Trees

Introduction and Terminology

Trees

A tree is a finite set of nodes:

- it may be empty, in which case it’s called an empty tree (no nodes)
- in a non-empty tree, there is one special node called the “root” of the tree; it is the top-most element and has no parent
- all other nodes have exactly one parent (again, the root has no parent)
- each node can have one or more different associated nodes, called its children
- each node has some data of its own

Terminology

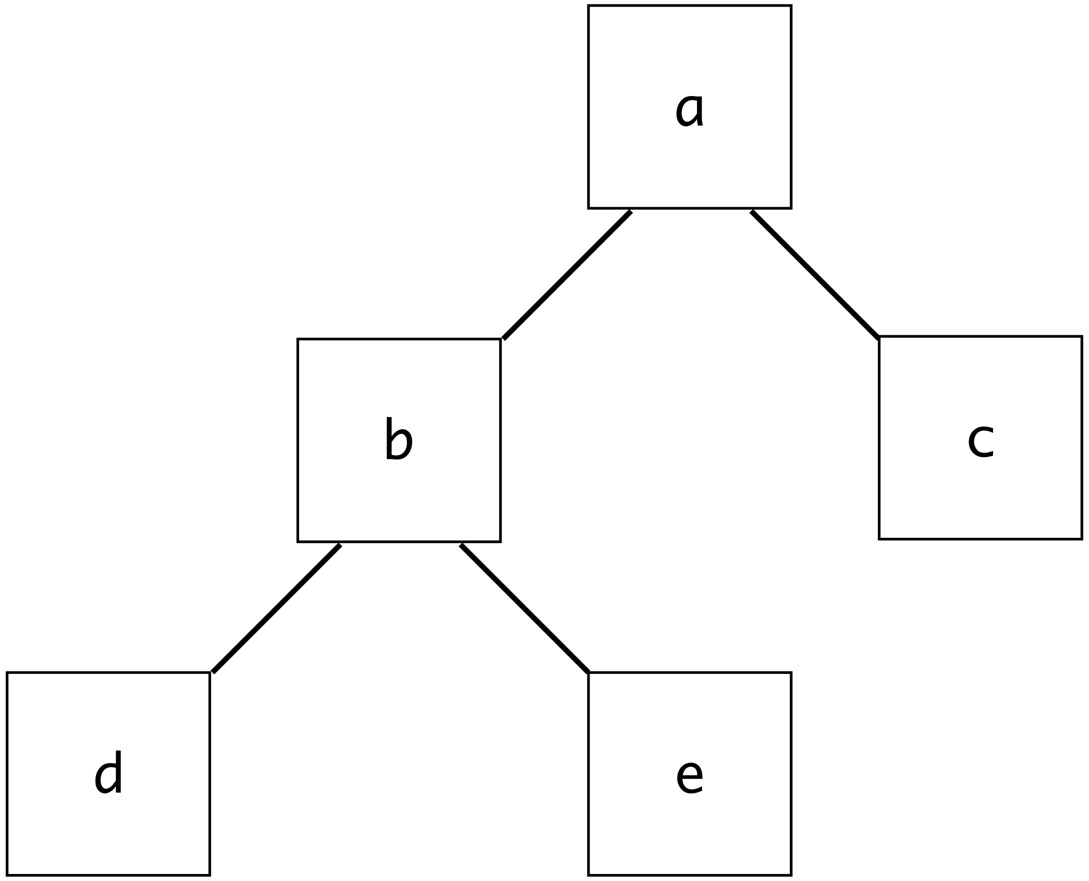
Trees have a lot of (admittedly simple) terminology:

- node
- root
- leaf
- ancestor
- parent
- descendant
- child
- sibling
- subtree
- node depth
- tree depth

Trees

Nodes of a tree

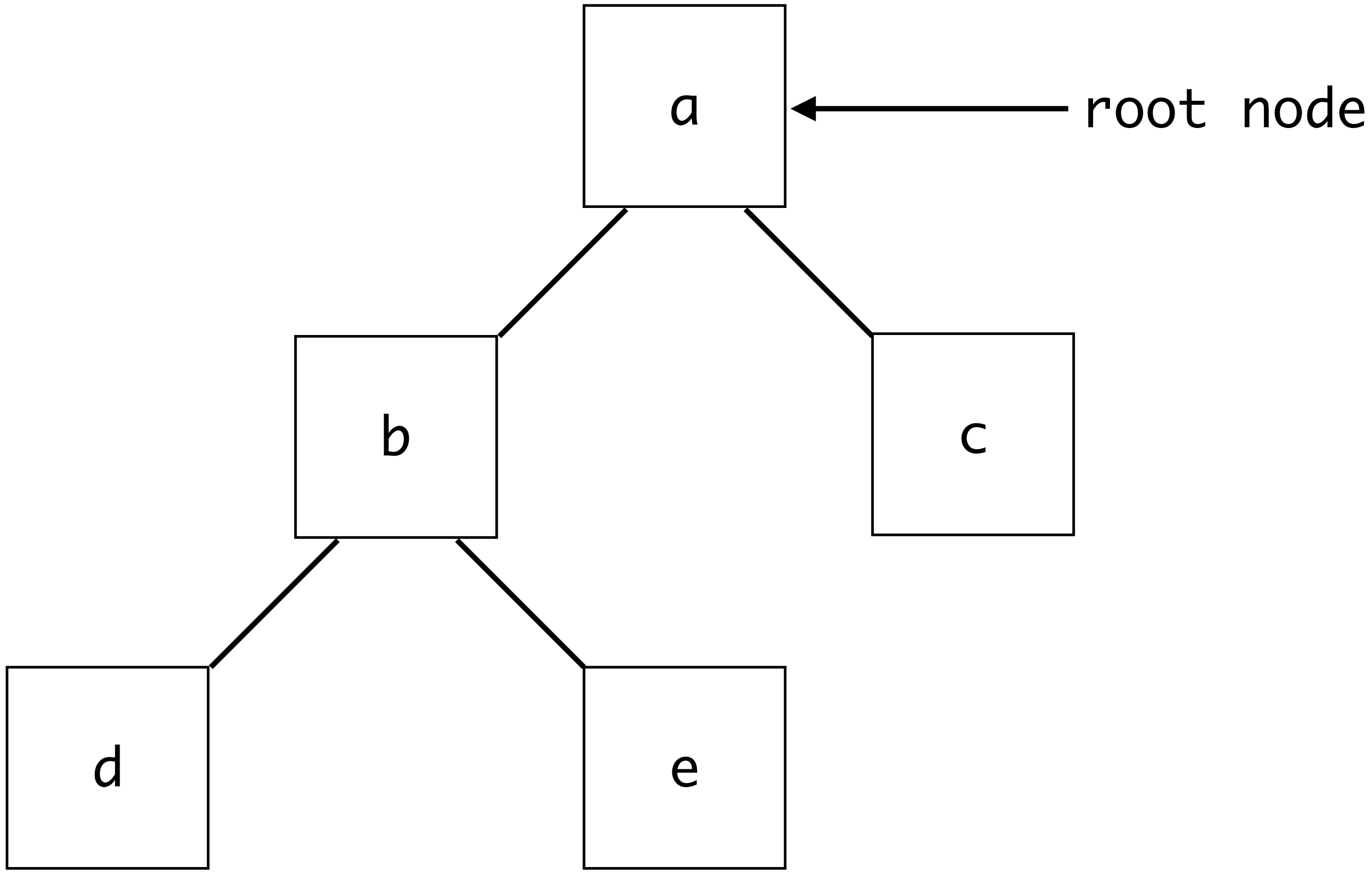
- a tree consists of zero or more individual nodes
- each box in the diagram below represents a single node



