# **General Tree Introduction**

## **General Tree Definition**

A **general tree** is a set ***T*** of one or more **nodes** such that ***T*** is partitioned into disjoint subsets:

* A single node r, the root
* Sets that are general trees, called subtrees of r

## **General Tree Implementation**

A tree is a **collection** of **nodes**.

One way to implement a tree would be to have in each **node**

* Data
* A link to each child of the node

However, in a general tree, the number of children per node can greatly vary.

Since there is no way to know how many children a node can have in advance, it is infeasible to make the children direct links in the data structure, because there would be **too much wasted space**.

The solution is simple: **Keep the children of each node in a linked list of tree nodes**.

## **TreeNode Declaration**

**class TreeNode**

**{**

**Object element;**

**TreeNode firstChild;**

**TreeNode nextSibling;**

**}**

## **Tree Representation**

The following figure shows the **first child/next sibling** representation of the tree.

* Arrows that point downward are firstChild links.
* Horizontal arrows are nextSibling links.
* Null links are not drawn, because there are too many.

A picture containing watch

Description automatically generated

In the tree, node *E* has two links

* a link to a sibling (*F*)
* a link to a child (*I*)

All nodes have these pointers, but some will not point to anything

For example, Node *N* does not link to a sibling or a child.