Discrete Mathematics and Its Applications

Kenneth H. Rosen

Chapter 9

Trees

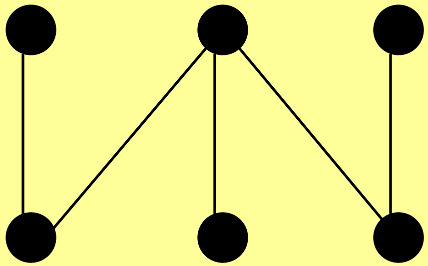
Tree

Definition 1. **A tree is a connected undirected graph with no simple circuits.**

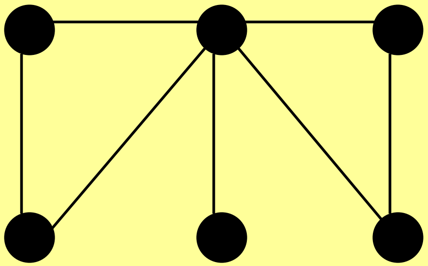
Theorem 1. **An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.**

Which graphs are trees?

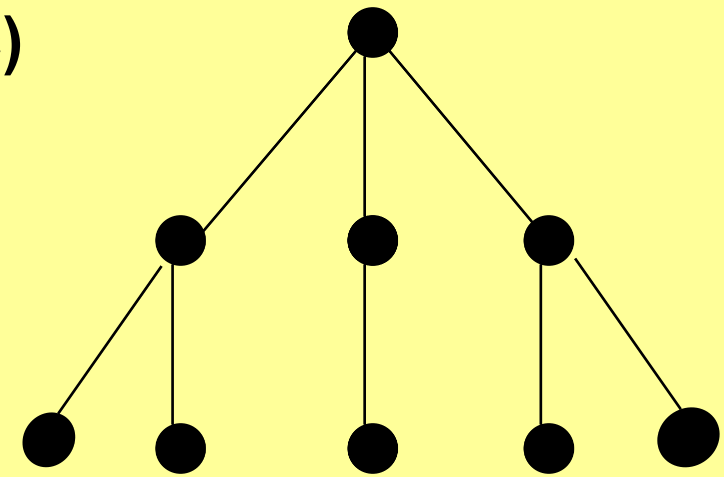
a)



b)

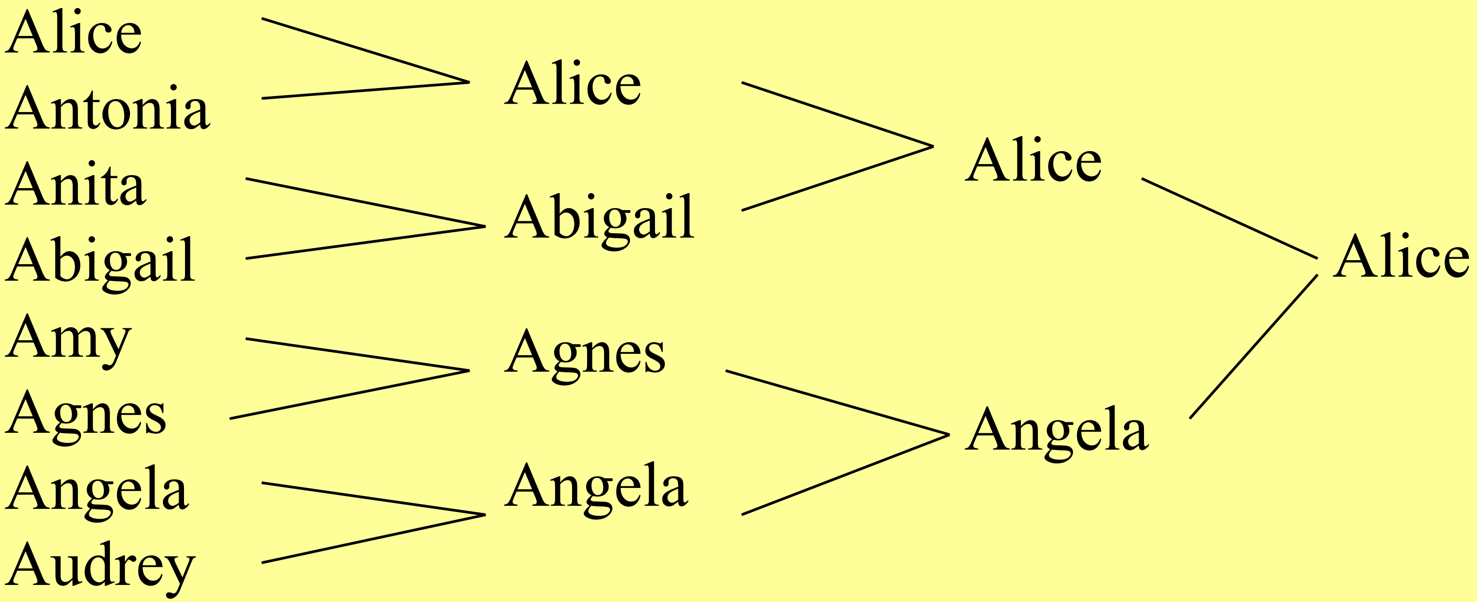


c)



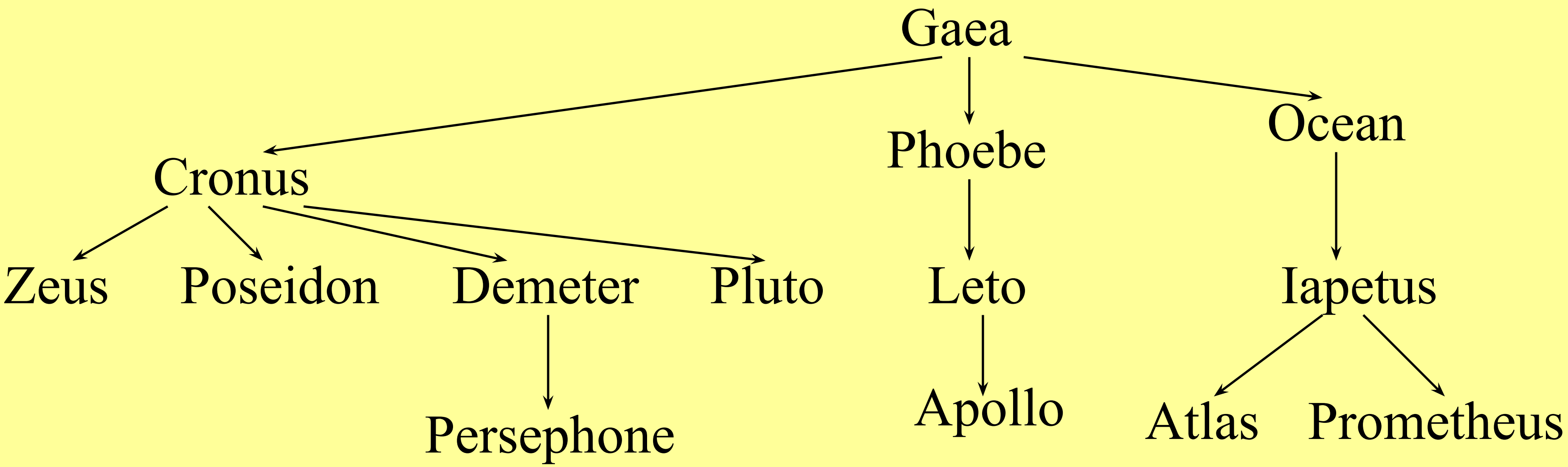
Tournament Trees

A common form of tree used in everyday life is the tournament tree, used to describe the outcome of a series of games, such as a tennis tournament.



A Family Tree

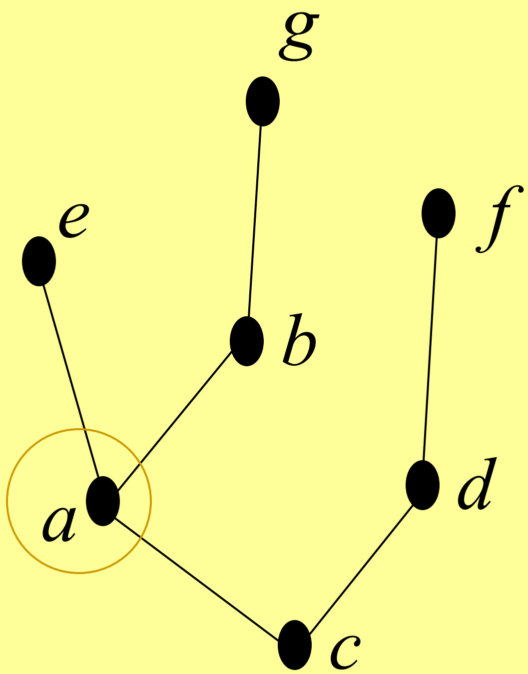
Much of the tree terminology derives from family trees.

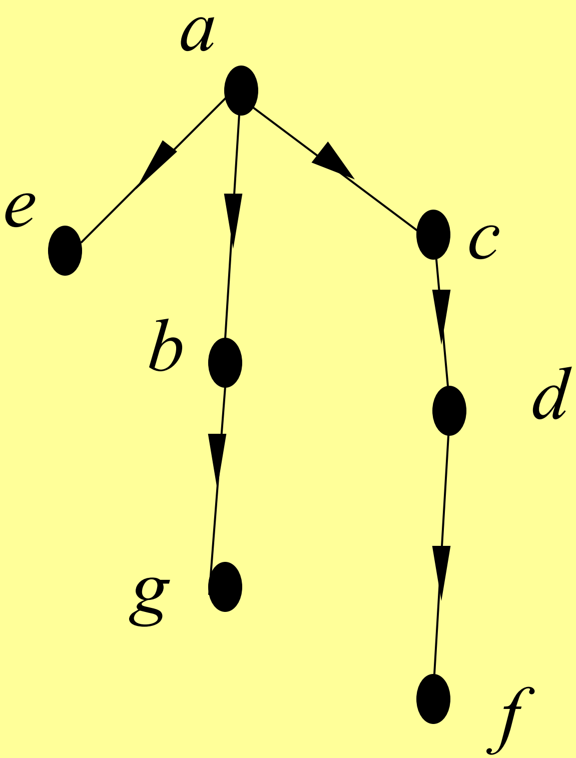


Rooted Trees

Once a vertex of a tree has been designated as the *root* of the tree, it is possible to assign direction to each of the edges.

Rooted Trees

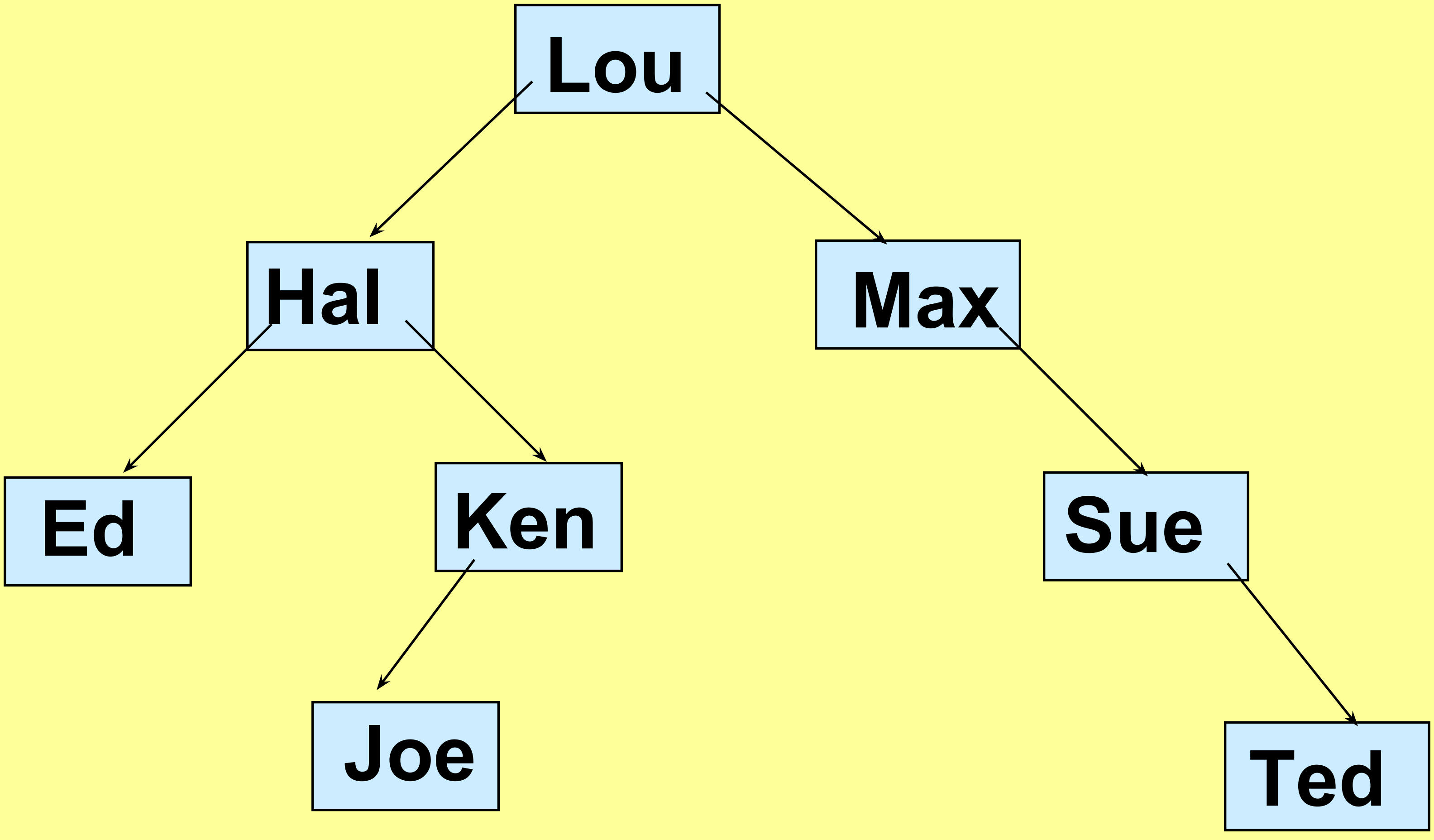




Parent

- **The parent of a non-root vertex v is the unique vertex u with a directed edge from u to v.**

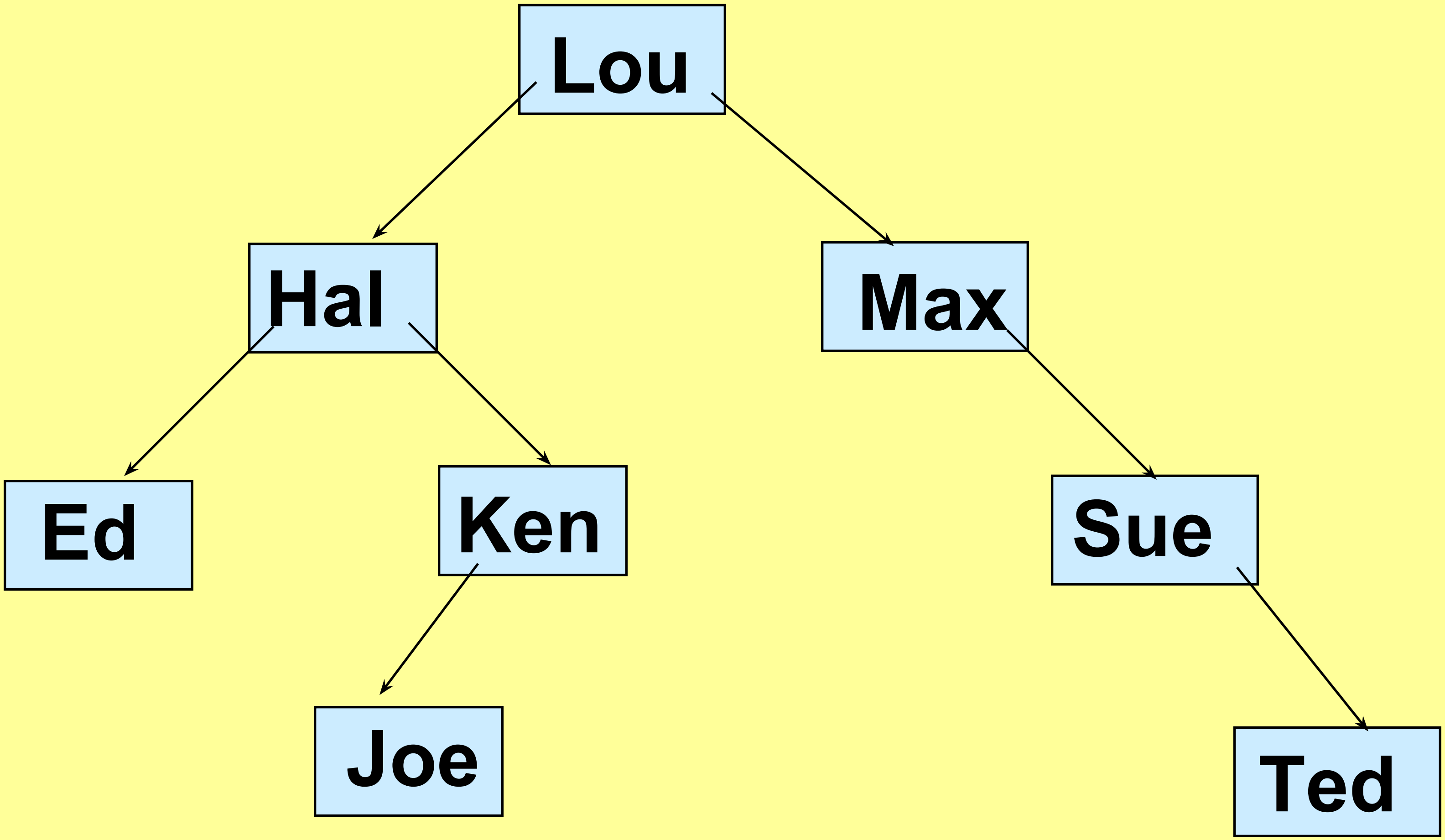
What is the parent of Ed?