Trees

The root of a tree:

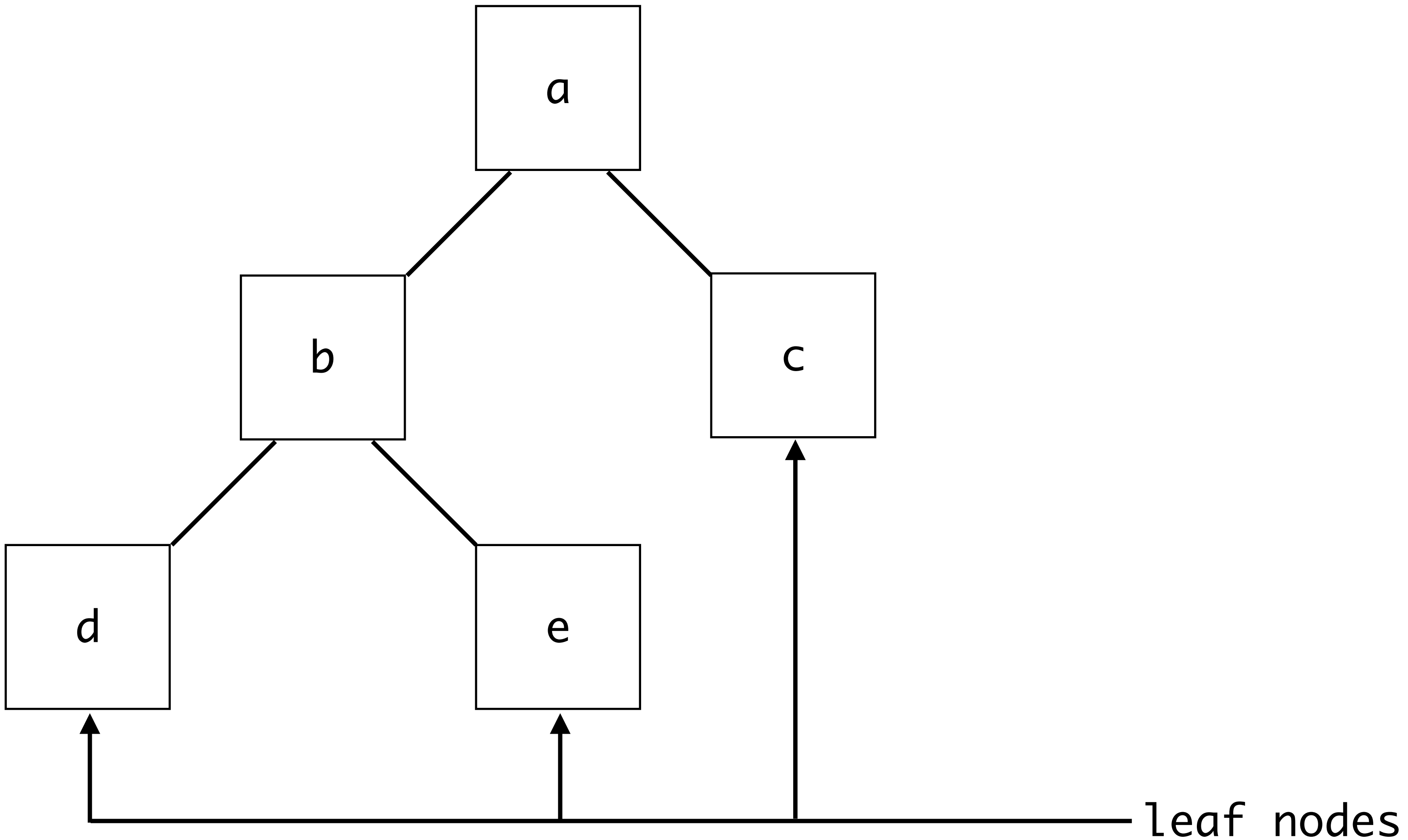
- the top-most node of a tree
- a tree has at most one root node (none when the tree is empty)



Trees

Leaf nodes:

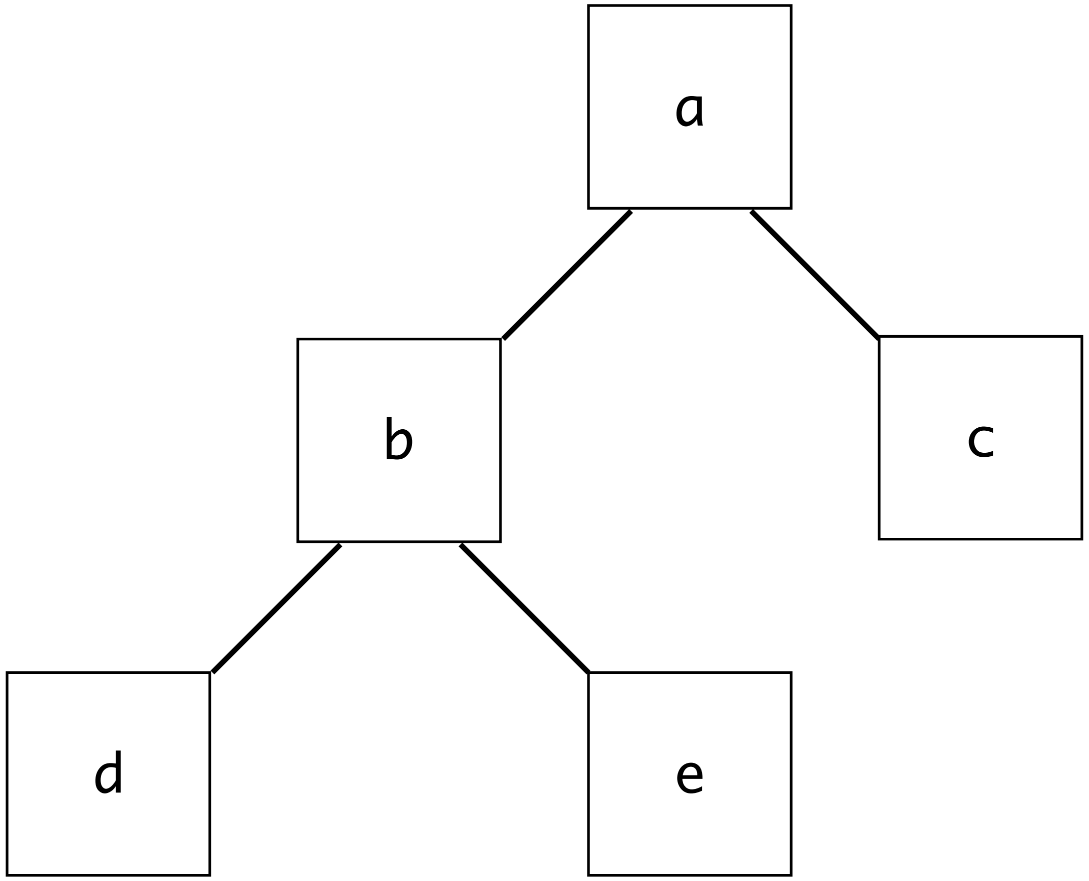
- nodes at the bottom of the tree that have no other nodes beneath them (no children)
- these can be considered the endpoints of a tree



