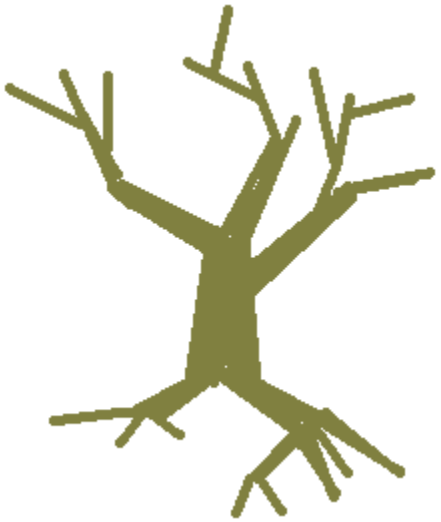
Trees 树

Discrete Mathematics and Its Applications

Kenneth H. Rosen

Trees

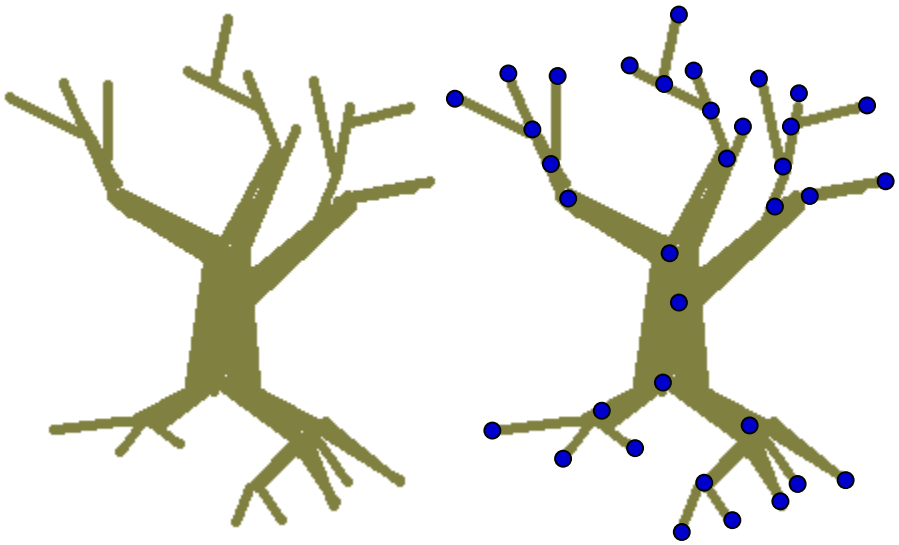
A very important type of graph in CS is called a *tree*:



Real Tree

Trees

A very important type of graph in CS is called a *tree*:

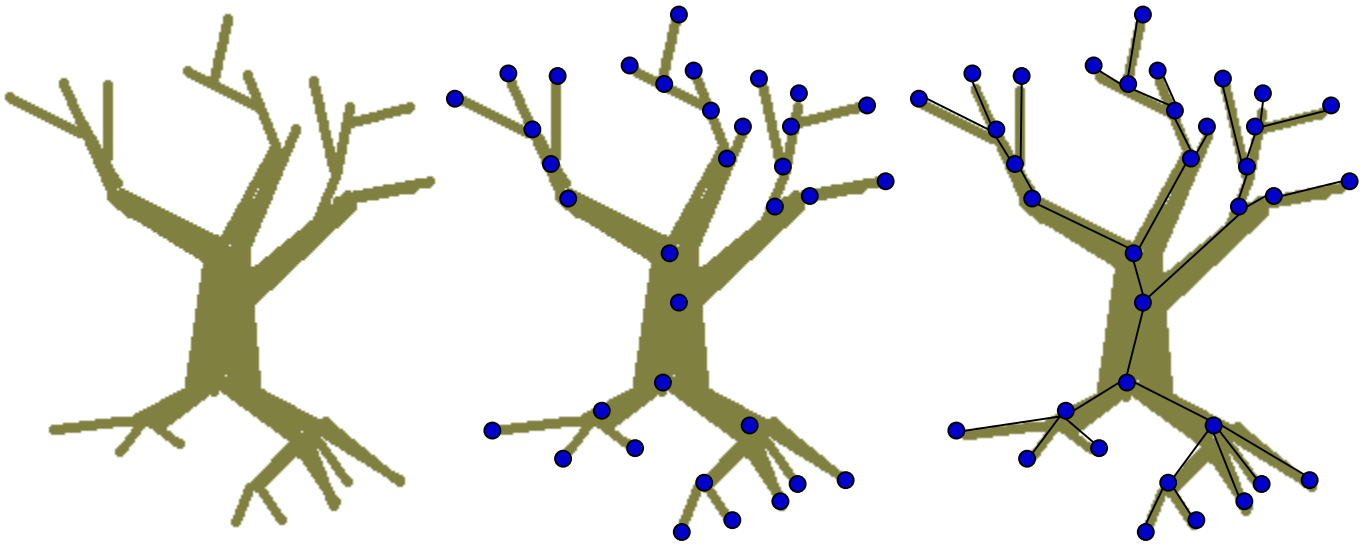


Real Tree

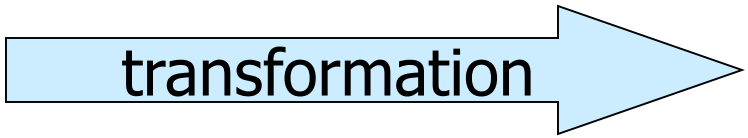
transformation

Trees

A very important type of graph in CS is called a *tree*:

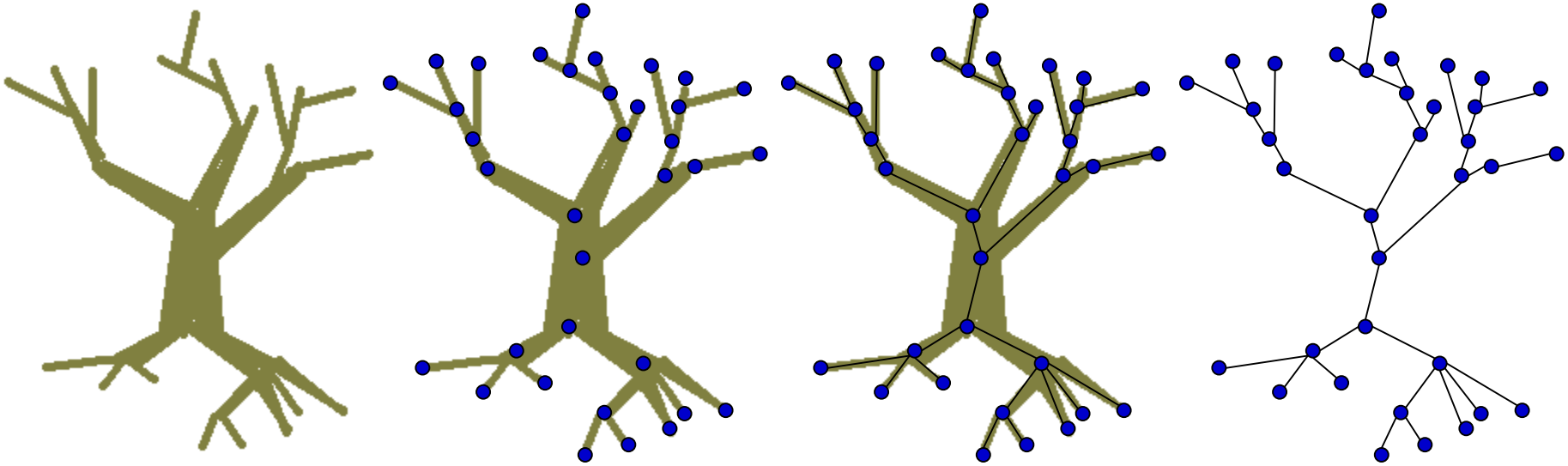


Real Tree



Trees

A very important type of graph in CS is called a *tree*:



Real Tree

transformation

Abstract Tree

Tree

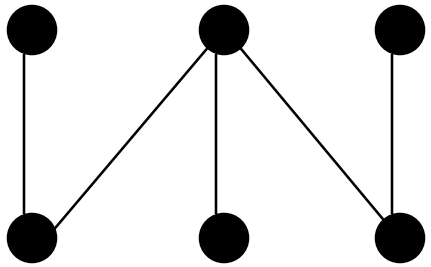
Def 1. A tree is a connected undirected graph with no simple circuits.

没有简单回路的连通无向图称为树，也称为无向树

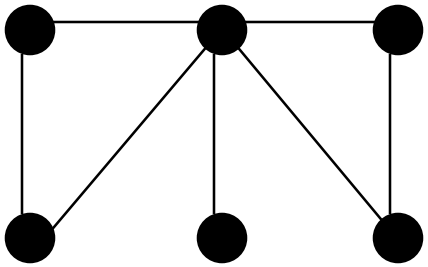
无向树中, 度为1的结点称为叶结点，简称树叶。

Which graphs are trees?