Leaf

- **A vertex is called a leaf if it has no children.**

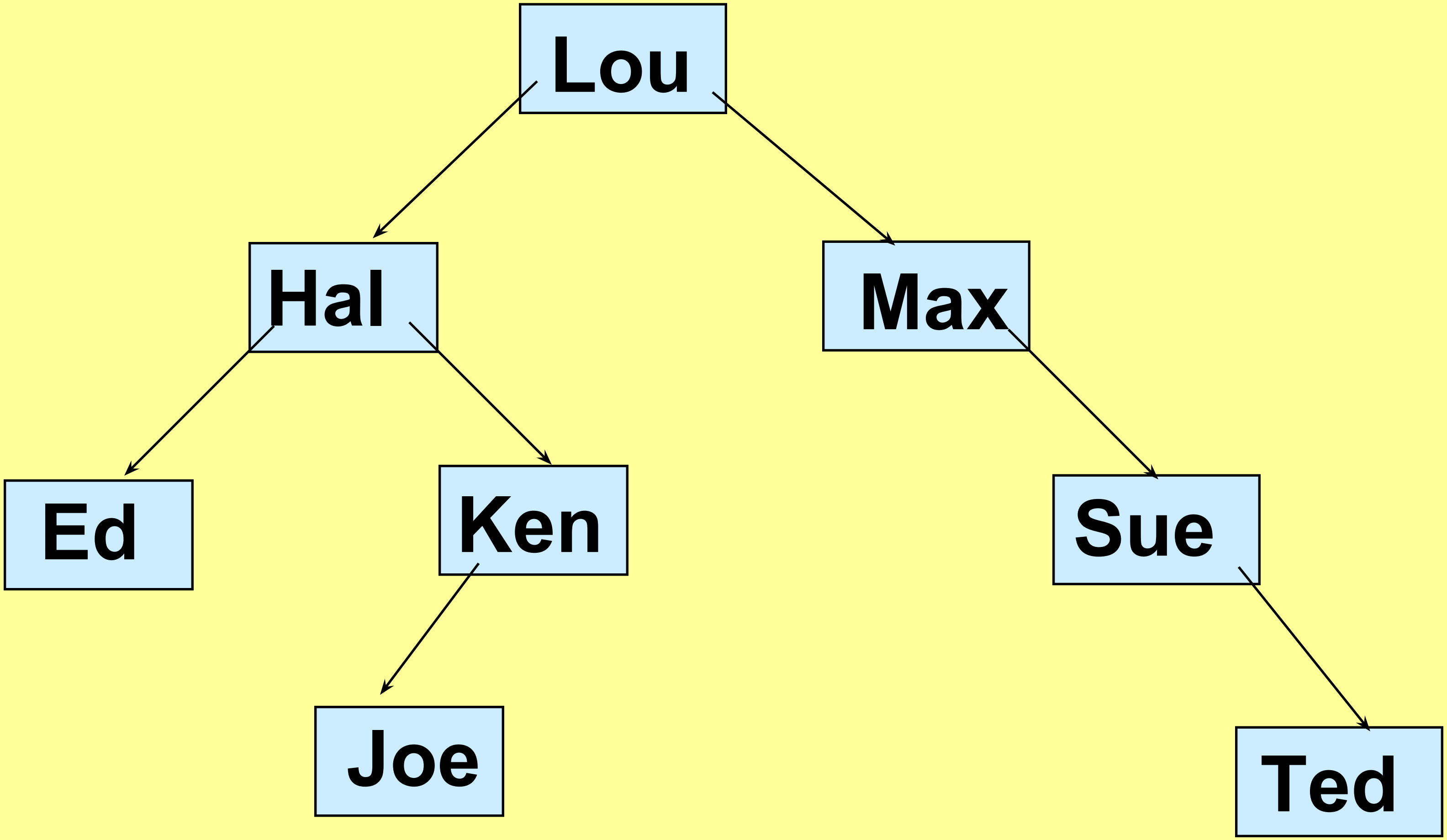
How many leaves?



Ancestors

- The **ancestors of a non-root vertex** are all the vertices in the path from root to this vertex.

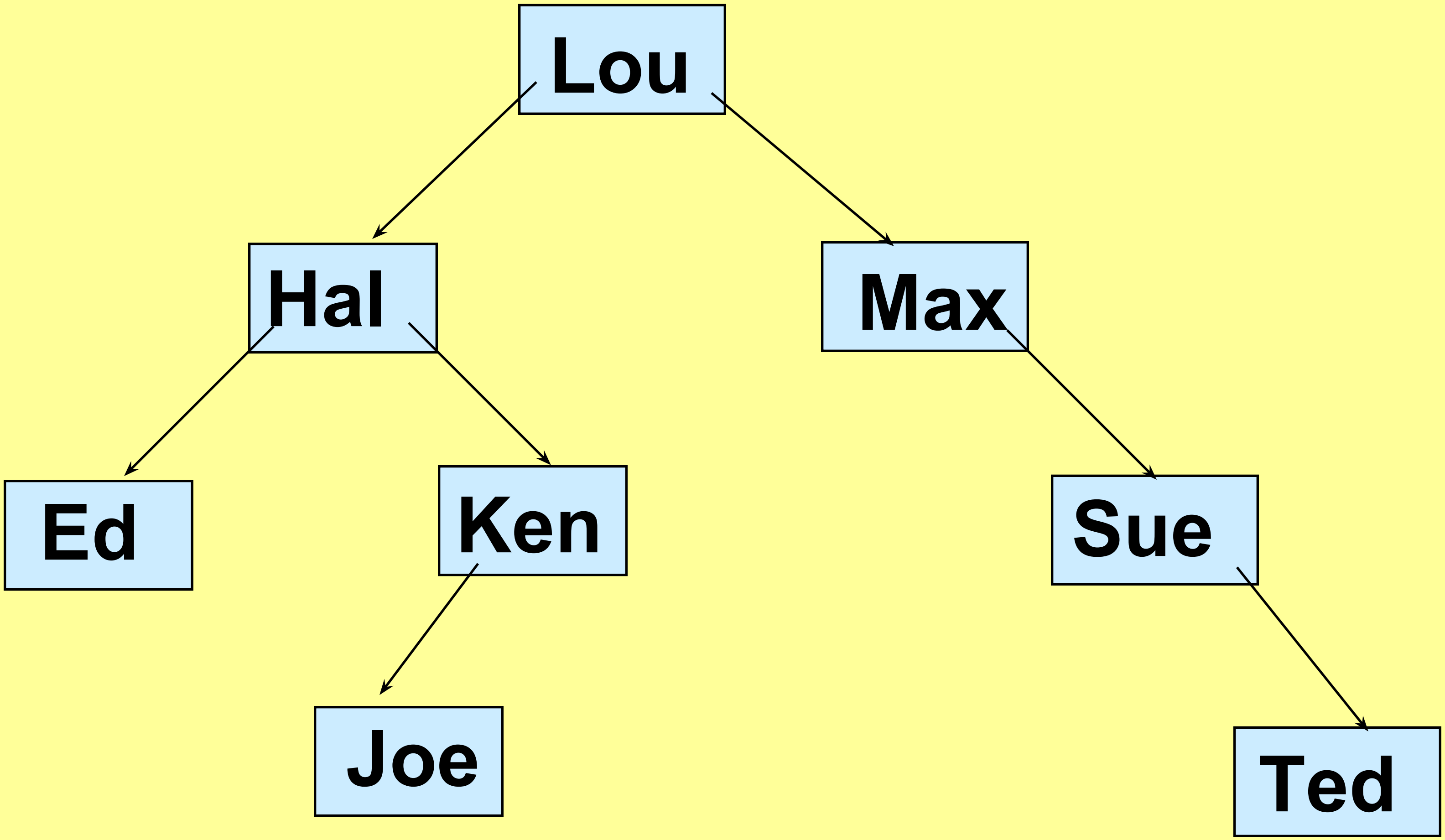
How many ancestors of Ken?



Descendants

- **The descendants of vertex v are all the vertices that have v as an ancestor.**

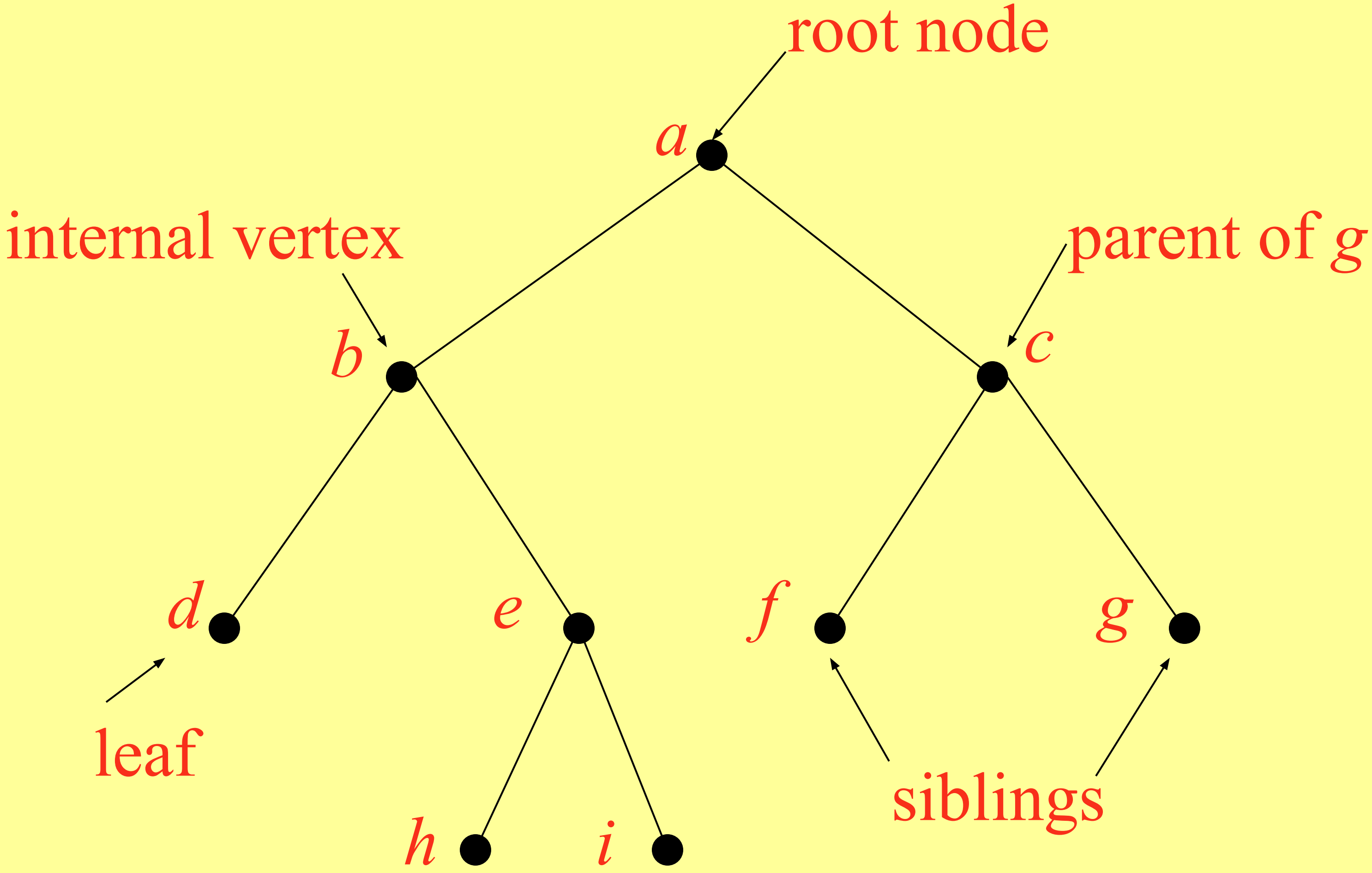
How many descendants of Hal?