Trees

Ancestor nodes:

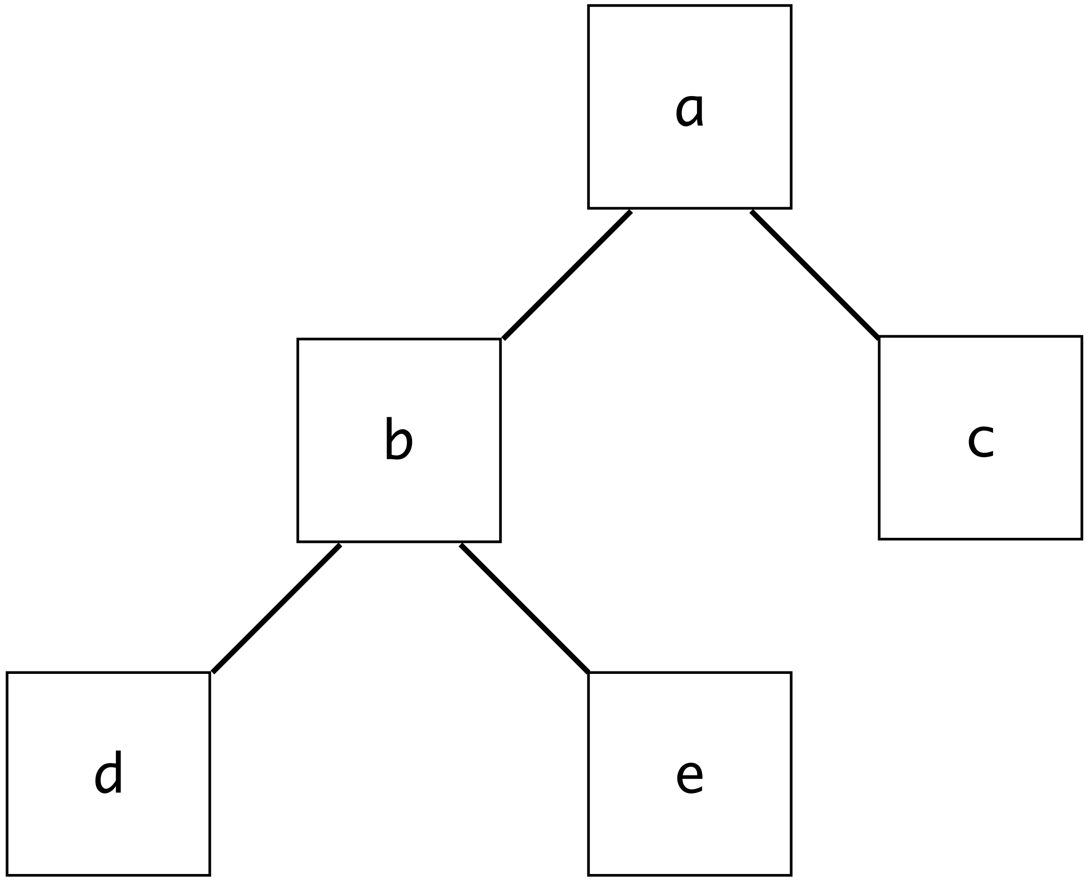
- the nodes appearing directly above (closer to the root) any given node are its ancestors
- in the diagram below, nodes *b* and *a* are the ancestors of node *d* (*c* is not).
- the root node is the ancestor of all other nodes in a tree



Trees

Parent nodes:

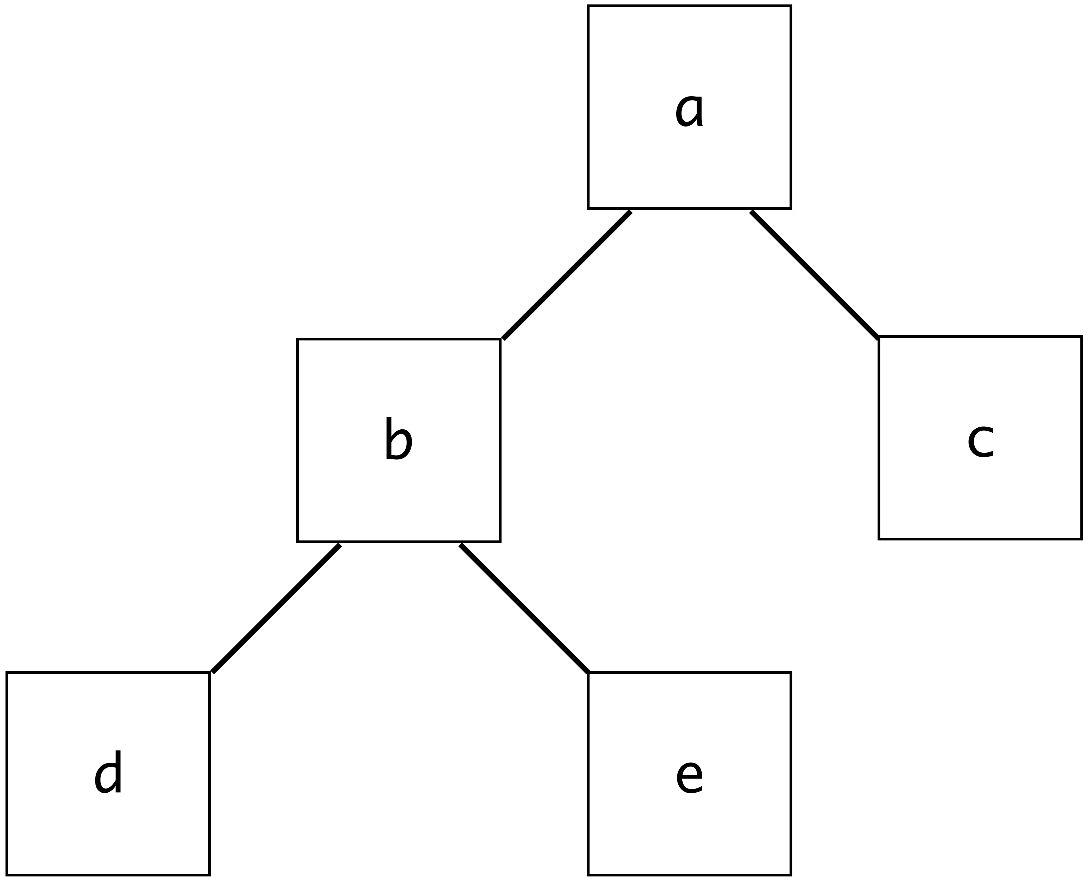
- the immediate ancestor of a node is called its parent node
- in the diagram below, node *b* is the parent of nodes *d* and *e*, and node *a* is the parent of nodes *b* and *c*.