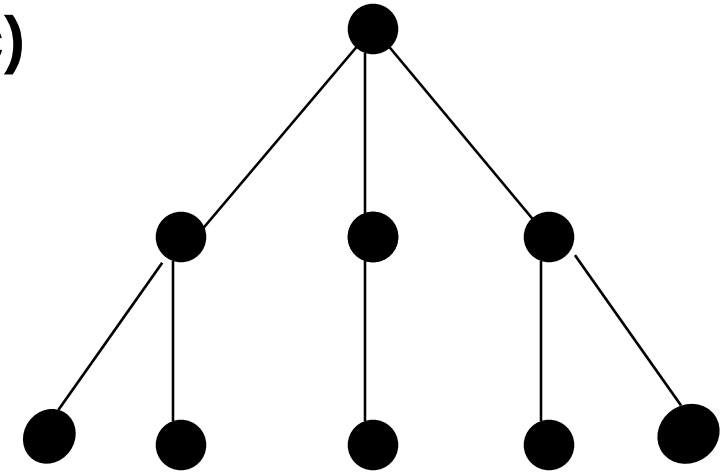
a)



b)



c)



树的基本性质

Theorem 1. An undirected simple graph is a tree if and only if there is a unique simple path between any two distinct vertices.一个简单无向图为树当且仅当任意两个不同结点之间有唯一的一条简单路 （**Why?**）

性质1：树的每一条边都是割边。

性质2：树一定是平面图

Theorem 2. A tree with n vertices has just n-1 edges. *(important result)*

For any (n,m)树必然有：n=m+1

Note: Using induction to prove this theorem （第2数学归纳法证明）.

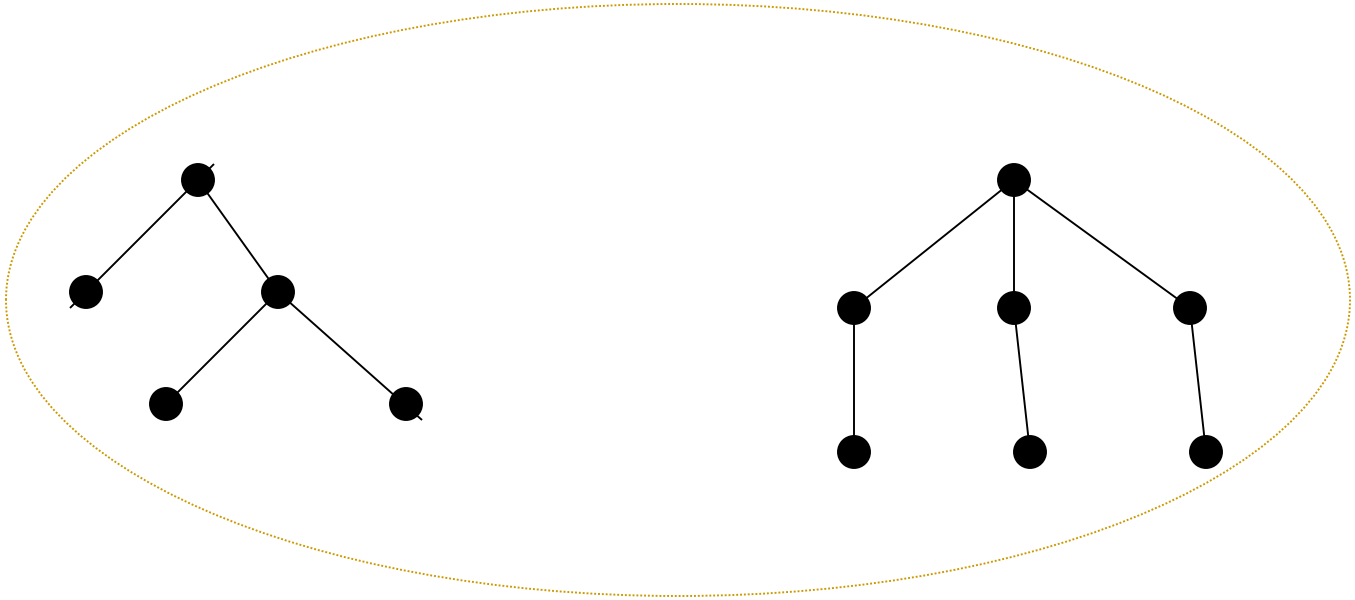
定理2的证明

- 证明：
- （1）对于只有一个结点或者两个结点的树，结论显然成立
- （2）假设图G有n个结点，m条边，n>1. 归纳假设当k小于n时，具有k个结点的树的边数是k-1.
- 从图G种任意去掉一条边，得到两个连通分支，每个分支都是一棵树，而且其结点数都小于n；分别记这两个分支的树为T1,T2， 它们的结点数分别为n1,n2, 边数分别为m1, m2;
- (3) 那么根据归纳假设n1=m1+1, n2=m2+1;
- (4) 显然m1+m2+1 = m
- 于是……

- 证明方法2：用平面图的欧拉公式证明

Forest 树林

Graphs containing no simple circuits that are not connected, but each connected component is a tree.