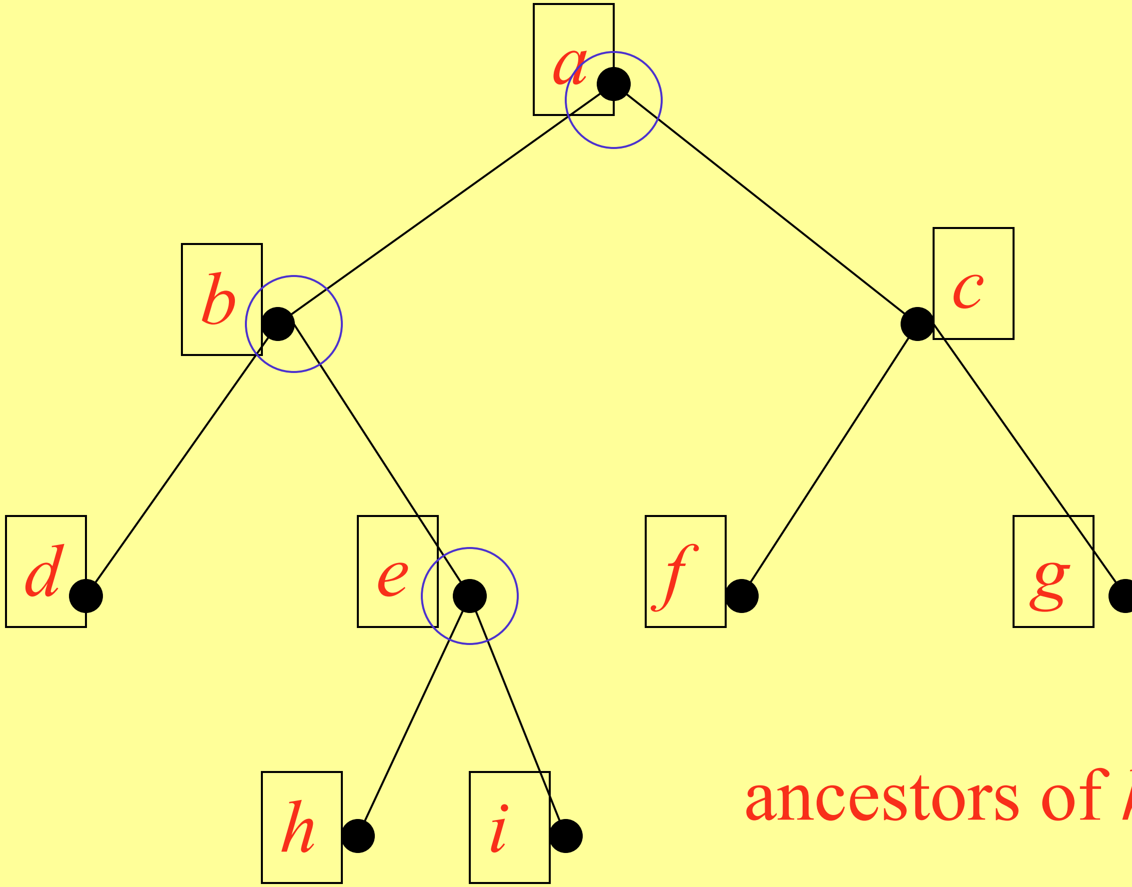


Internal Vertex

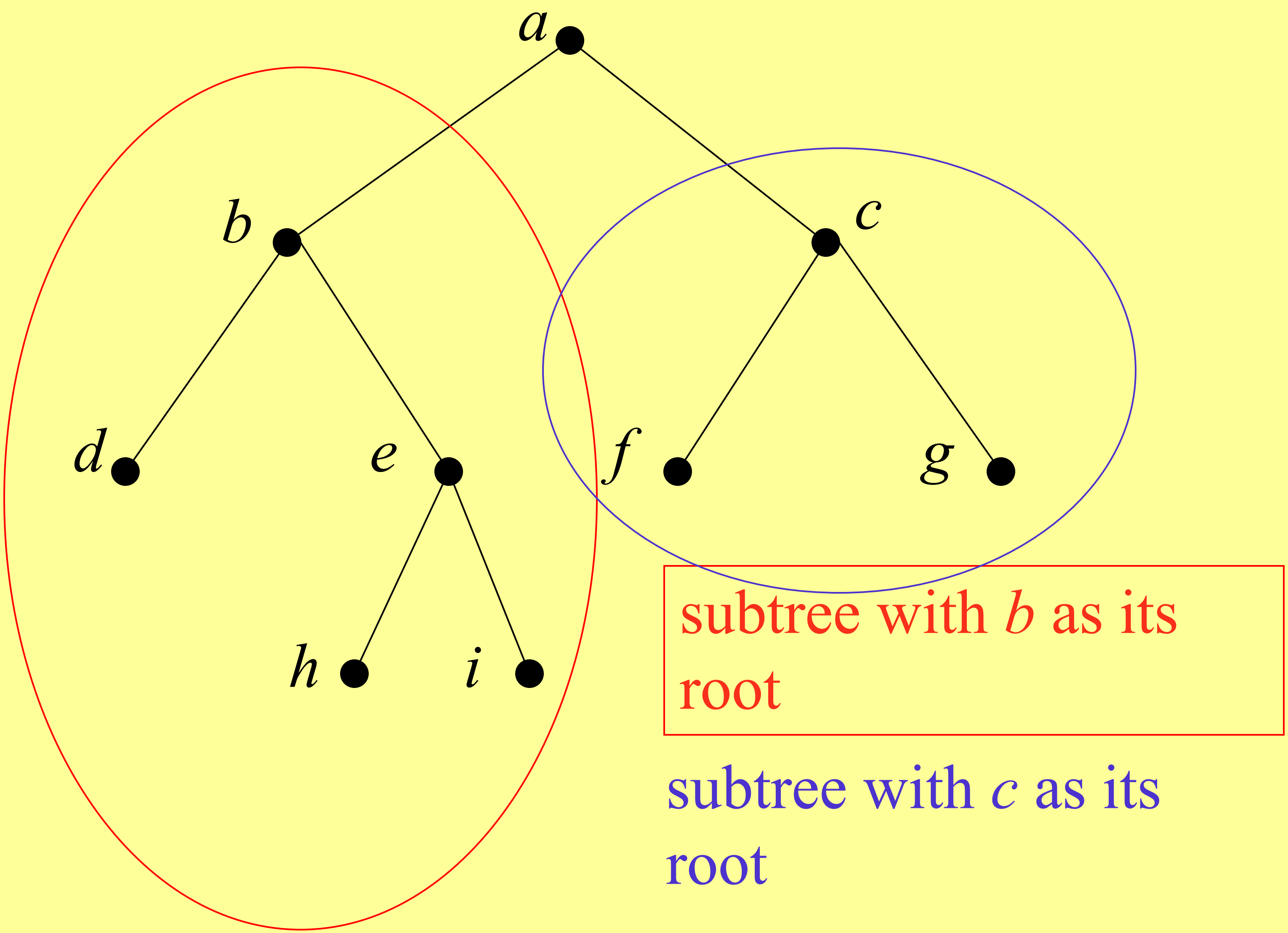
A vertex that has children is called an **internal vertex.**

The **subtree at vertex v** is the subgraph of the tree consisting of vertex v and its descendants and all edges incident to those descendants.