Trees

Descendant nodes:

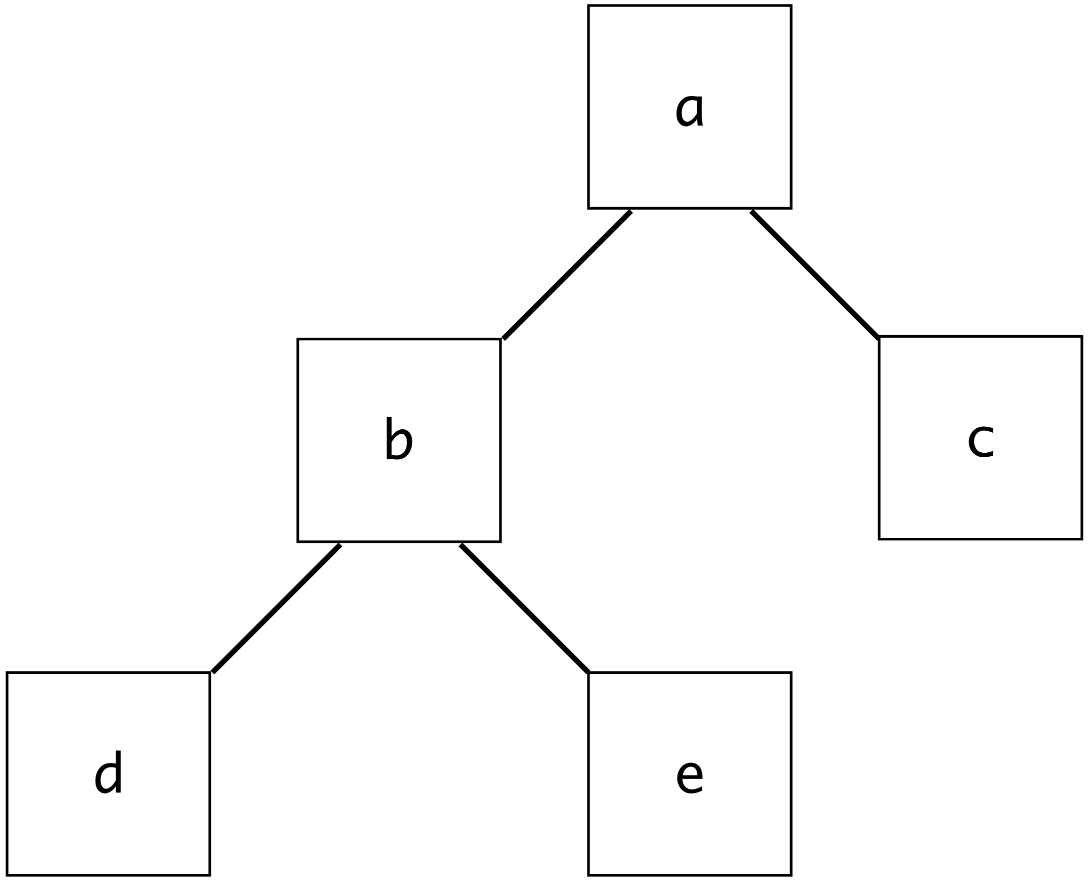
- the nodes appearing directly below any given node are its descendants
- in the diagram below, nodes d and e are the descendants of node b.
- all nodes are the descendants of the root node (except the root node itself, of course)



Trees

Child nodes:

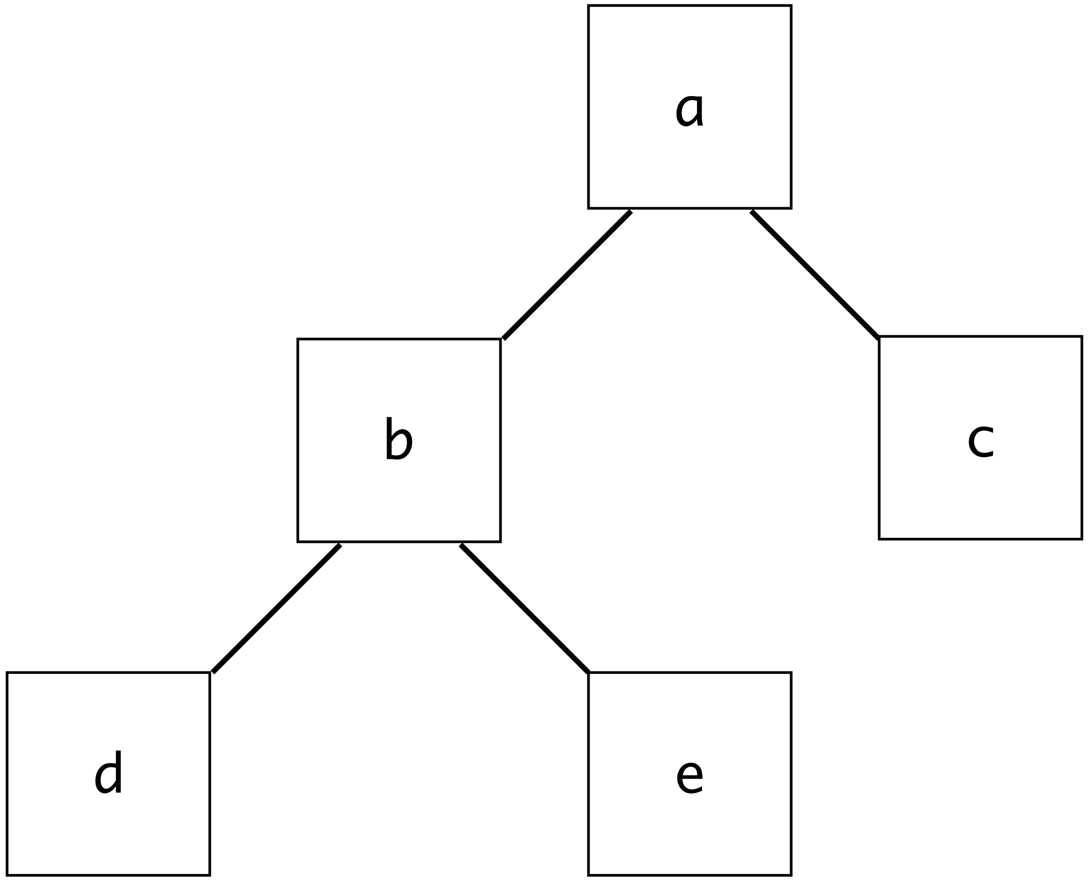
- the immediate descendants of a node are called its children, or child nodes
- in the diagram below, nodes *d* and *e* are the child nodes of node *b*, and nodes *b* and *c* are the child nodes of node *a*.