思考问题：树林的边的数目与结点数的关系如何？

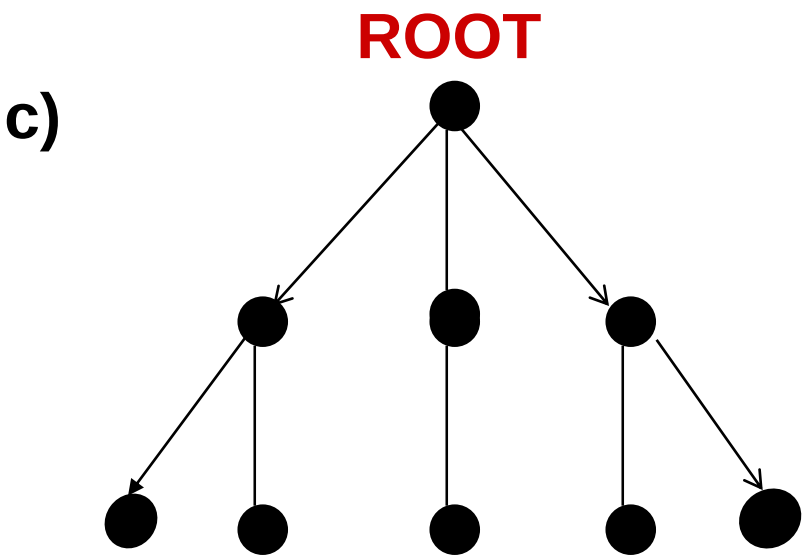
Root Tree 根树

- **Definition**: A *rooted tree* is a tree in which one vertex has been designated as the root and every edge is directed away from the root. 一棵指定了根结点，而且每条边的方向是离开根的方向的树

- Think about the directory tree
- Organization tree
- XML documents
- 数据结构和算法中用到的树，大多数指的是根树

Specify a vertex as root
指定一个结点为根

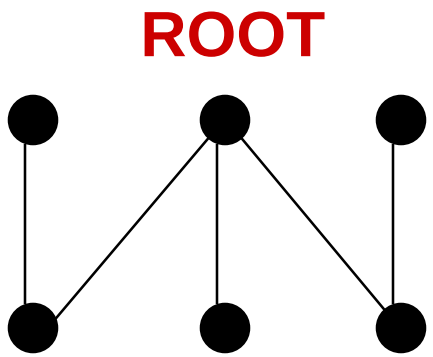
Then, direct each edge away from the root.
给每条边指定一个方向：离开根的方向



Specify a root.

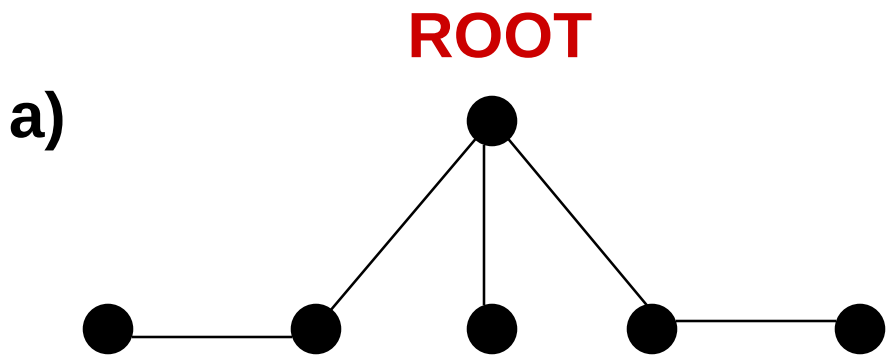
Then, direct each edge away from the root.

a)



Specify a root.

Then, direct each edge away from the root.

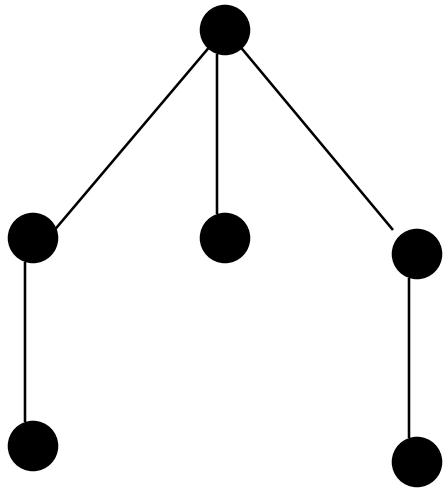


Specify a root.

Then, direct each edge away from the root.

ROOT

a)



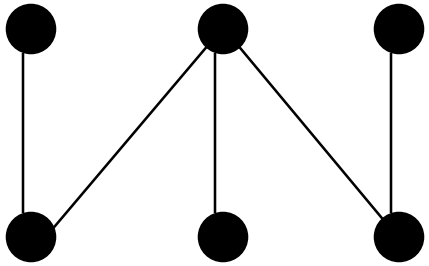
The result is a directed graph, called a rooted tree 一个有向图

What if a different root is chosen?