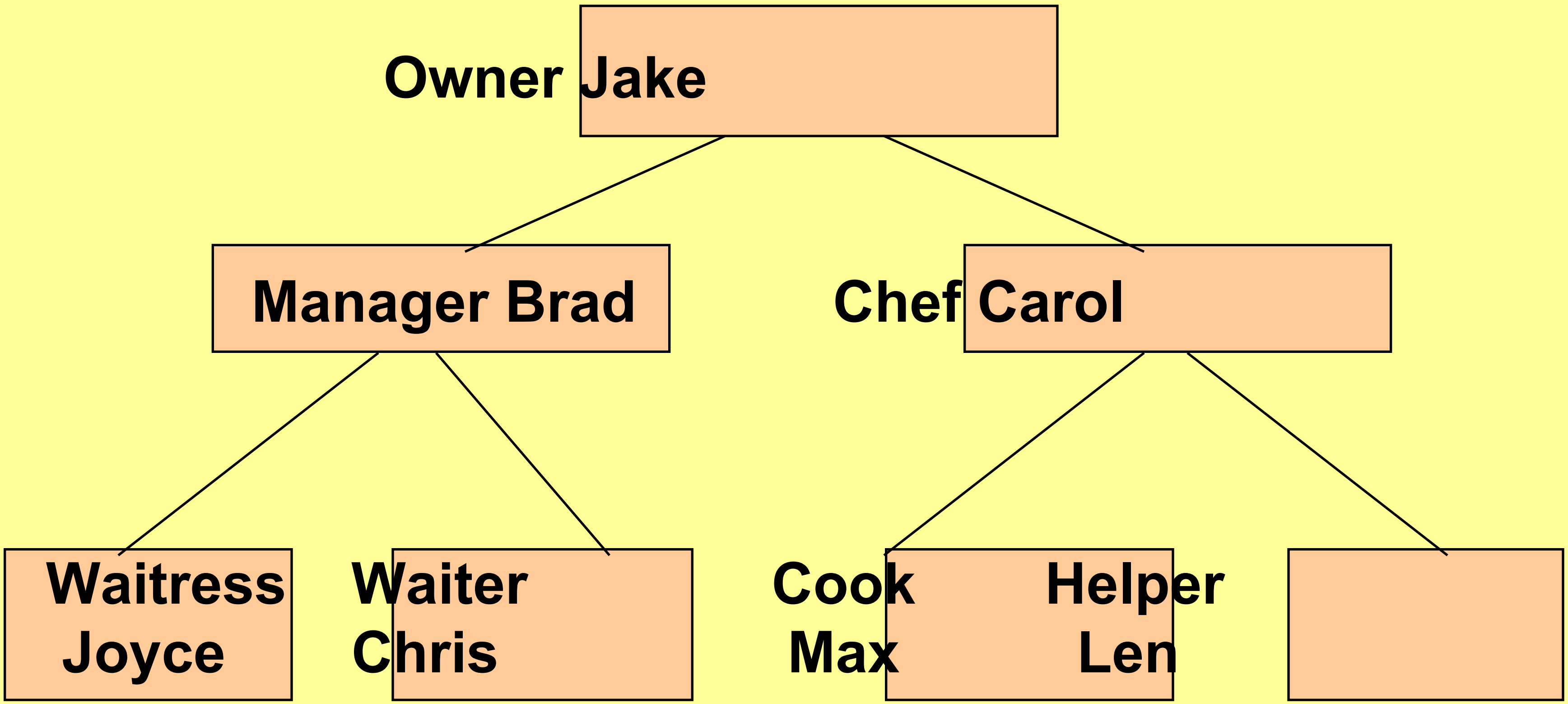

root node
a
internal vertex
parent of g
b
c
d
e
f
g
leaf
siblings
h
i



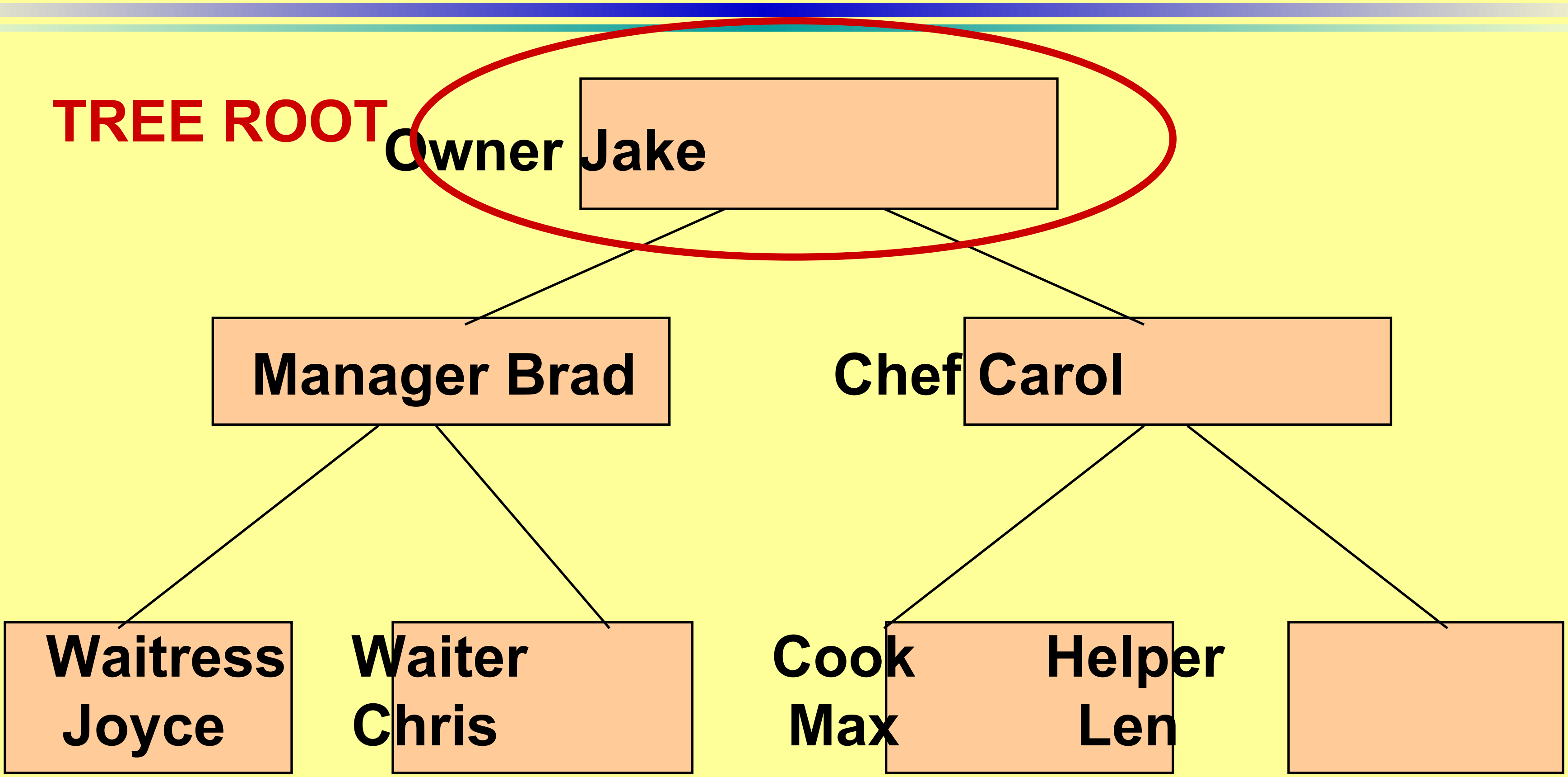
ancestors of *h* and *i*


a
b
c
d
e
f
g
h
i
subtree with b as its root
subtree with c as its root

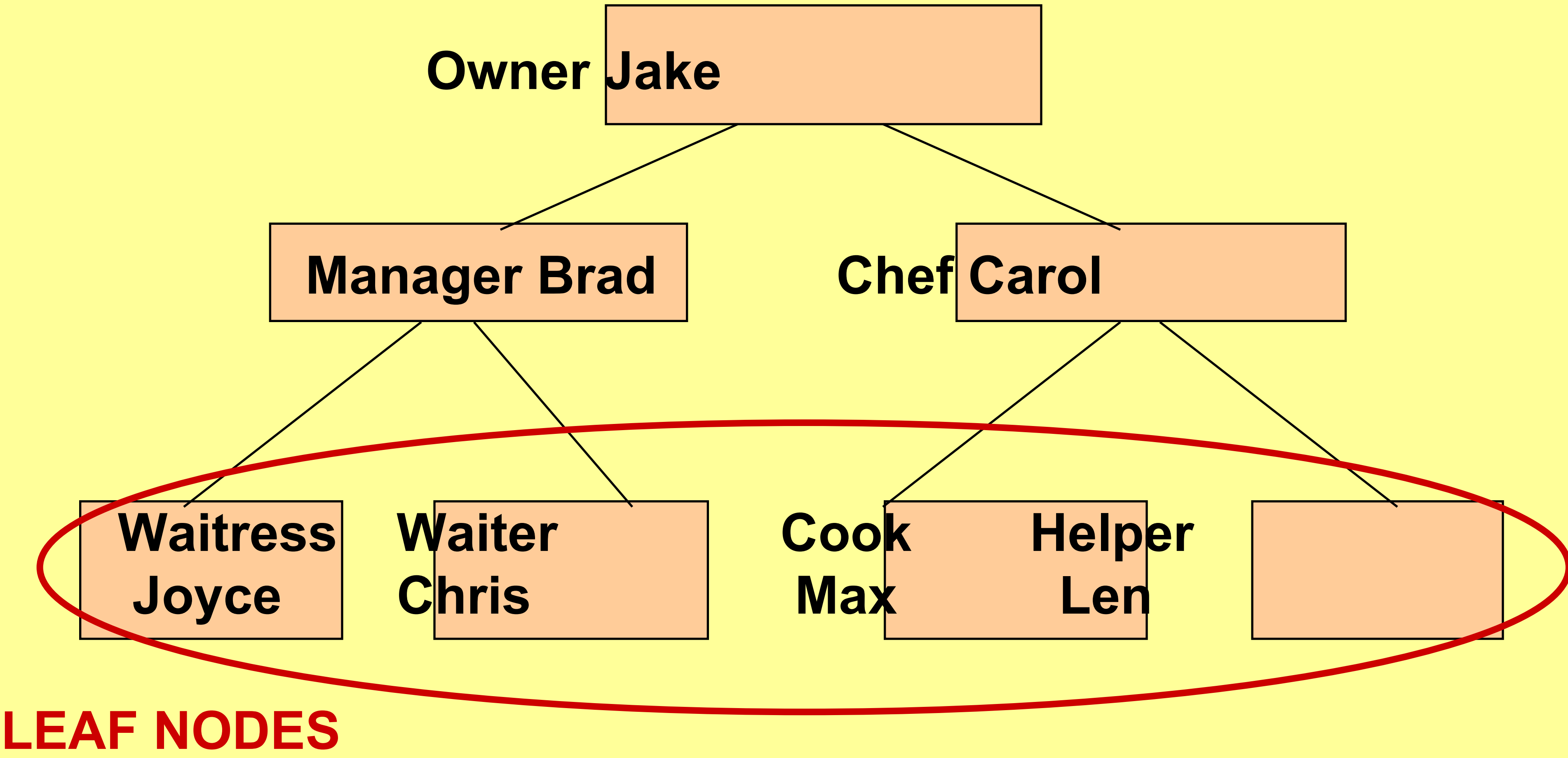
Jake's Pizza Shop Tree



A Tree Has a Root

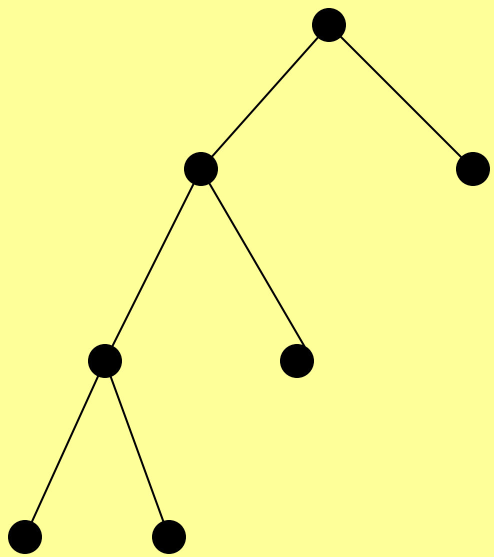


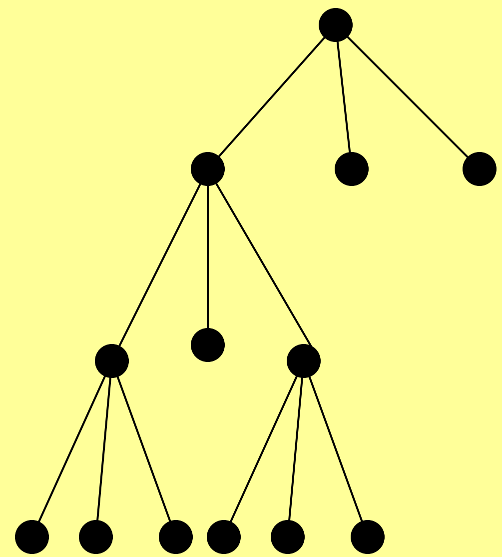
Leaf nodes have no children

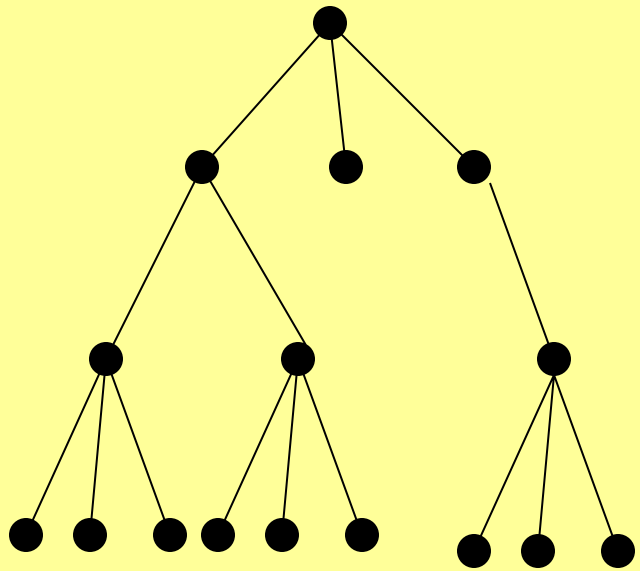


***m*-ary trees**

A rooted tree is called an *m-ary tree* if every internal vertex has no more than *m* children. The tree is called a *full m-ary tree* if every internal vertex has exactly *m* children. An *m*-ary tree with *m*=2 is called a *binary tree*.





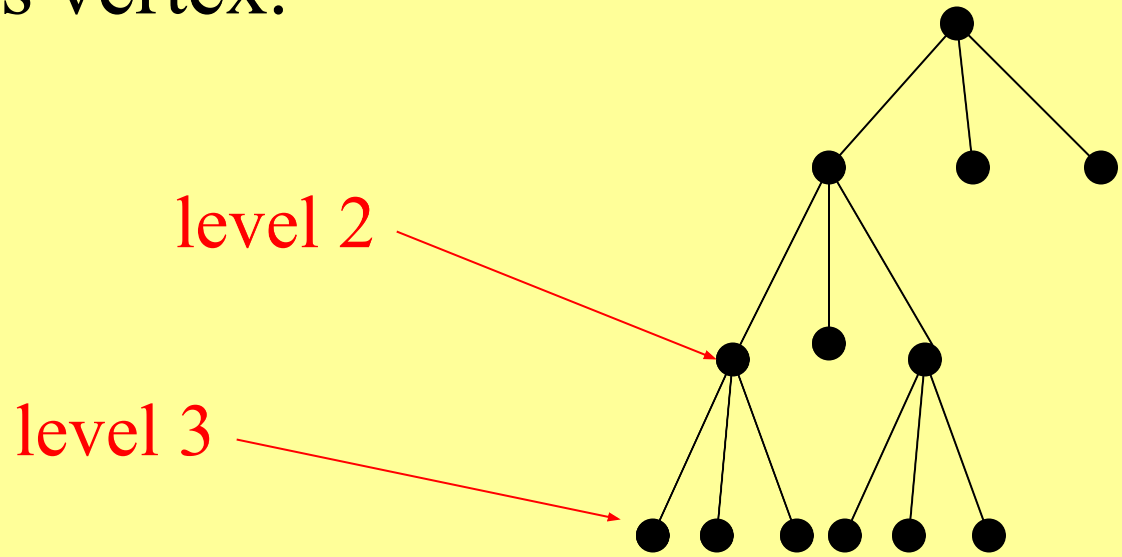


Level of tree

- The level of a vertex in a rooted tree is the length of the unique path from the root to this vertex.
- The level of the root is defined to be zero.
- The **height** of the rooted tree is the maximum of the levels of vertices.

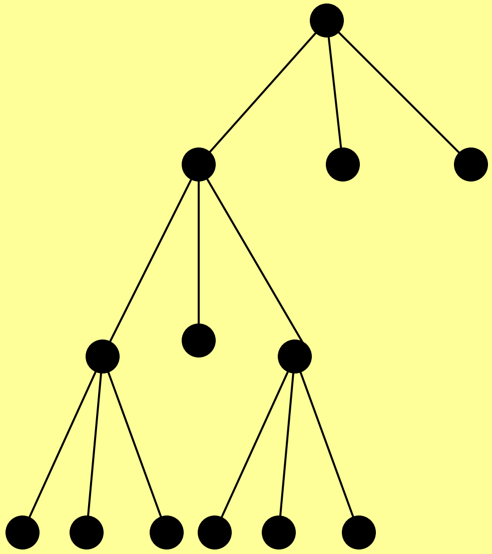
Properties of Trees

The level of a vertex *v* in a rooted tree is the length of the unique path from the root to this vertex.



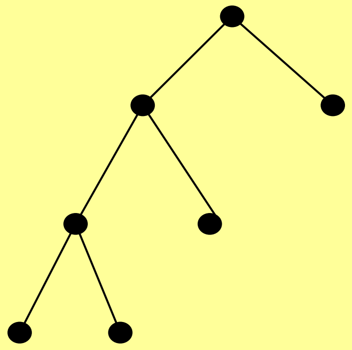
Properties of Trees

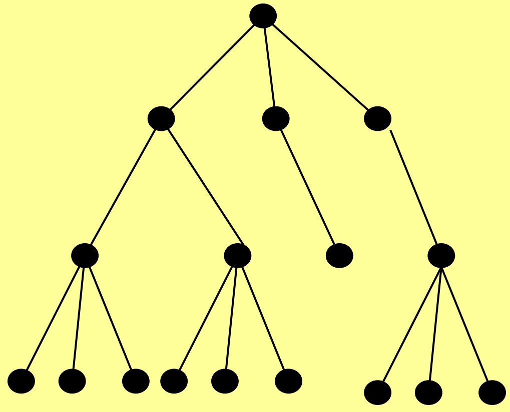
The *height* of a rooted tree is the maximum of the levels of vertices.

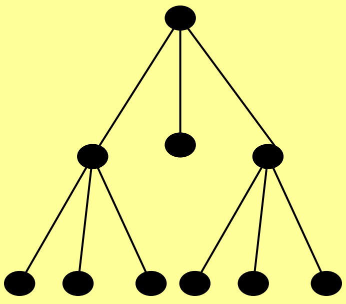


Properties of Trees

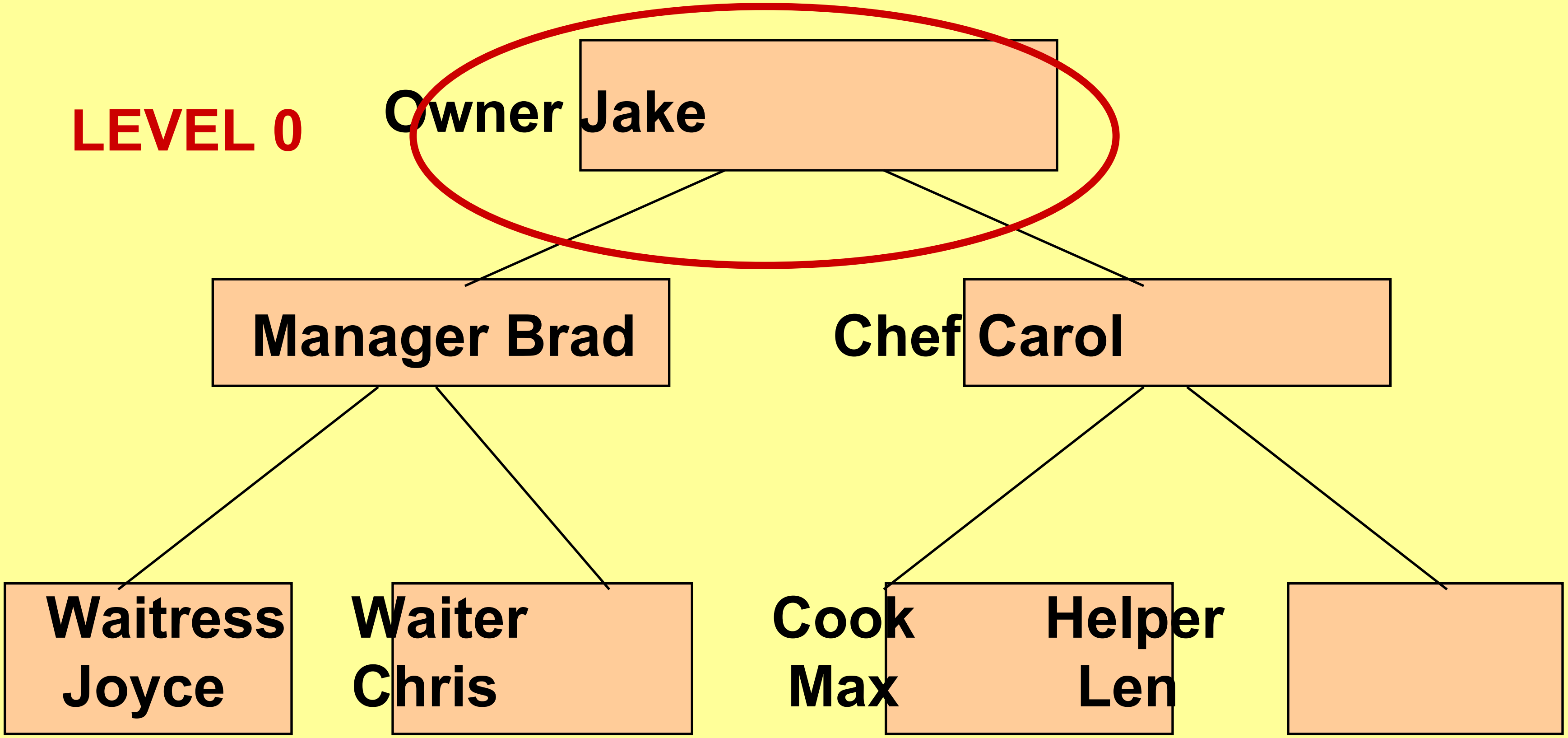
A rooted *m*-ary tree of height *h* is called *balanced* if all leaves are at levels *h* or *h*-1.



