the tree.
A tree is either empty or a root node connected to 0 or more trees (called subtrees).
A tree is either empty or a root node connected to 0 or more trees (called subtrees).
A tree is either empty or a root node connected to 0 or more trees (called subtrees).
A tree is either empty or a root node connected to 0 or more trees (called subtrees).
Trees (jargon)

Each node is either a **leaf** or an **internal node**

- an **internal node** has one or more children
- a **leaf node** (external node) has no children
Each node is either a **leaf** or an **internal node**. An **internal node** has one or more children. A **leaf node** (external node) has no children. Nodes with the same parent are **siblings**.
Paths

Root

Diagram of a tree structure with the root node labeled 'Root'.
A path from node $v_0$ to $v_n$ is a sequence of nodes $v_0, v_1, v_2, \ldots, v_n$, where there is an edge from one node to the next.
A path from node $v_0$ to $v_n$ is a sequence of nodes $v_0, v_1, v_2, \ldots, v_n$, where there is an edge from one node to the next.

The descendants of a node $v$ are all nodes reached by a path from node $v$ to the leaf nodes.
A path from node $v_0$ to $v_n$ is a sequence of nodes $v_0, v_1, v_2, \ldots, v_n$, where there is an edge from one node to the next.

The descendants of a node $v$ are all nodes reached by a path from node $v$ to the leaf nodes.

The ancestors of a node $v$ are all nodes found on the path from the root to node $v$. 

Paths
Depth and Height
Depth and Height

The length of a path is the number of edges in the path.

length = 3
Depth and Height

The **length** of a path is the number of edges in the path.

The **depth** (level) of a node \(v\) is the length of the path from \(v\) to root.
Depth and Height

The length of a path is the number of edges in the path.

The depth (level) of a node $v$ is the length of the path from $v$ to root.

The height of a node $v$ is the length of the path from $v$ to its deepest descendant.
Properties

[Diagram of a tree structure]
Properties

Depth of tree is the depth of the deepest node.
Properties

Depth of tree is the depth of the deepest node.

Height of tree is the height of the root.
Linked Structure for Trees

Every node has:

- data
- parent
- children array
k-ary trees
k-ary trees

In a k-ary tree, every node has between 0 and k children.
k-ary trees

In a **k-ary tree**, every node has between 0 and \( k \) children.

In a **full (proper) k-ary** tree, every node has exactly 0 or \( k \) children.
k-ary trees

In a **k-ary tree**, every node has between 0 and k children.

In a **full (proper) k-ary** tree, every node has exactly 0 or k children.

In a **complete k-ary** tree, every level is entirely filled, except possibly the deepest, where all nodes are as far left as possible.
k-ary trees

In a **k-ary tree**, every node has between 0 and k children.

In a **full (proper) k-ary** tree, every node has exactly 0 or k children.

In a **complete k-ary** tree, every level is entirely filled, except possibly the deepest, where all nodes are as far left as possible.

In a **perfect k-ary** tree, every leaf has the same depth and the tree is full.
Traversals
Preorder Traversal

**algorithm** preorder(p) {
    visit(p)
    for each child c of p {
        preorder(c)
    }
}
Postorder Traversal

algorithm postorder(p) {
    for each child c of p {
        postorder(c)
    }
    visit(p)
}
Example: What type of traversal?

Compute space used by files in folders and subfolders:

```
$ du -h -d 2
```