Then, direct each edge away from the root.

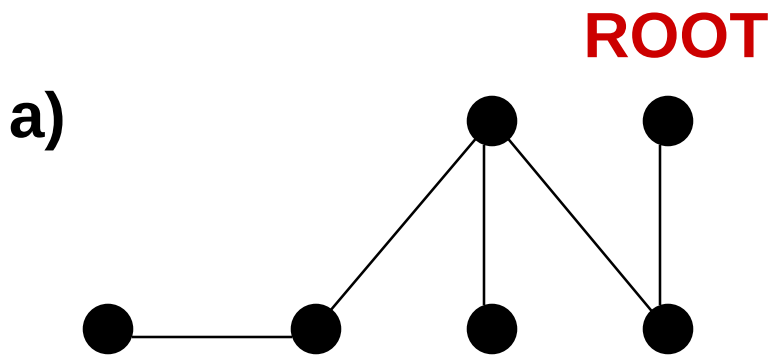
ROOT

a)



What if a different root is chosen?

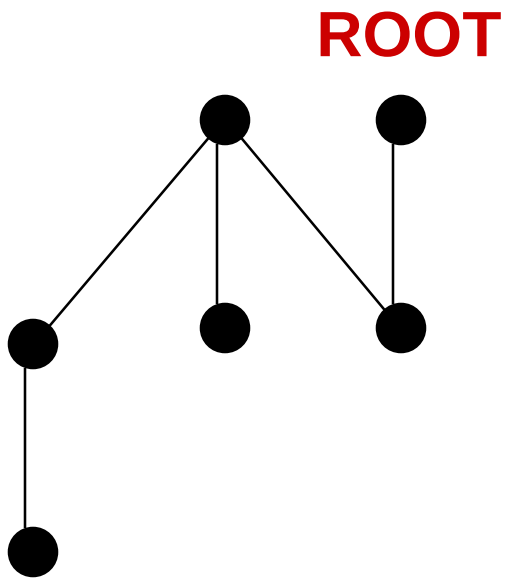
Then, direct each edge away from the root.



What if a different root is chosen?

Then, direct each edge away from the root.

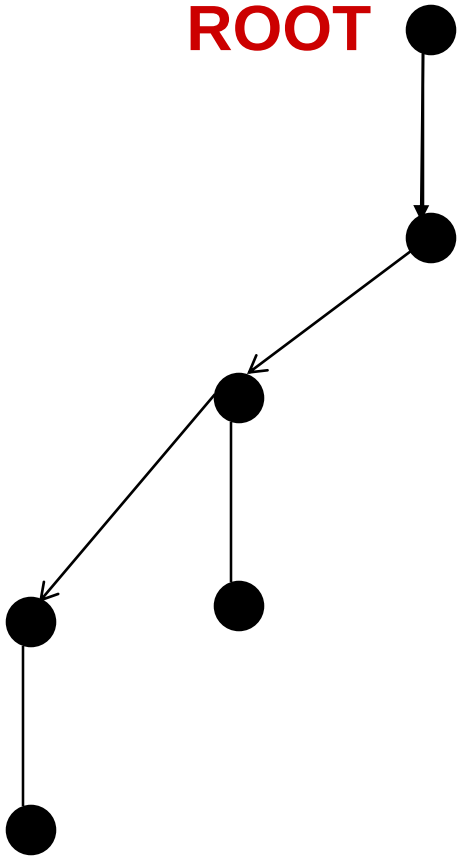
a)



What if a different root is chosen?

Then, direct each edge away from the root.

a)