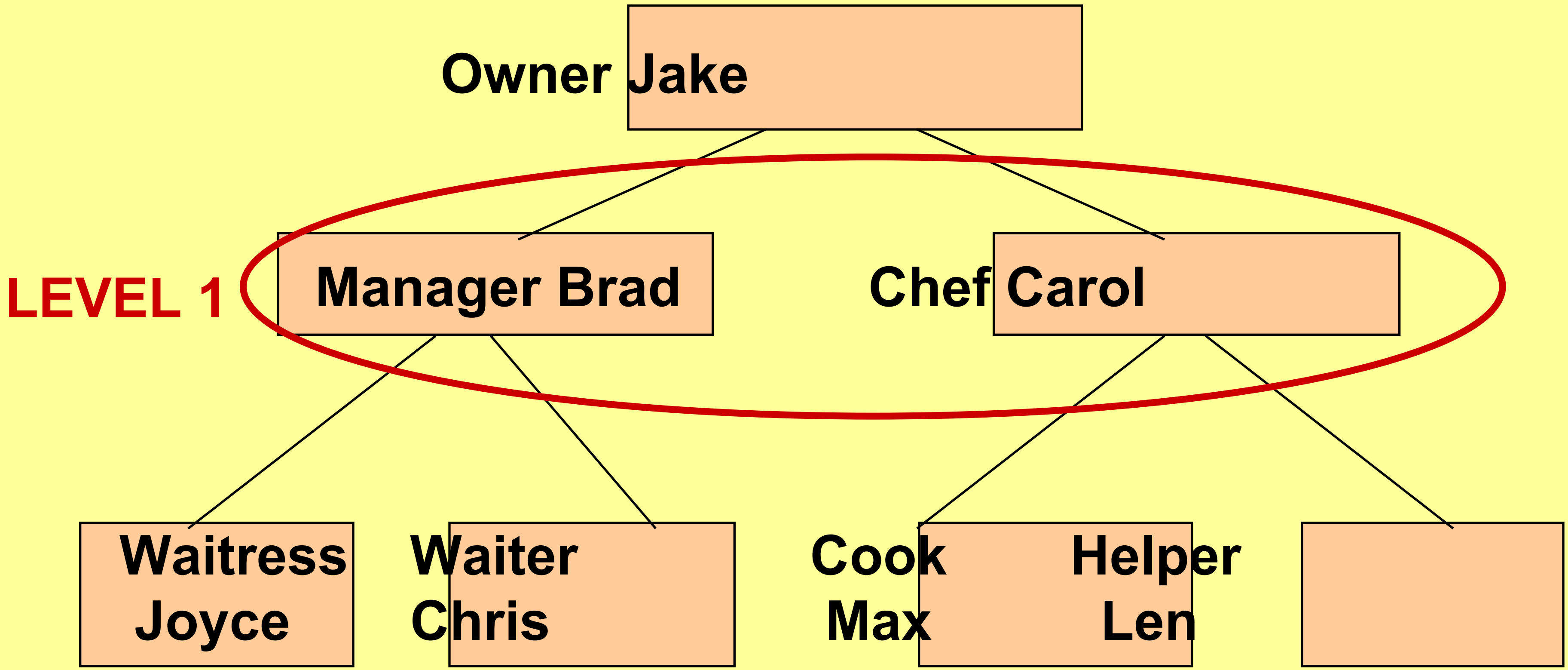




A Tree Has Levels

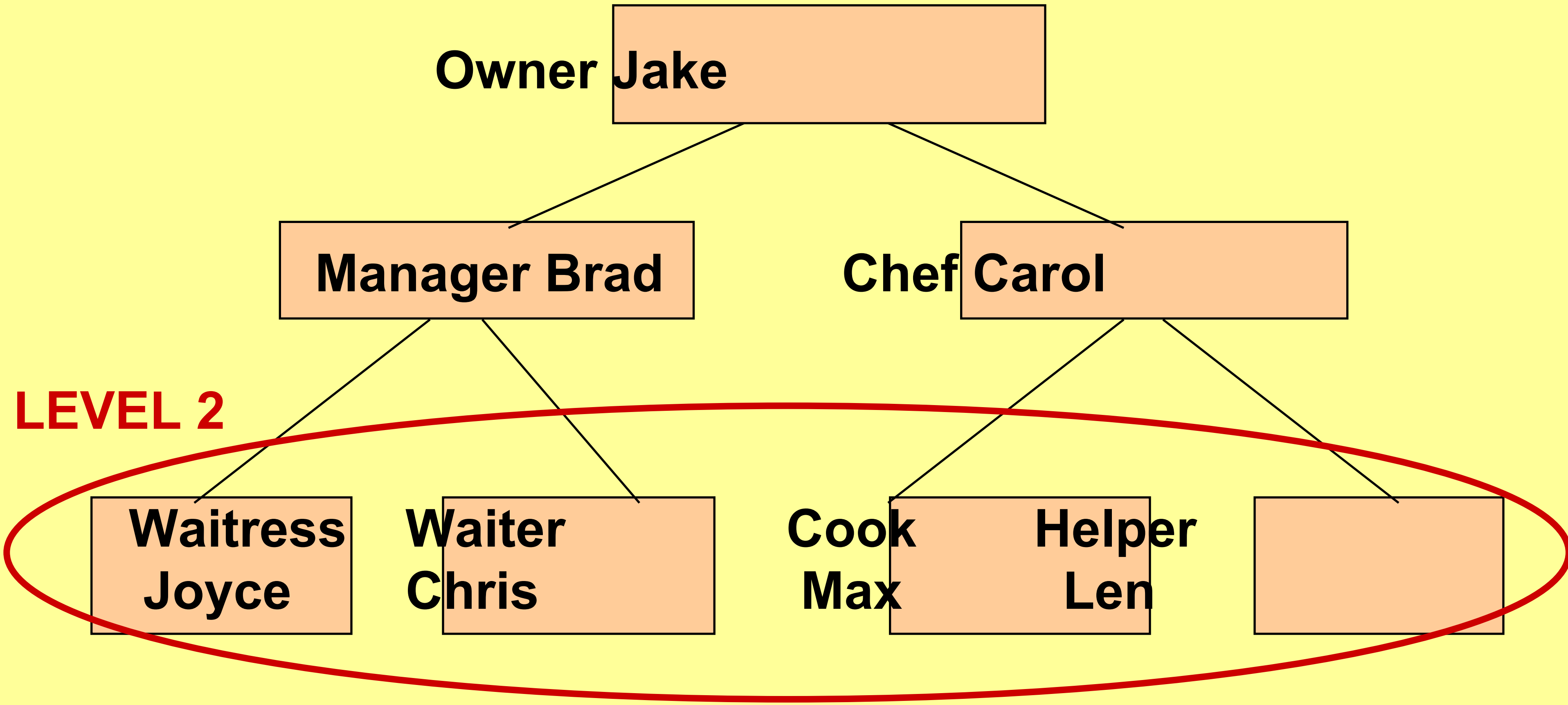


Level One



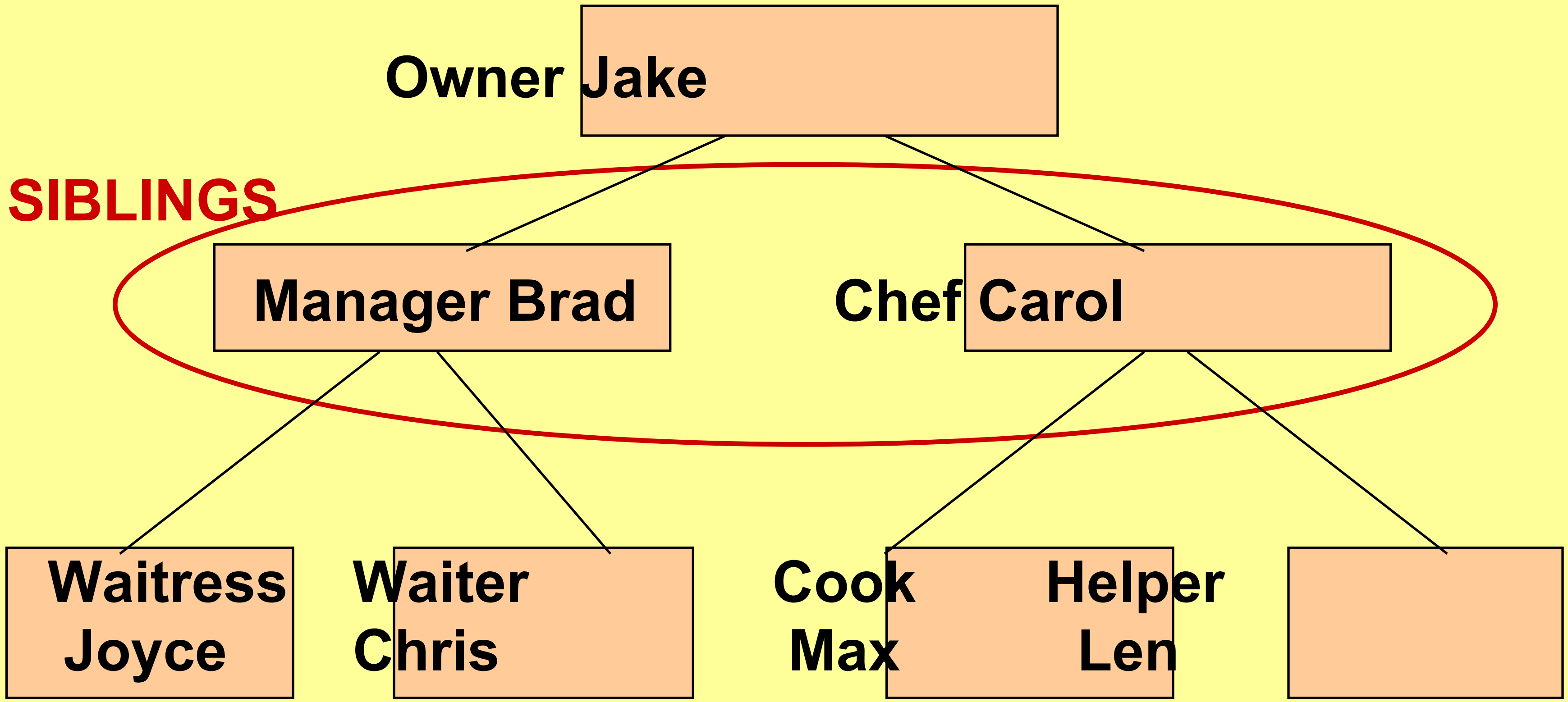
Owner Jake
LEVEL 1
Manager Brad
Chef Carol
Waitress Joyce
Waiter Chris
Cook Max
Helper Len

Level Two

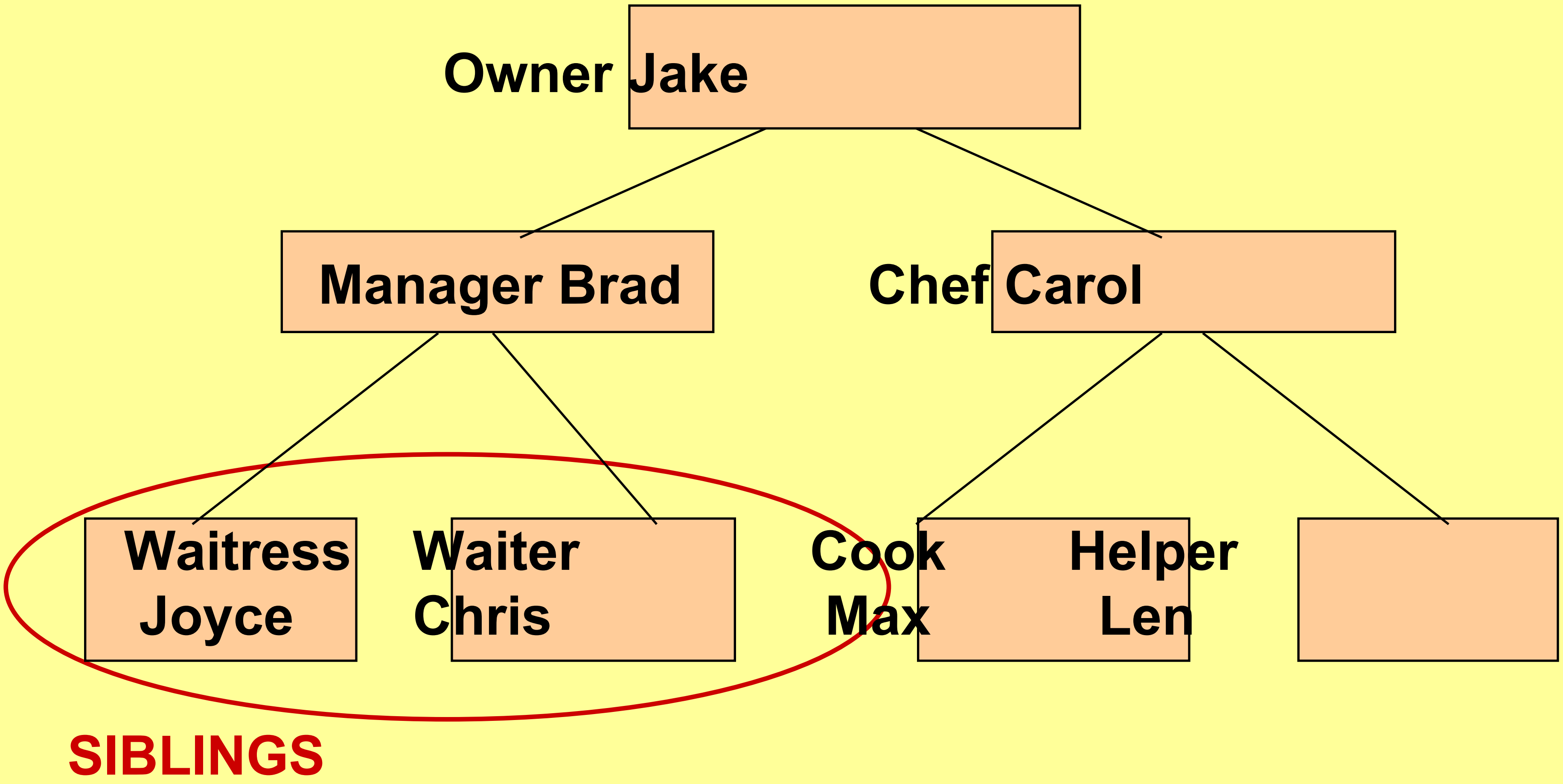


Owner Jake
Manager Brad
Chef Carol
LEVEL 2
Waitress Joyce
Waiter Chris
Cook Max
Helper Len

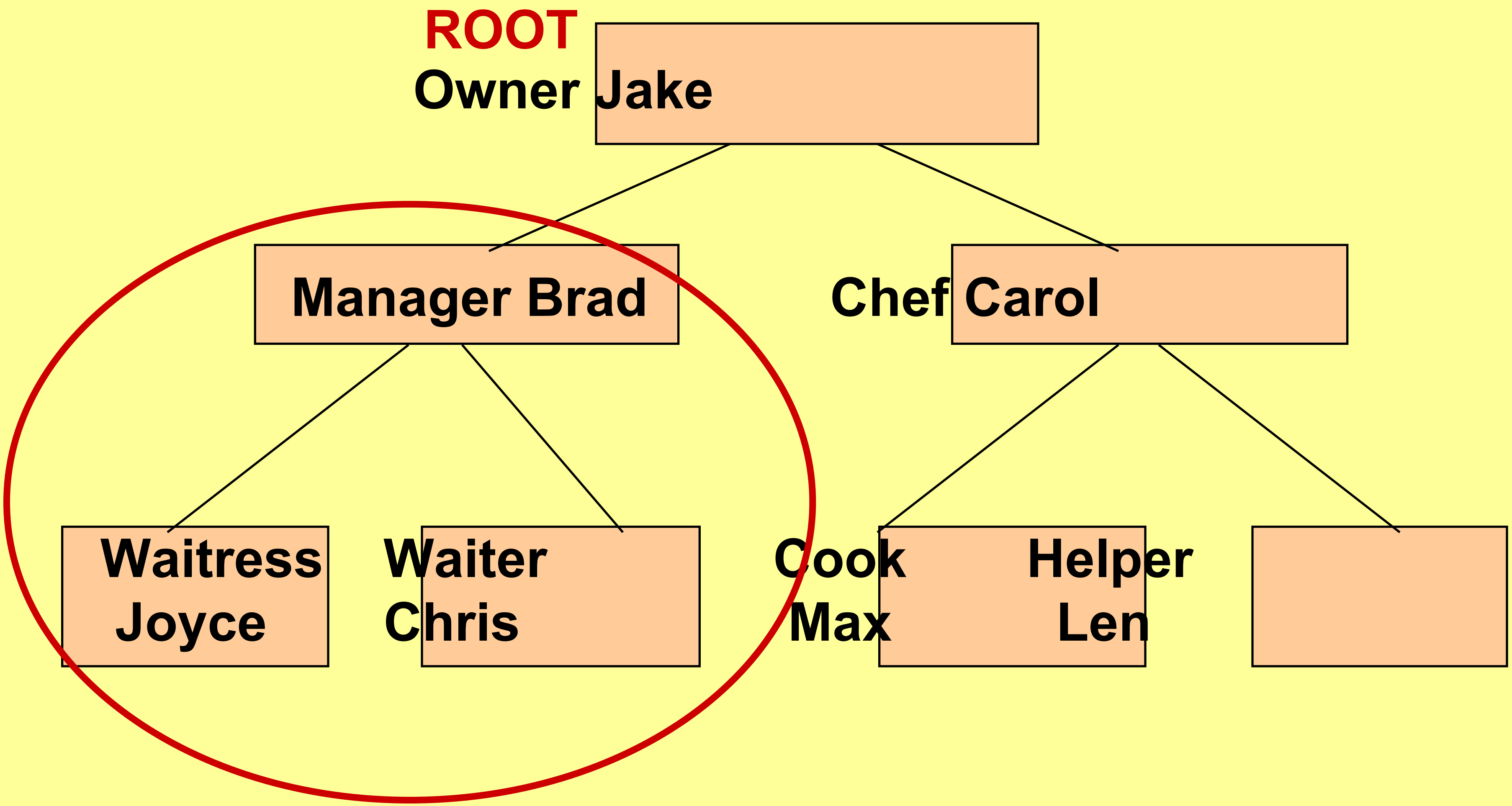
Sibling nodes have same parent



Sibling nodes have same parent

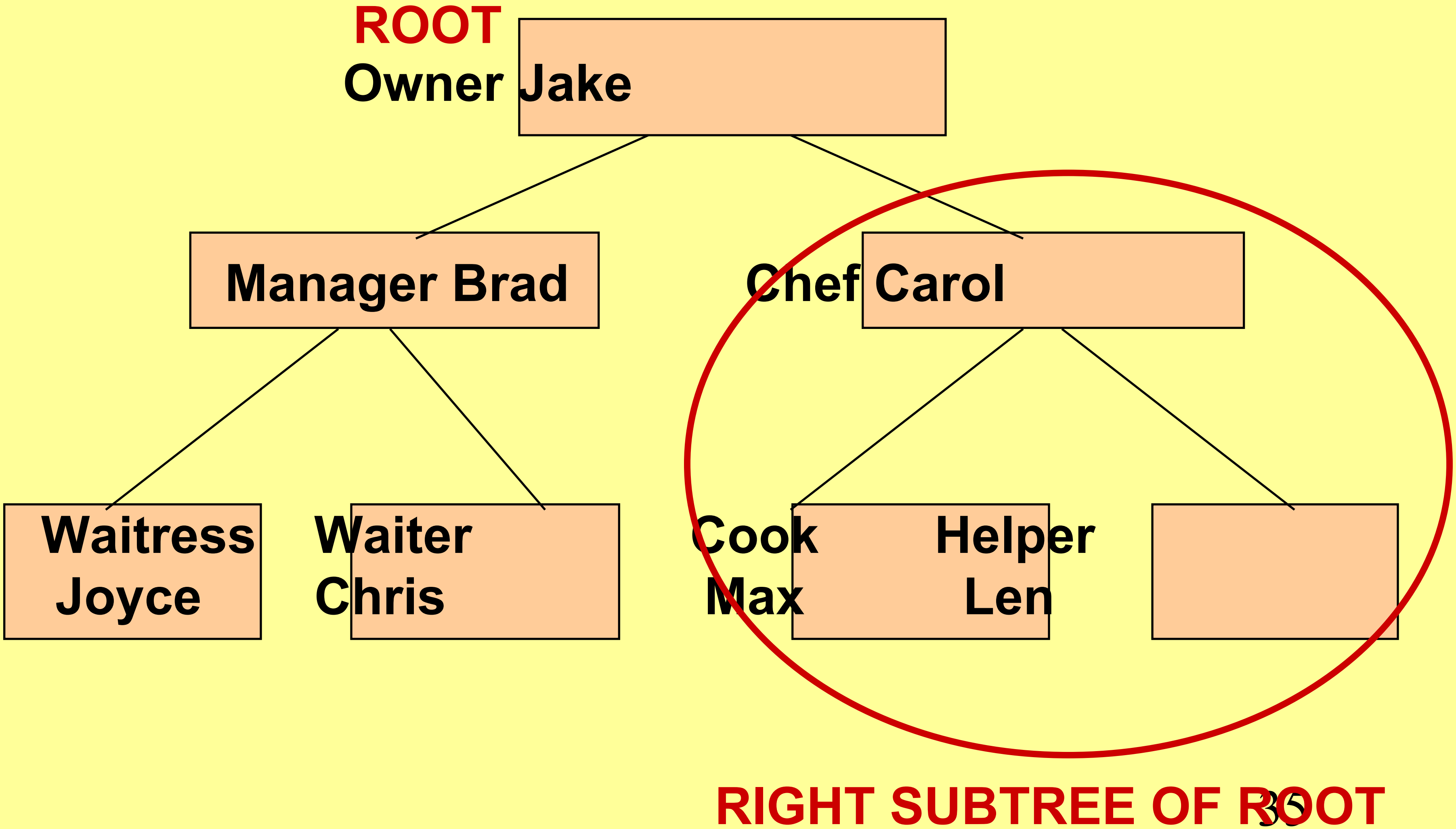


A Subtree



LEFT SUBTREE OF ROOT

Another Subtree



Binary Tree

Definition 2′. A rooted tree is called a **binary tree** if every internal vertex has no more than 2 children.

