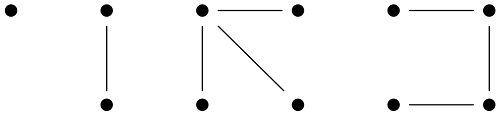
tree

Canonical name	Tree
Date of creation	2013-03-22 12:28:57
Last modified on	2013-03-22 12:28:57
Owner	Logan (6)
Last modified by	Logan (6)
Numerical id	9
Author	Logan (6)
Entry type	Definition
Classification	msc 05C05
Related topic	Graph
Related topic	TreeTraversals
Related topic	BinaryTree
Related topic	BalancedTree
Related topic	SpanningTree
Related topic	NullTree
Related topic	GraphTheory
Related topic	BetheLattice
Defines	free tree
Defines	rooted tree
Defines	forest

Formally, a *forest* is an undirected, acyclic graph. A forest consists of *trees*, which are themselves acyclic, connected graphs. For example, the following diagram represents a forest, each connected component of which is a tree.



All trees are forests, but not all forests are trees. As in a graph, a forest is made up of vertices (which are often called nodes interchangeably) and edges. Like any graph, the vertices and edges may each be labelled — that is, associated with some atom of data. Therefore a forest or a tree is often used as a data structure.

Often a particular node of a tree is specified as the *root.* Such trees are typically drawn with the root at the top of the diagram, with all other nodes depending down from it (however this is not always the case). A tree where a root has been specified is called a *rooted tree.* A tree where no root has been specified is called a *free tree.* When speaking of tree traversals, and most especially of trees as datastructures, rooted trees are often implied.

The edges of a rooted tree are often treated as directed. In a rooted tree, every non-root node has exactly one edge that leads to the root. This edge can be thought of as connecting each node to its *parent.* Often rooted trees ae considered directed in the sense that all edges connect parents to their children, but not vice-versa. Given this parent-child relationship, a *descendant* of a node in a directed tree is defined as any other node reachable from that node (that is, a node's children and all their descendants).

Given this directed notion of a rooted tree, a *rooted subtree* can be defined as any node of a tree and all of its descendants. This notion of a rooted subtree is very useful in dealing with trees inductively and defining certain algorithms inductively.

Because of their simple structure and unique properties, trees and forests have many uses. Because of the simple definition of various tree traversals, they are often used to store and lookup data. Many algorithms are based upon trees, or depend upon a tree in some manner, such as the heapsort algorithm or Huffman encoding. There are also a great many specific forms and families of trees, each with its own constraints, strengths, and weaknesses.