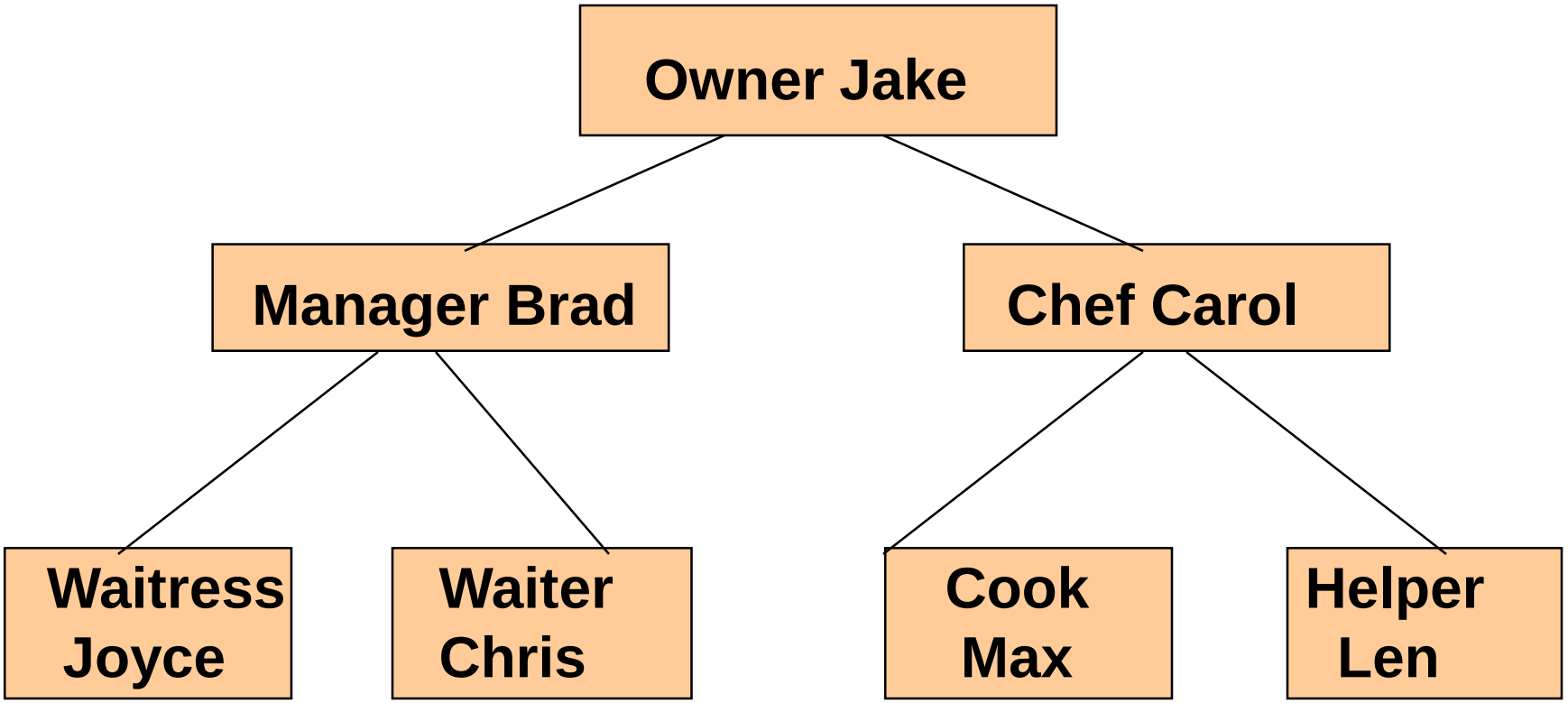


A different rooted tree

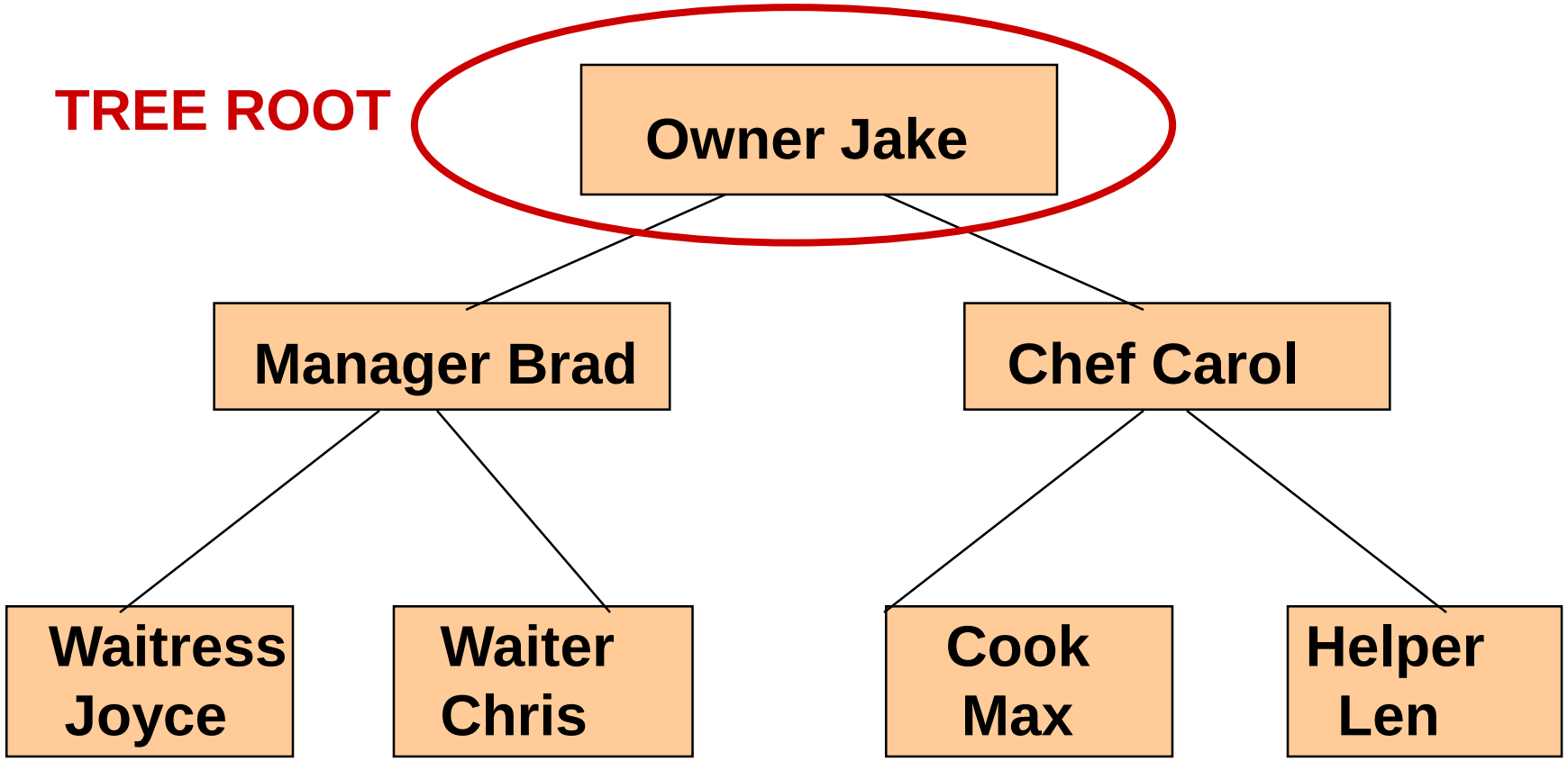
根树相关概念

- 根树：是有向树 （特殊的有向图）
- 根：入度为0的点（唯一的）
- 所有其它结点入度为1；
- 叶结点：入度为1出度为0的结点
- 除叶结点外，根树所有结点出度都大于0