The tree is called a **full** binary tree if every internal vertex has exactly 2 children.

Ordered Rooted Tree

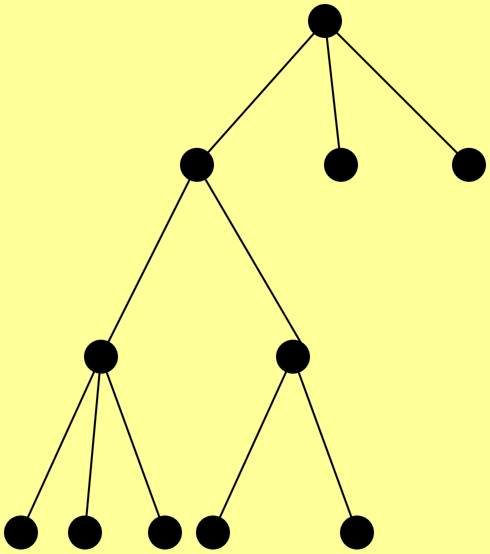
An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered. Ordered trees are drawn so that the children of each internal vertex are shown in order from left to right.

Tree Properties

Theorem 2. A tree with N vertices has N-1 edges.

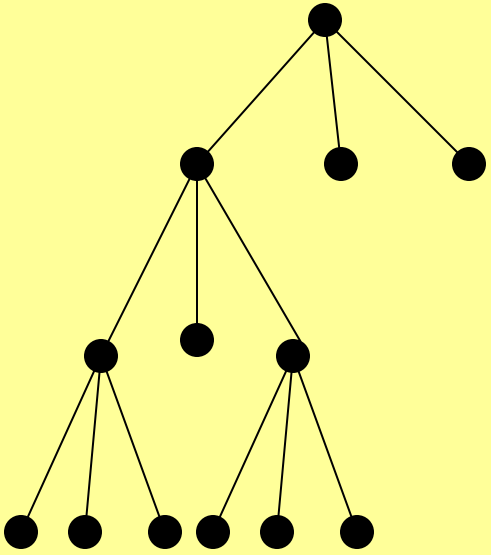
Properties of Trees

A tree with n vertices has n-1 edges.



Properties of Trees

A full m-ary tree with i internal vertices contains $n = mi+1$ vertices.



Properties of Trees

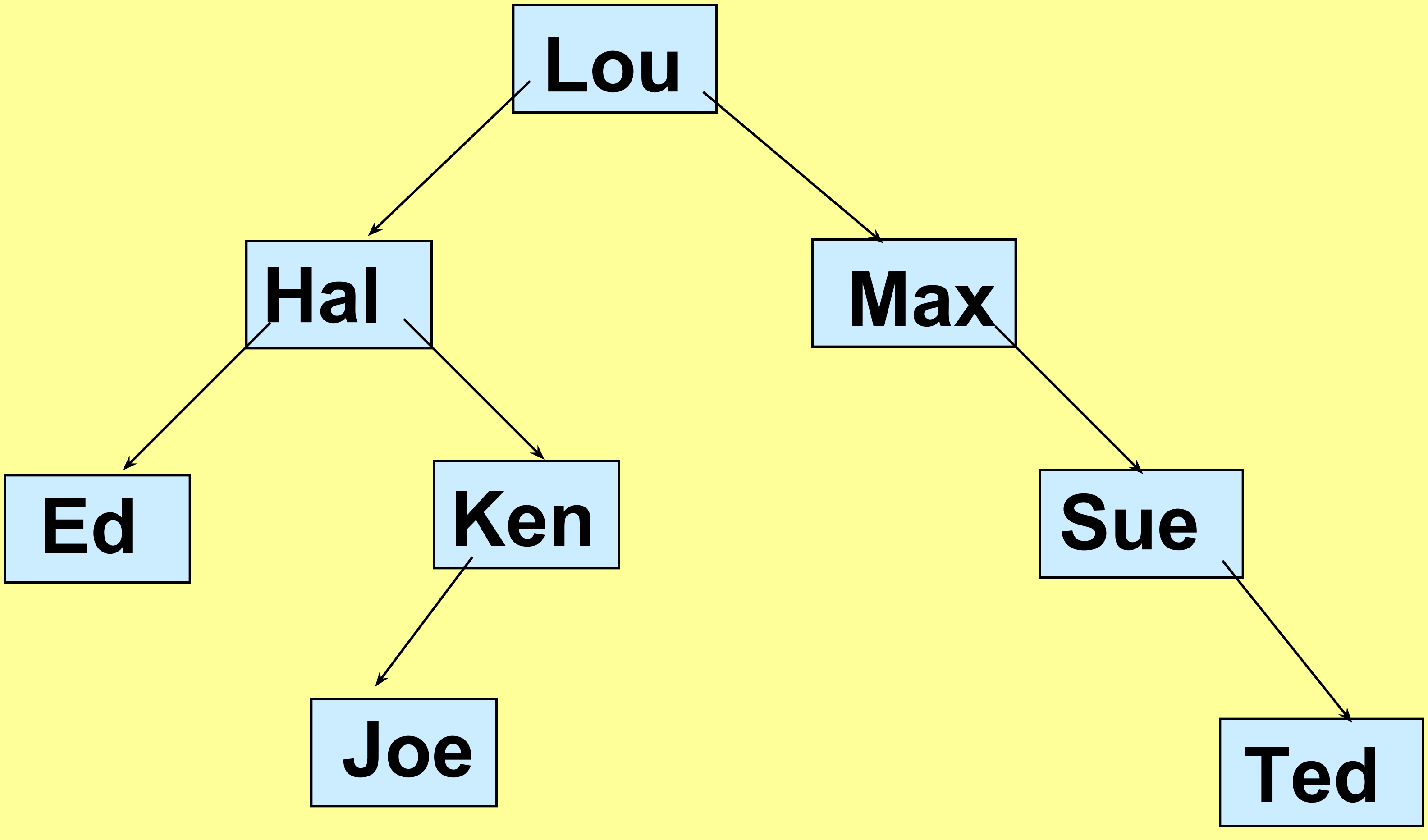
A full m-ary tree with

(i) n vertices has $i = (n-1)/m$ internal vertices and $l = [(m-1)n+1]/m$ leaves.

(ii) i internal vertices has $n = mi + 1$ vertices and $l = (m-1)\mathrm{i} + 1$ leaves.

(iii) l leaves has $n = (ml - 1)/(m-1)$ vertices and $i = (l-1)/(m-1)$ internal vertices.

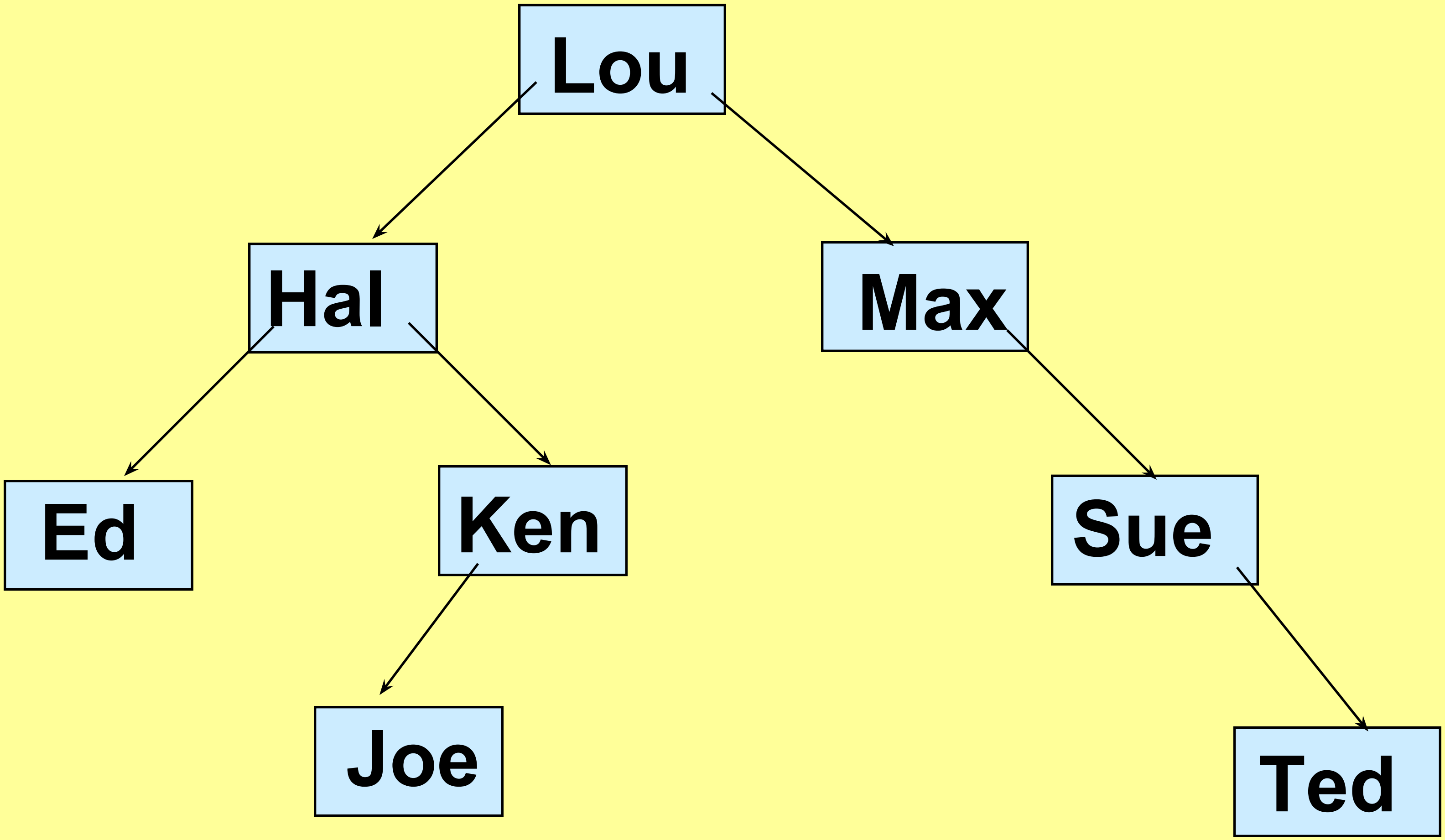
An Ordered Binary Tree



Balanced

- **A rooted binary tree of height H is called balanced if all its leaves are at levels H or H-1.**

Is this binary tree balanced?



Traversal Algorithms

- A traversal algorithm is a procedure for systematically visiting every vertex of an ordered binary tree.
- Tree traversals are defined recursively.
- Three traversals are named:

    preorder,

    inorder,

    postorder.

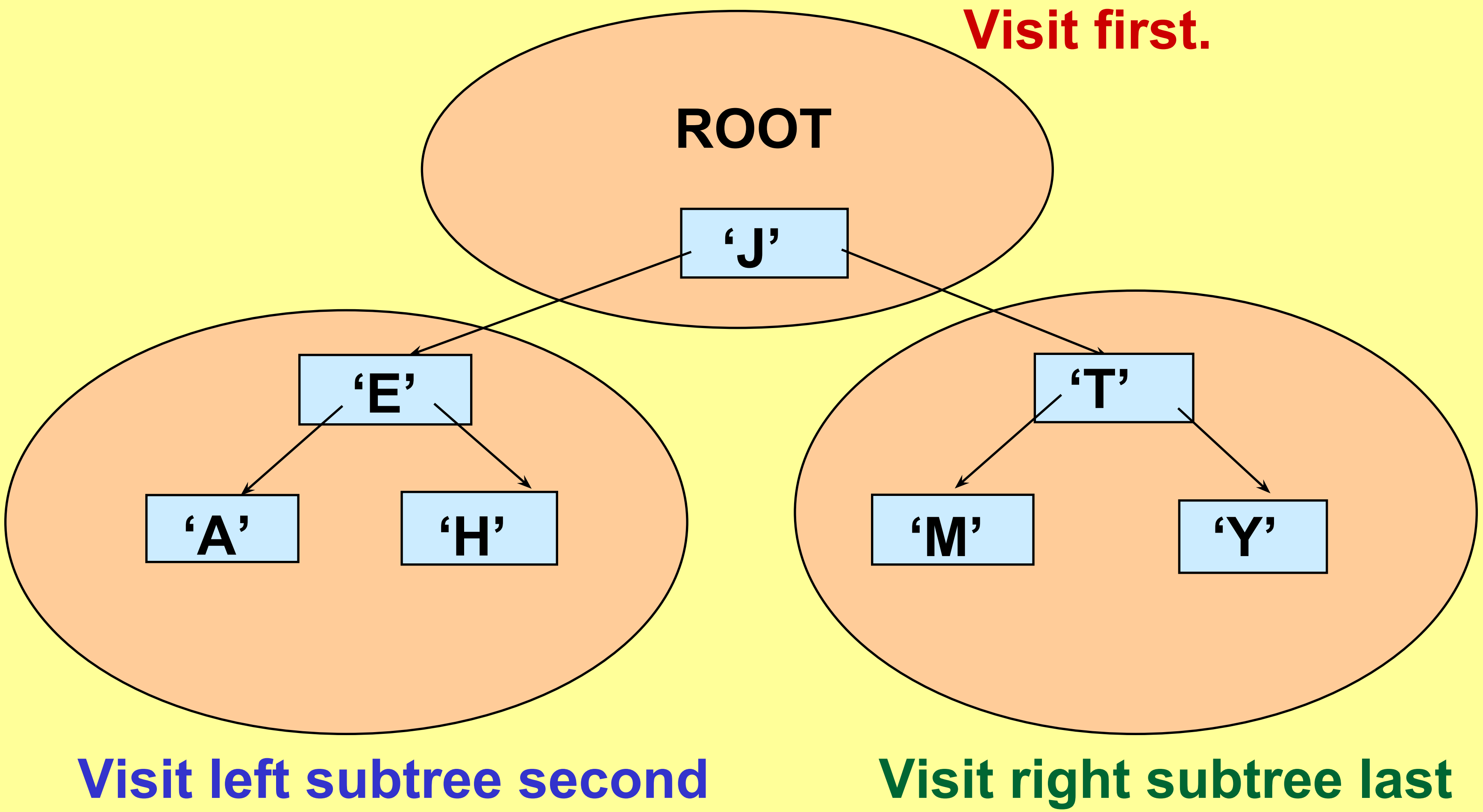
PREORDER Traversal Algorithm

Let T be an ordered binary tree with root r.

If T has only r, then r is the preorder traversal.

Otherwise, suppose T_1, T_2 are the left and right subtrees at r. The preorder traversal begins by visiting r. Then traverses T_1 in preorder, then traverses T_2 in preorder.