Trees

Sibling nodes:

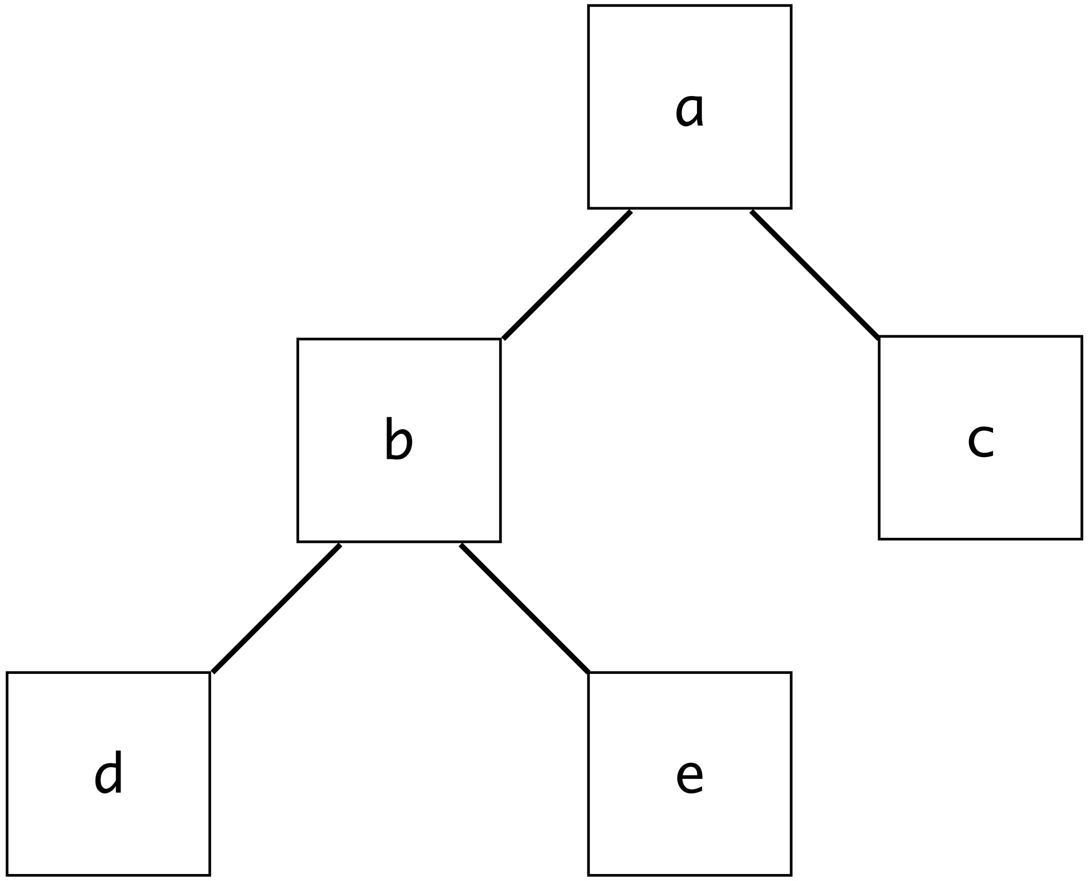
- two nodes are siblings if they share the same parent
- in the diagram below, nodes d and e are each other's siblings; nodes b and c are also siblings



Trees

Depth of a node:

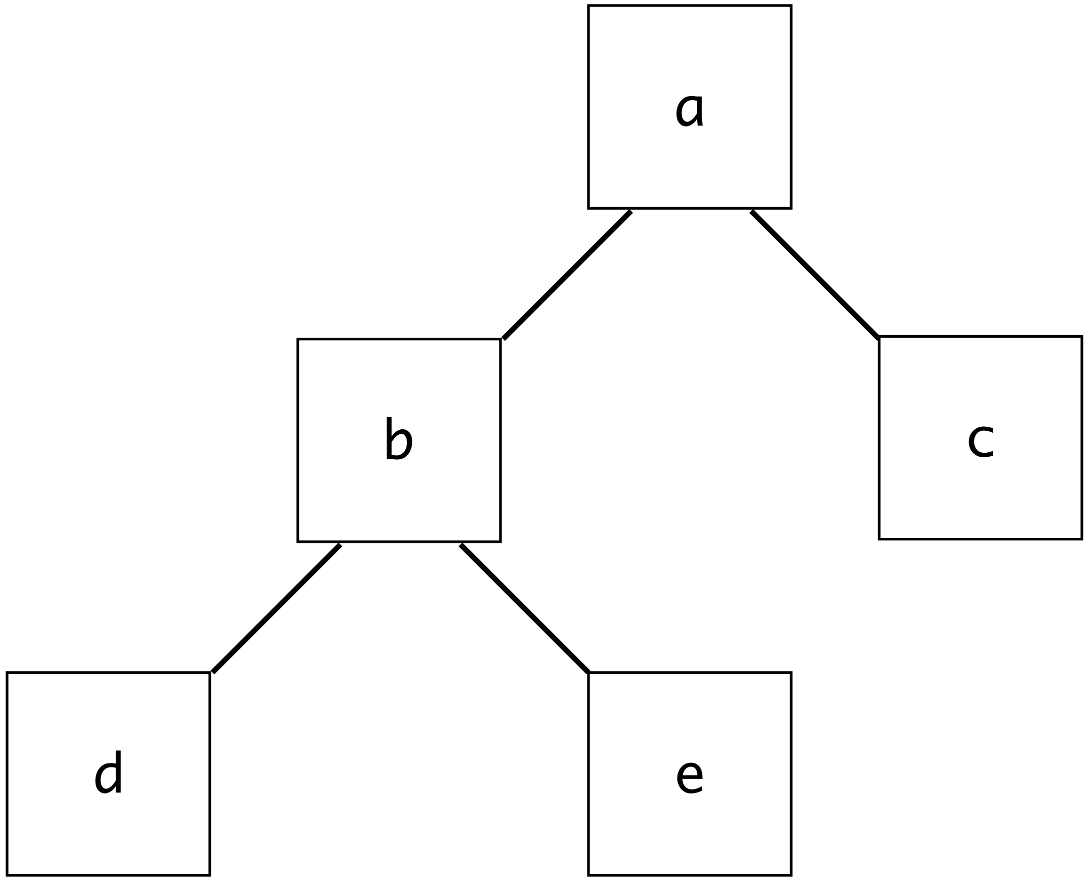
- the depth of a node is the number of steps (moves to a parent node) required to reach the root
- the depth of the root node is 0; the depth of nodes b and c is 1, and the depth of nodes d and e is 2.



Trees

Depth of the tree:

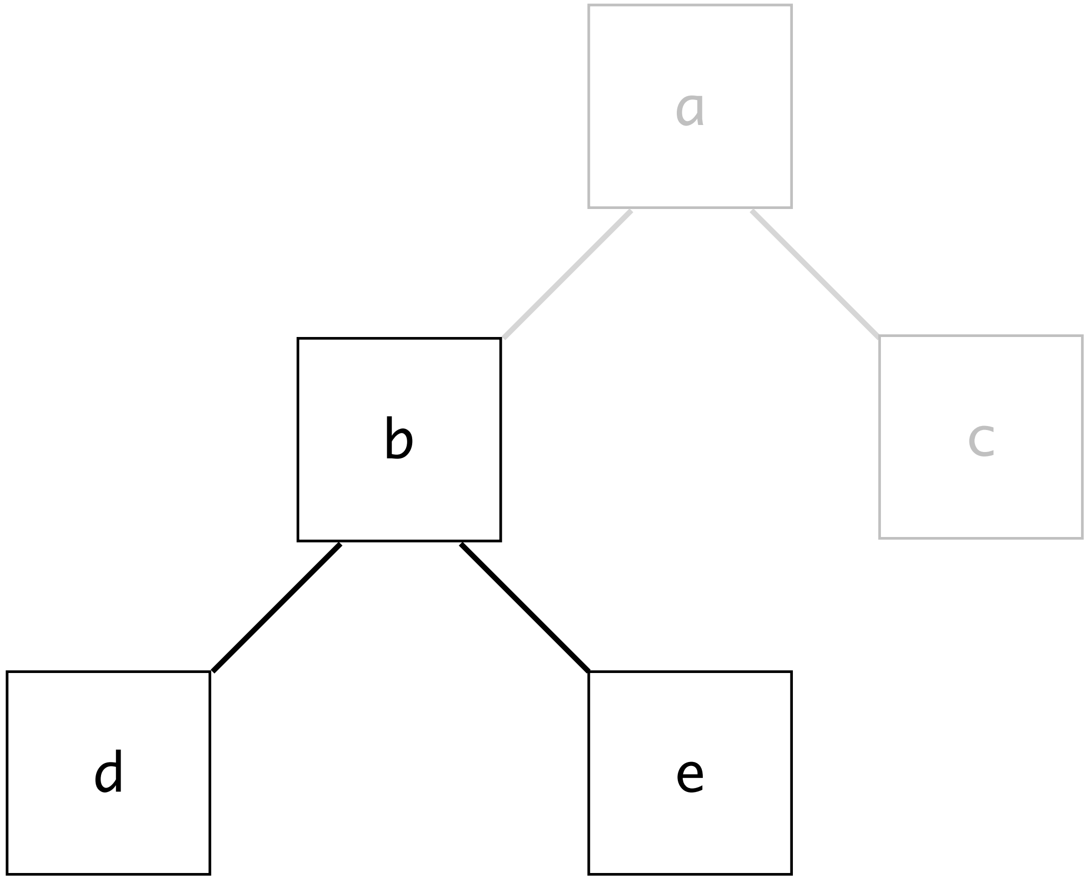
- the depth of the entire tree is the maximum depth of any of its leaves (leaf nodes)
- the depth of the following tree is 2, since the leaf nodes *d* and e have a depth of 2—the most of any leaf node



Trees

Subtrees:

- any non-root node in a tree can be viewed as the root of its own, smaller tree
- such a tree consists of that node and all its descendants
- in the diagram below, a subtree consisting of node *b* and its descendants is highlighed



Terminology

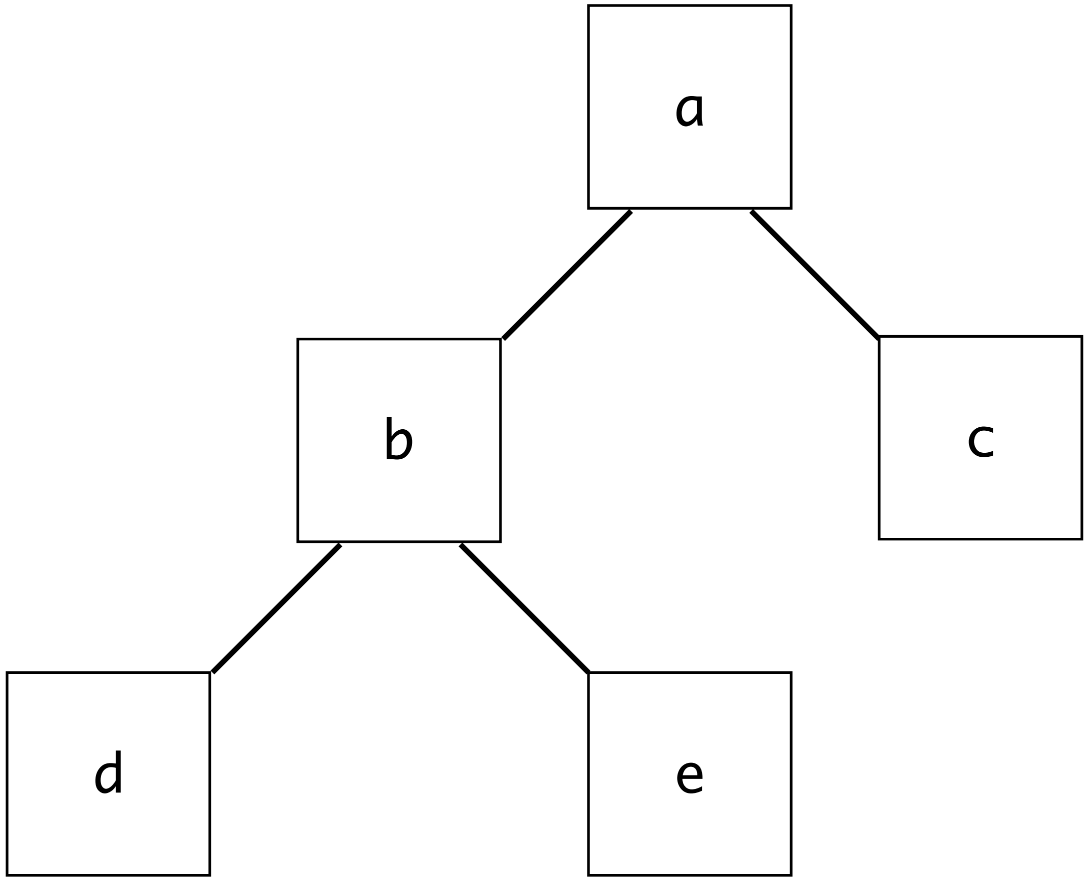
More terminology!

