Linear structures are not ideal in any sort of **hierarchical** model.

Tree:

* Abstract data structure, hierarchical storage. Information stored in the nodes.
* Non-empty set of nodes storing useful information, in a **parent-child relationship**.

Consists of: root node + arbitrary nodes (elements). Each node has a single parent, but an arbitrary number of children. Leaf nodes have no children.

Leaf - external node, no children

Children implies an internal node.

Subtree - set of all descendants, from a root node in the original tree.

Ordered tree - if a **linear ordering relation** is defined for all children of each node. An order can be defined among them. (think of a book)

Binary tree - each node can only have at most 2 children. Are ordered. Left child should always be lower than the right child in the ordered relationship.

Proper Binary Tree - if each internal node has exactly 2 children

Accessor Methods

root() - returns root node of the tree

parent(v) - returns the parent of node v

child(v) - returns an iterator of the children of node v

Query Methods

isInternal(v)

isExternal(v)

isRoot(v)

Generic Methods

size()

elements() - iterator for all elements stored at nodes in the tree

positions() - iterator for all nodes in tree

swapElements(v,w)

replaceElement(v,e)

Node Depth - The depth of a node v, is defined as the number of **ancestors** from the root to itself (not inclusive)

Root depth = 0

Tree Height - Depth from root to deepest leaf node.

Tree Traversal

Systematic way of accessing all nodes of the tree, respecting the hierarchy of the tree

**Pre-order** - visit a node, perform an action, then visit all sub-trees rooted in it’s children. Respects the ordering of the tree. Used where parental nodes are ordered before children. Of running time O(n \* CA) where n is number of nodes, and CA is the cost of the action.

**Post-order -** Traverse the tree, reaching leaf nodes before performing the defined operation. Useful in for example, defining the size of each directory in a file structure.

Binary Trees

leftChild(v)

rightChild(v)

sibling(v)

Binary trees have another traversal mechanism, **Inorder traversal**

Inorder - Action happens **between** each node. GoLeft, performAction, goRight