Preorder Traversal: J E A H T M Y



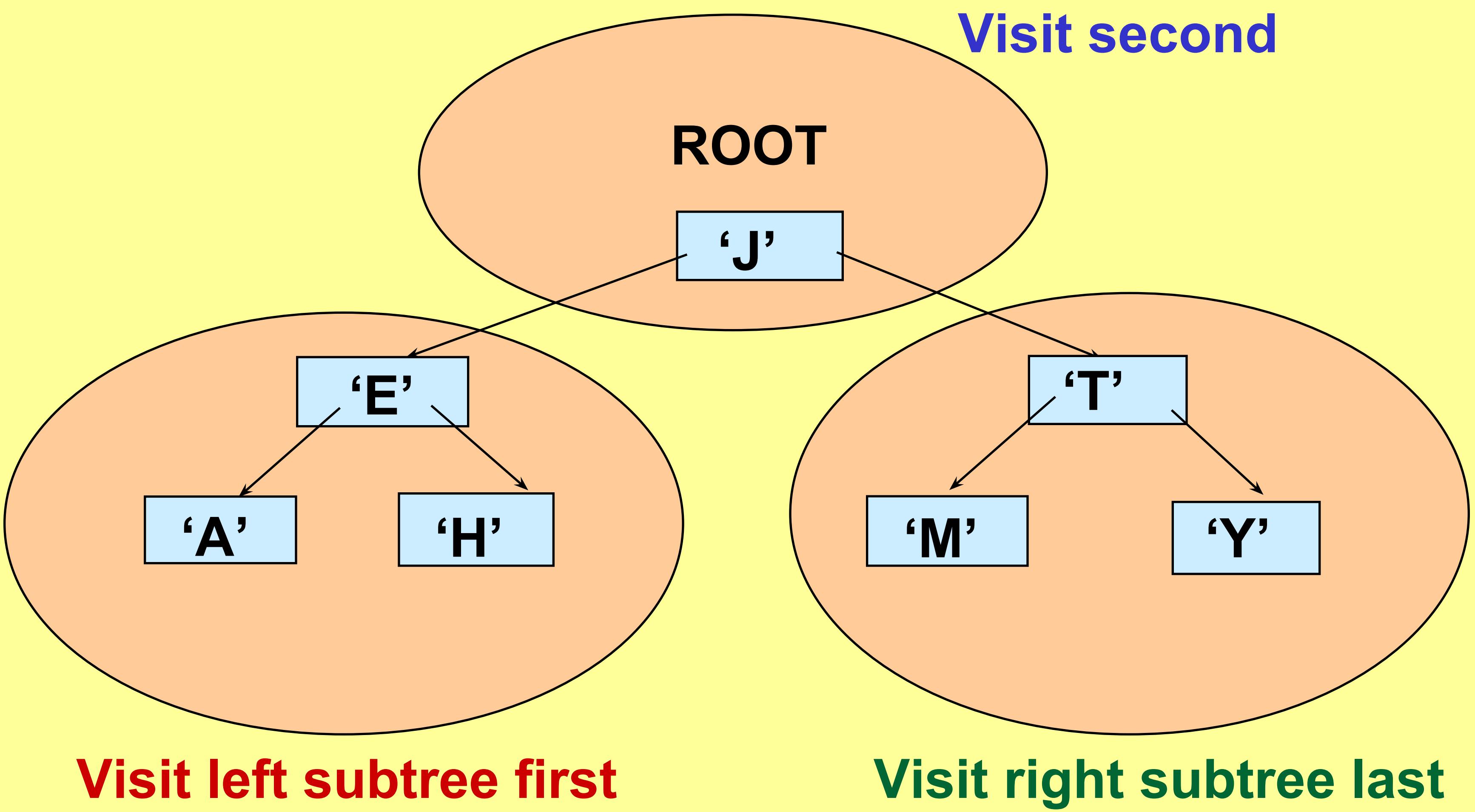
INORDER Traversal Algorithm

Let T be an ordered binary tree with root r.

If T has only r, then r is the inorder traversal.

Otherwise, suppose T_1, T_2 are the left and right subtrees at r. The inorder traversal begins by traversing T_1 in inorder. Then visits r, then traverses T_2 in inorder.

Inorder Traversal: A E H J M T Y



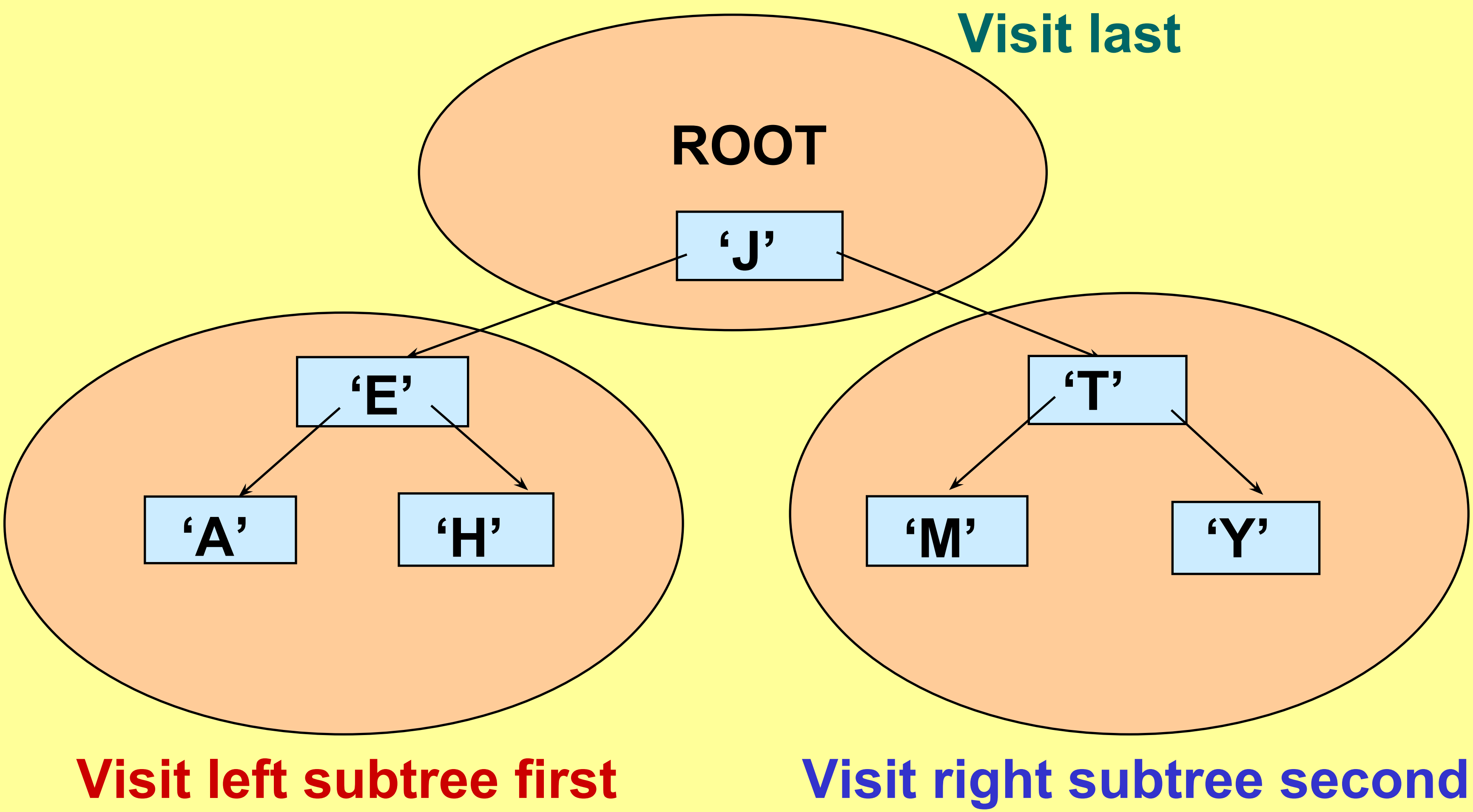
POSTORDER Traversal Algorithm

Let T be an ordered binary tree with root r.

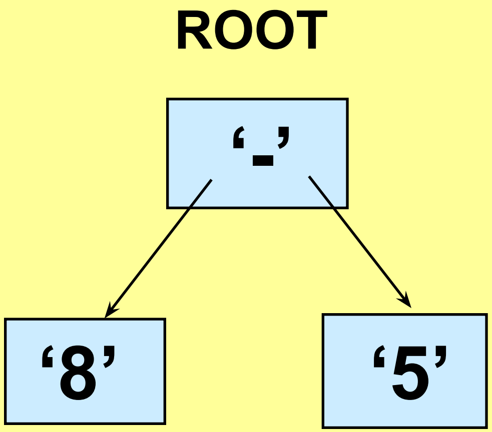
If T has only r, then r is the postorder traversal.

Otherwise, suppose T_1, T_2 are the left and right subtrees at r. The postorder traversal begins by traversing T_1 in postorder. Then traverses T_2 in postorder, then ends by visiting r.

Postorder Traversal: A H E M Y T J



A Binary Expression Tree



INORDER TRAVERSAL: `8 - 5 has value 3`

PREORDER TRAVERSAL: `- 8 5`

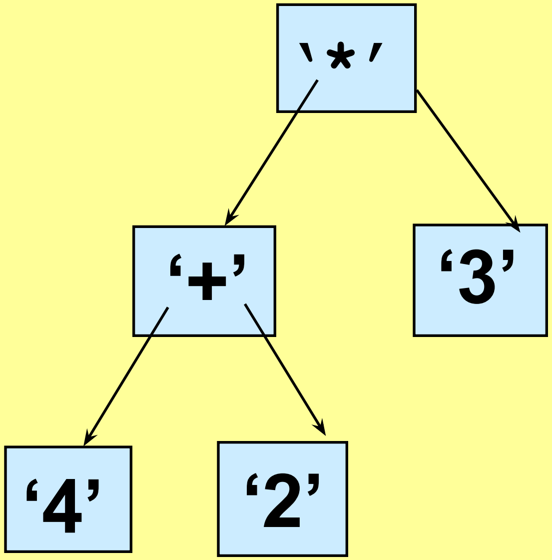
POSTORDER TRAVERSAL: `8 5 -`

A Binary Expression Tree is . . .

A special kind of binary tree in which:

1. Each leaf node contains a single operand,
2. Each nonleaf node contains a single binary operator, and
3. The left and right subtrees of an operator node represent subexpressions that must be evaluated before applying the operator at the root of the subtree.

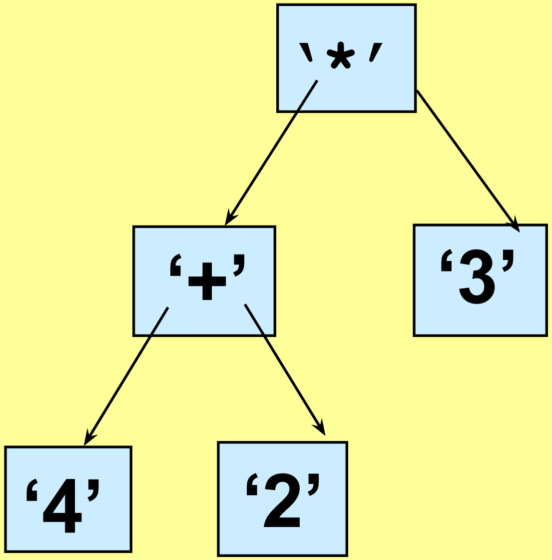
A Binary Expression Tree



What value does it have?

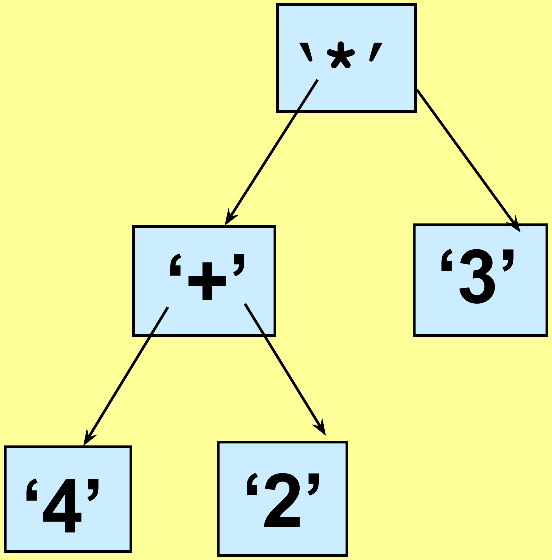
(4 + 2) * 3 = 18

A Binary Expression Tree



What infix, prefix, postfix expressions does it represent?

A Binary Expression Tree



Infix: ((4 + 2) * 3)

Prefix: * + 4 2 3 *evaluate from right*

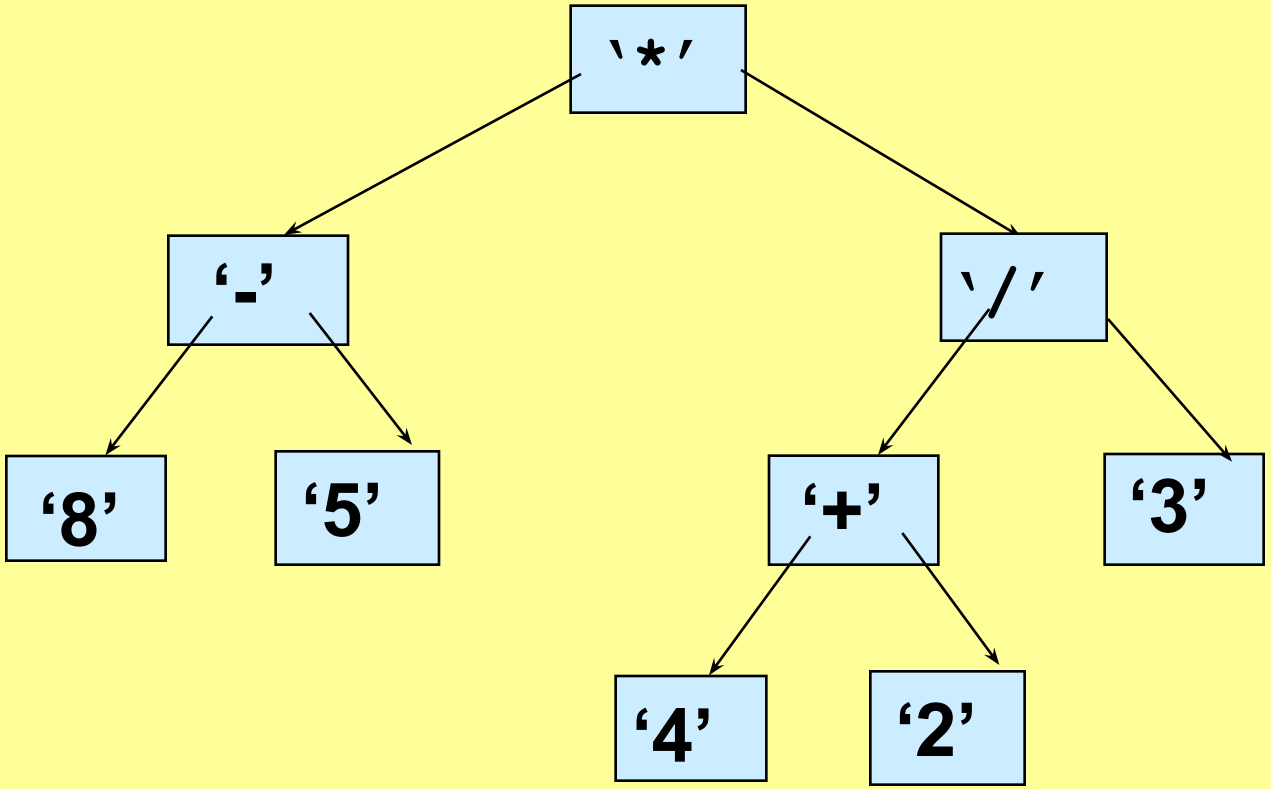
Postfix: 4 2 + 3 * *evaluate from left*

Levels Indicate Precedence

When a binary expression tree is used to represent an expression, the levels of the nodes in the tree indicate their relative precedence of evaluation.