- binary trees
- left / right child
- full binary trees
- complete binary trees

Binary Trees

A binary tree is a specific type of tree

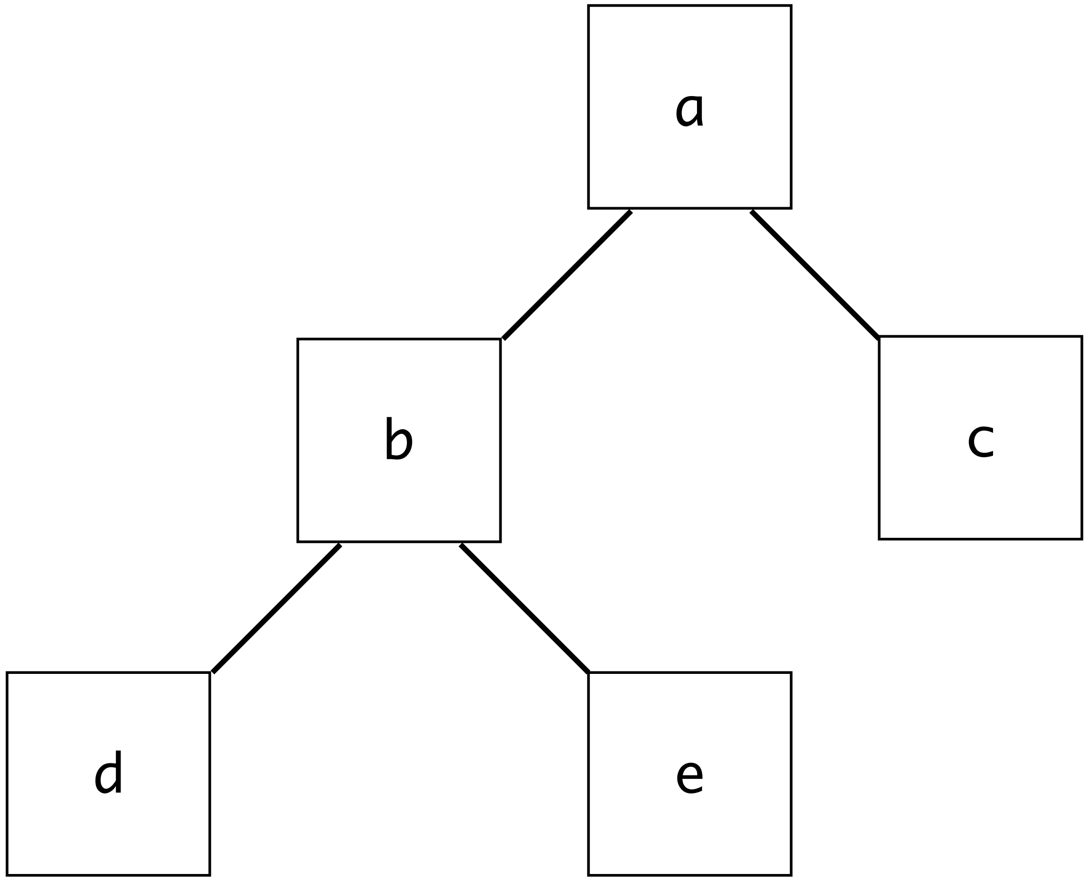
- it is a tree where any given node can have AT MOST two children
- the tree we've seen so far is a binary tree
- there are definitely other types of trees, but we'll focus on binary trees



Binary Trees

Left and right child nodes:

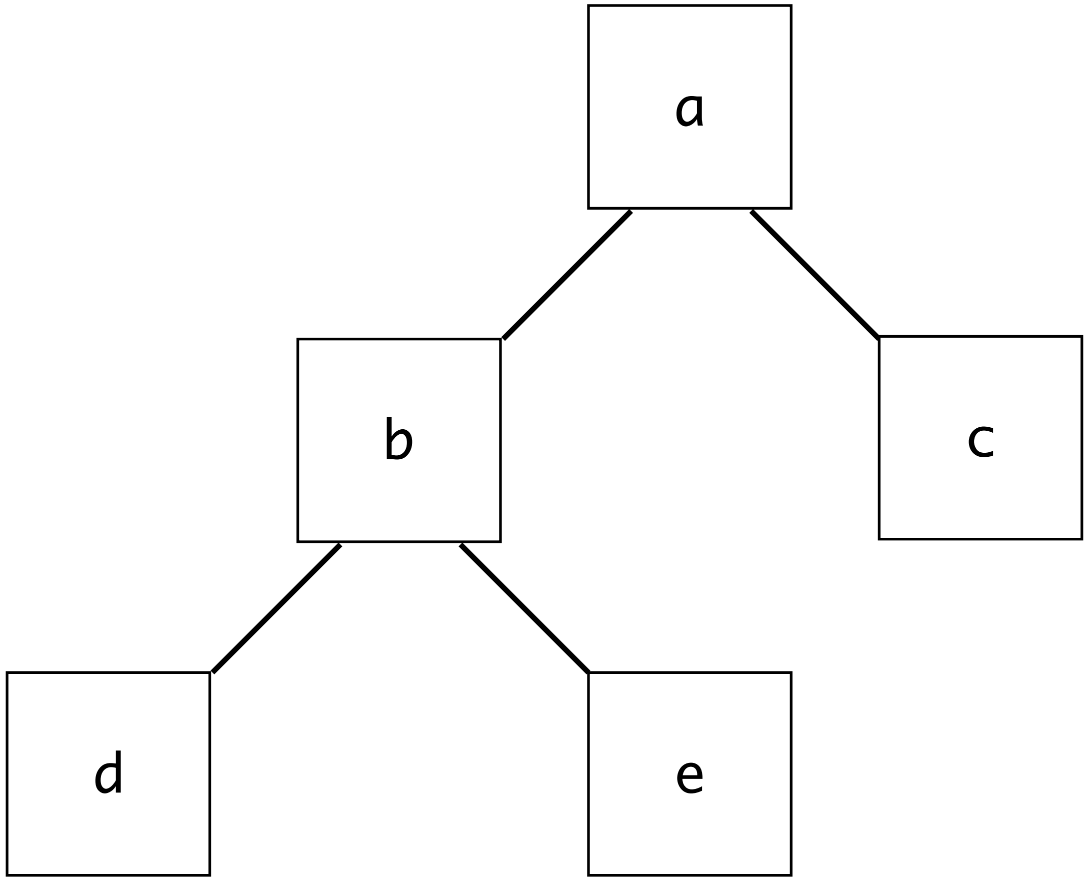
- the child node on the LEFT is the RIGHT CHILD, and vice-versa!!!
- not really... it's just like you think (node *b* is the left child of *a*, while node c is the right)
- how boring...



Binary Trees

A full binary tree:

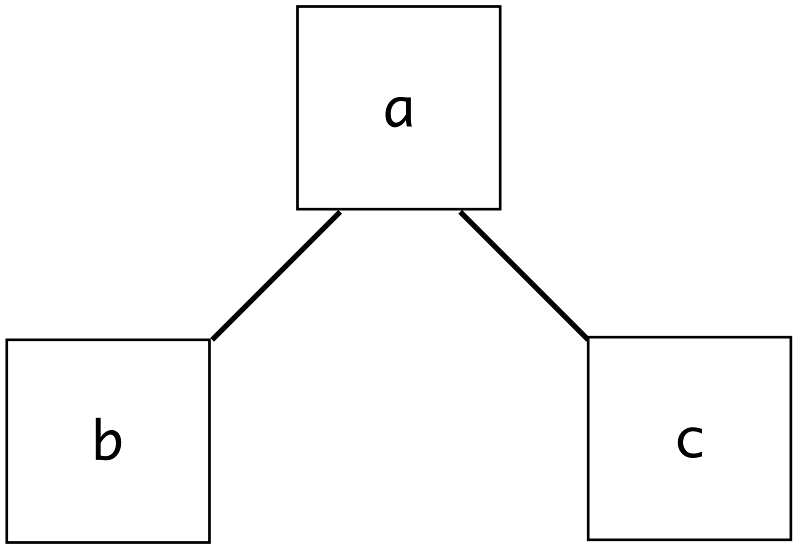
- a binary tree in which every leaf node has the same depth
- every non-leaf node must have exactly two children
- our trusty tree diagram is NOT a full binary tree



Binary Trees

A full binary tree:

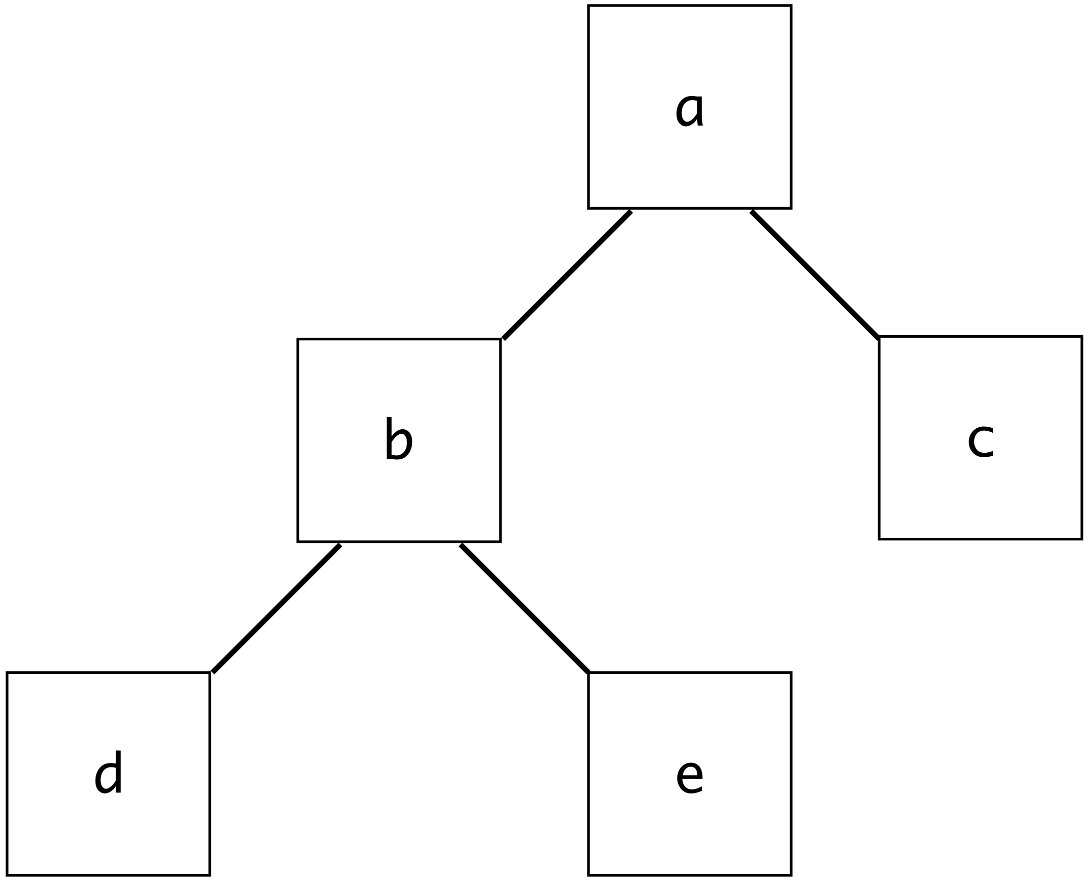
- a binary tree in which every leaf node has the same depth
- this means that every non-leaf node has exactly two children
- however, this one is!



Binary Trees

A complete binary tree:

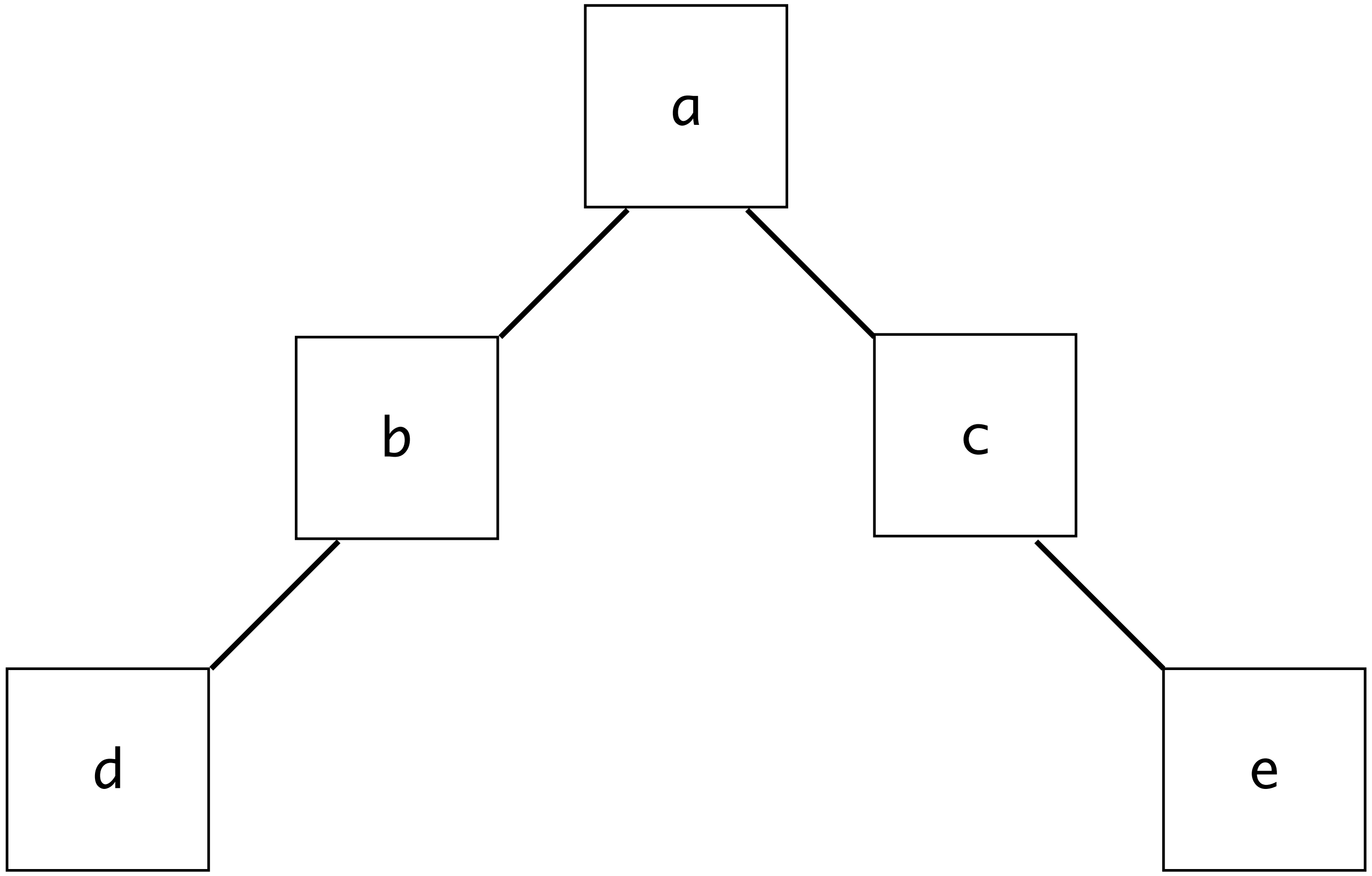
- all depths of the tree but the last are completely filled (nodes have two children each)
- children are added to the last depth starting with the left-most possible location (all leaf nodes are as far left as possible)
- our trusty tree diagram is a complete binary tree



Binary Trees

A complete binary tree:

- all depths of the tree but the last are completely filled (nodes have two children each)
- children are added to the last depth starting with the left-most possible location (all leaf nodes are as far left as possible)
- however, this one is not (node e is not as far left as it can be)



General Trees

General trees can have any number of child nodes per node...

- binary trees are just the most common you'll probably see
- all the other rules and terminology still apply