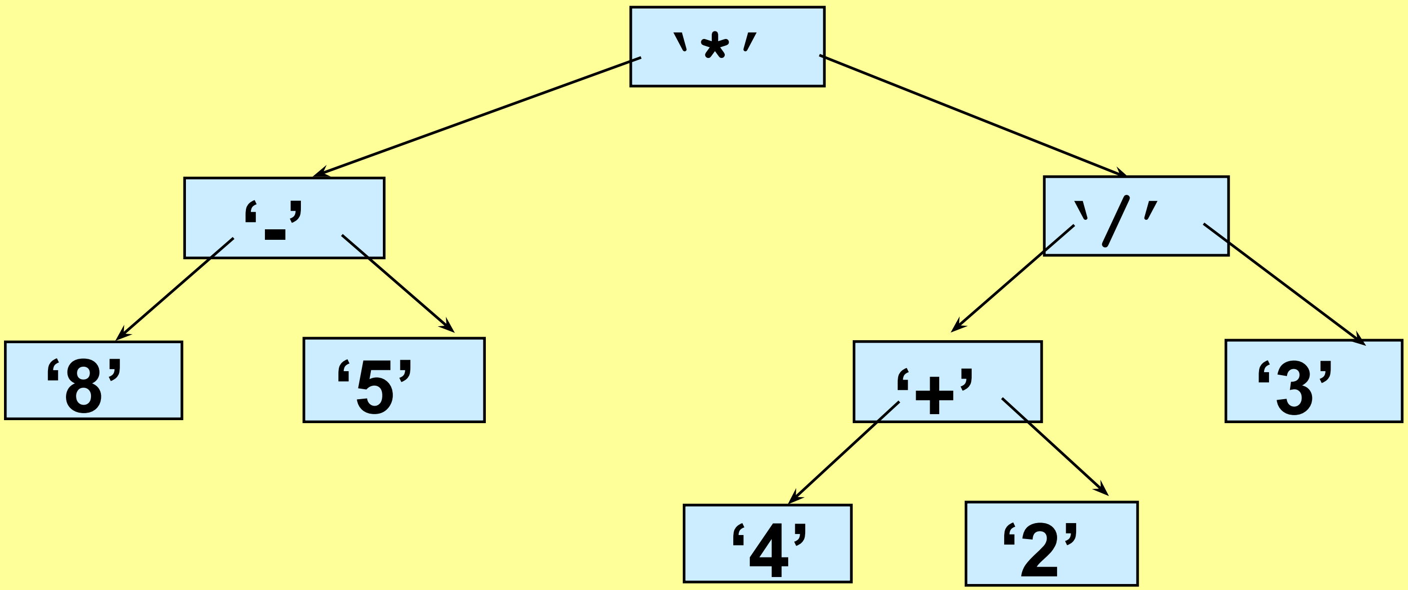
Operations at higher levels of the tree are evaluated later than those below them. The operation at the root is always the last operation performed.

Evaluate this binary expression tree



What infix, prefix, postfix expressions does it represent?

A binary expression tree

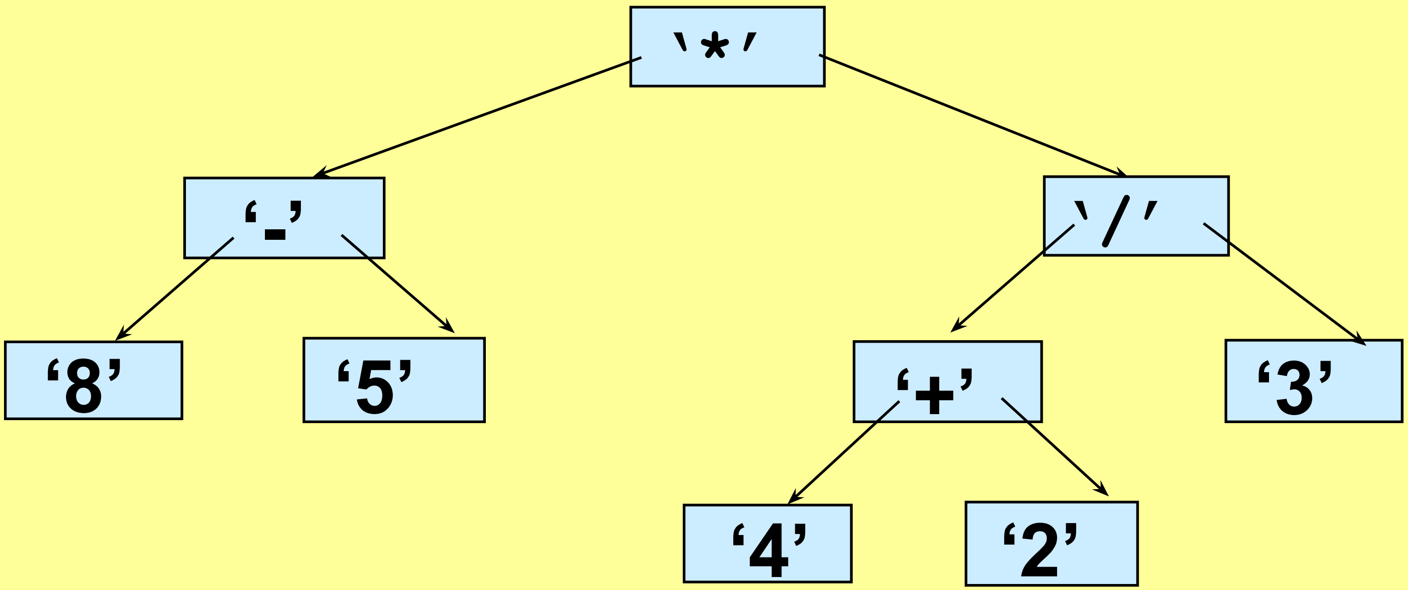


Infix: ((8 - 5) * ((4 + 2) / 3))

Prefix: * - 8 5 / + 4 2 3

Postfix: 8 5 - 4 2 + 3 / * *has operators in order used*

A binary expression tree

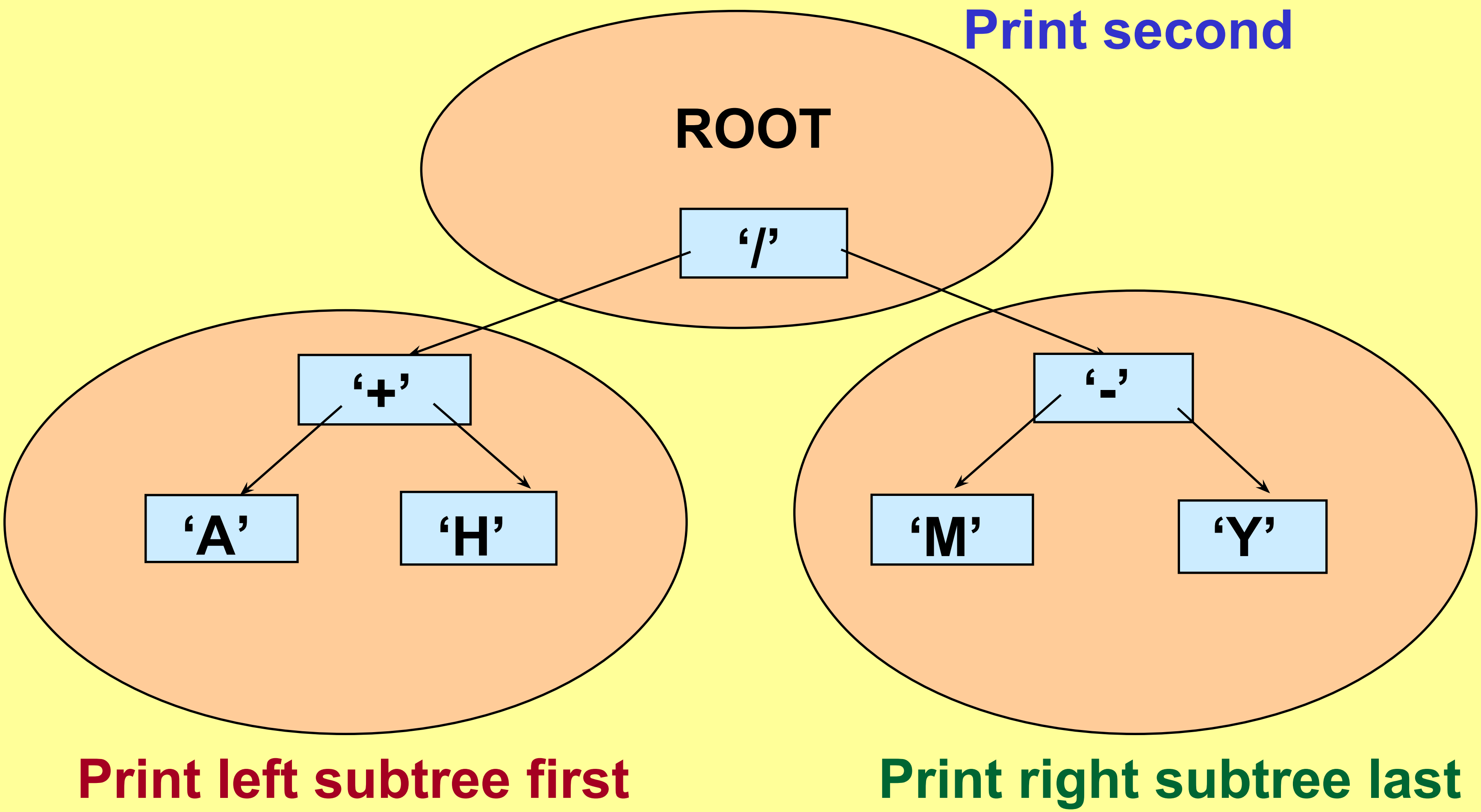


Infix: ((8 - 5) * ((4 + 2) / 3))

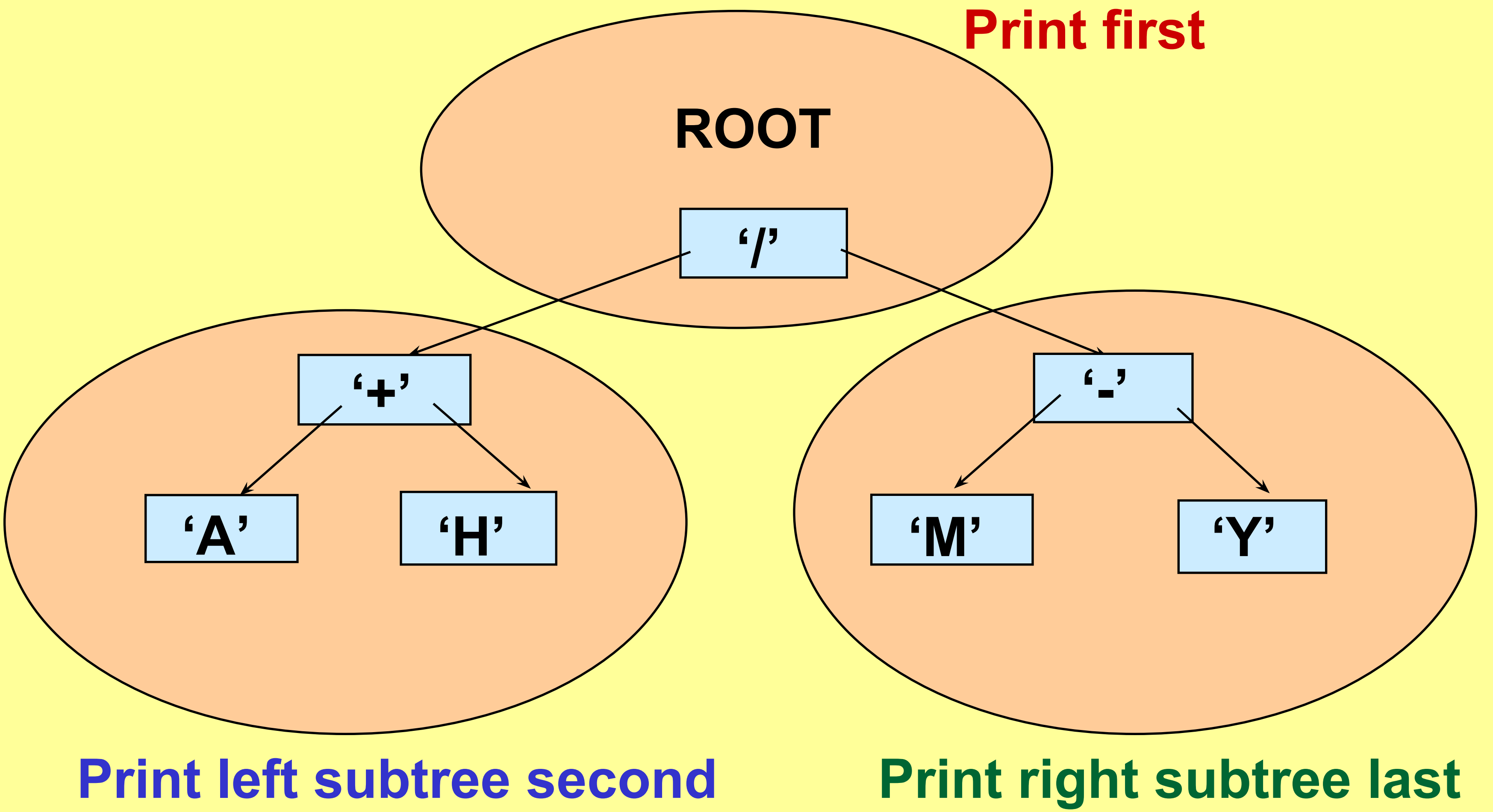
Prefix: * - 8 5 / + 4 2 3 *evaluate from right*

Postfix: 8 5 - 4 2 + 3 / * *evaluate from left*

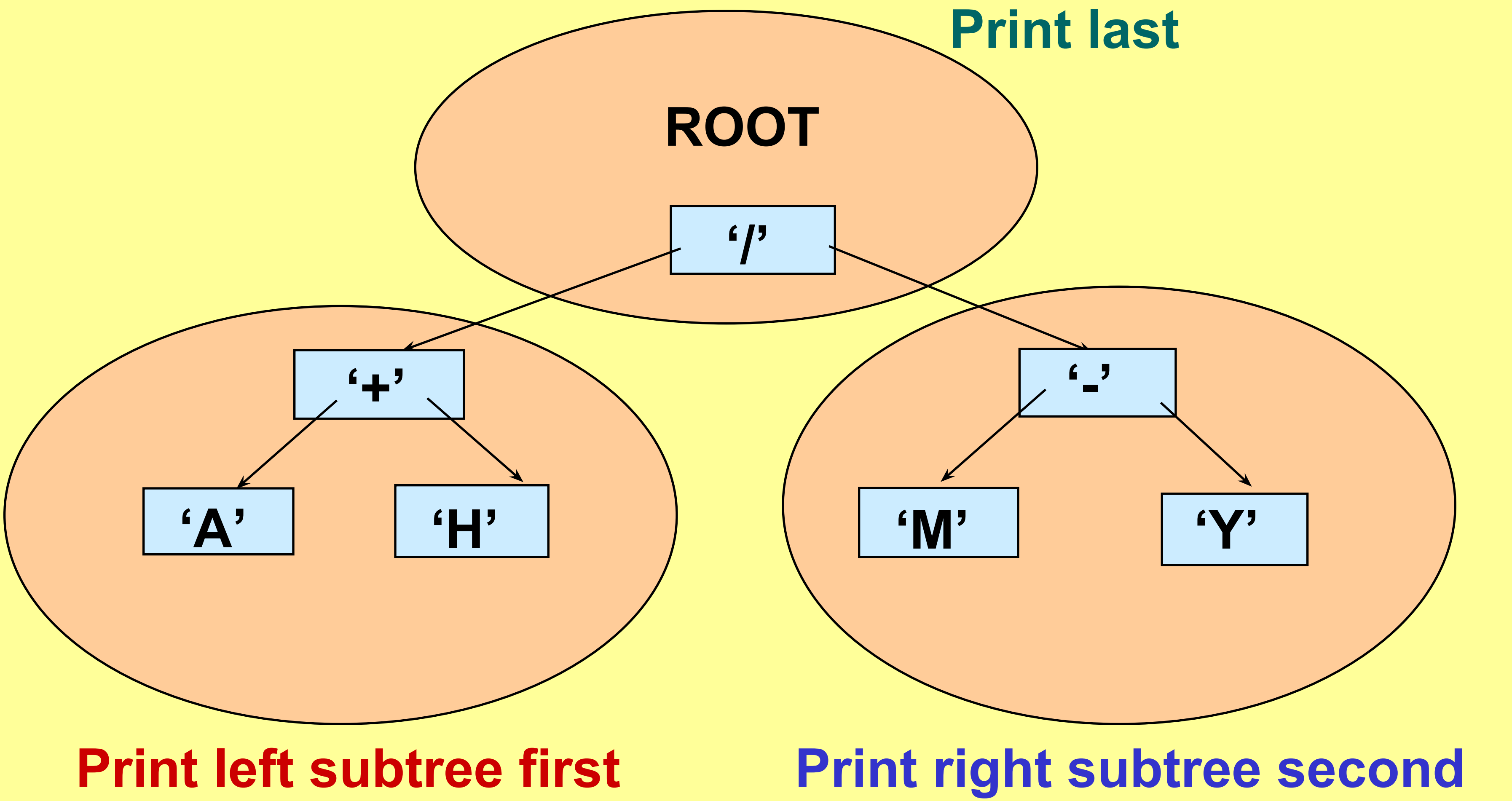
Inorder Traversal: (A + H) / (M - Y)



Preorder Traversal: / + A H - M Y



Postorder Traversal: A H + M Y - /



ACKNOWLEDGMENT:

This project was supported in part by the National Science Foundation under grant DUE-ATE 9950056.