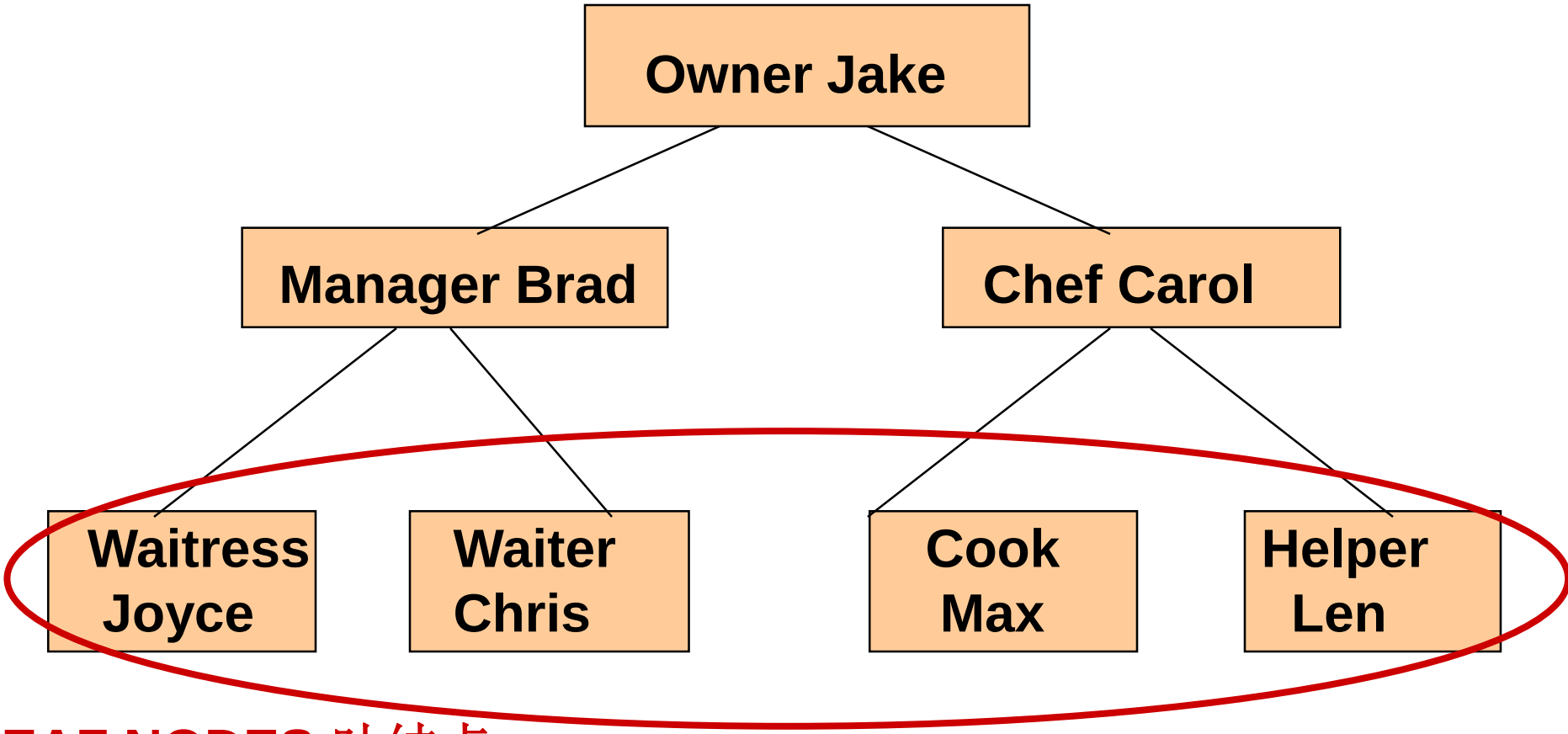
Jake's Pizza Shop Tree



A Tree Has a Root



Leaf nodes have no children

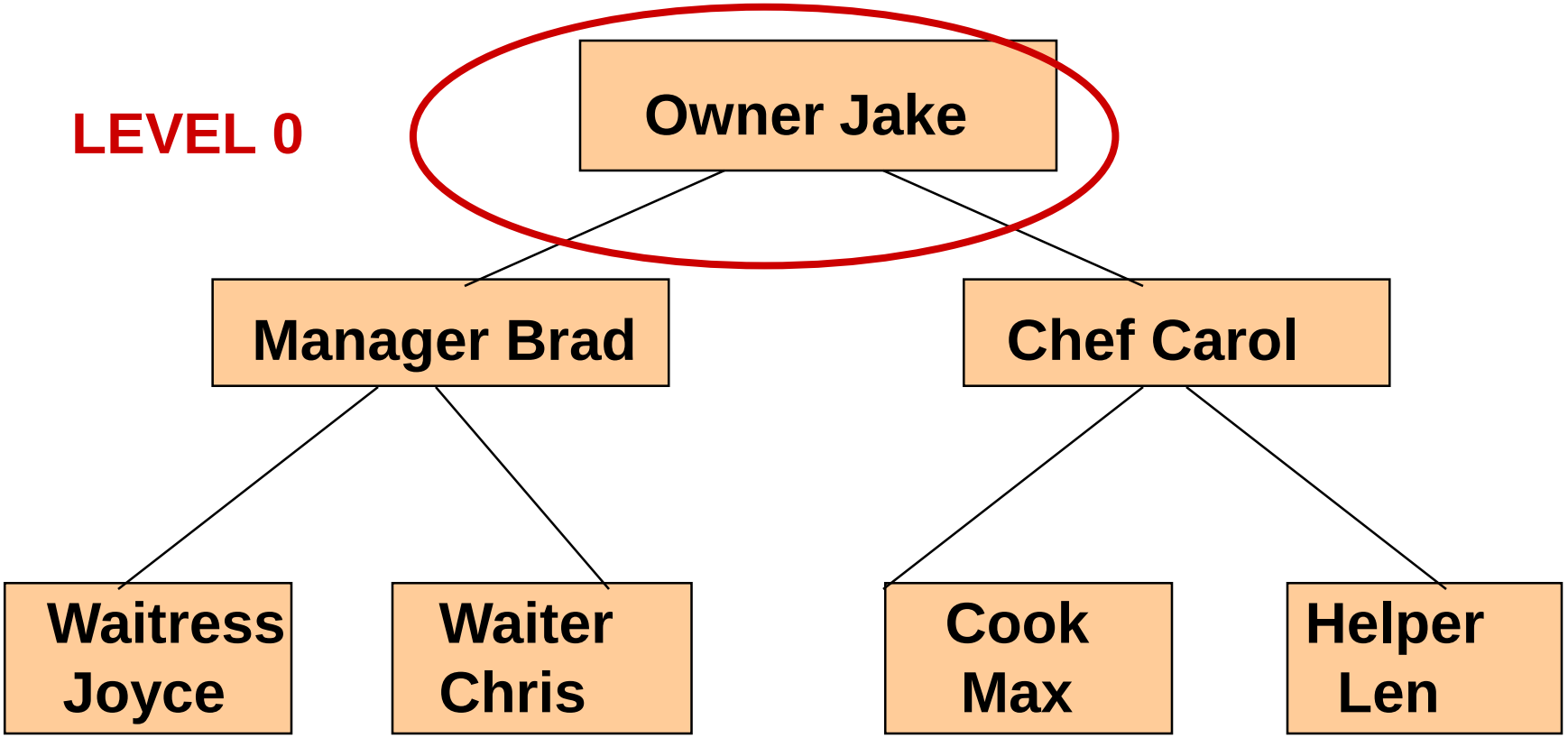


Level of Vertext 结点的层

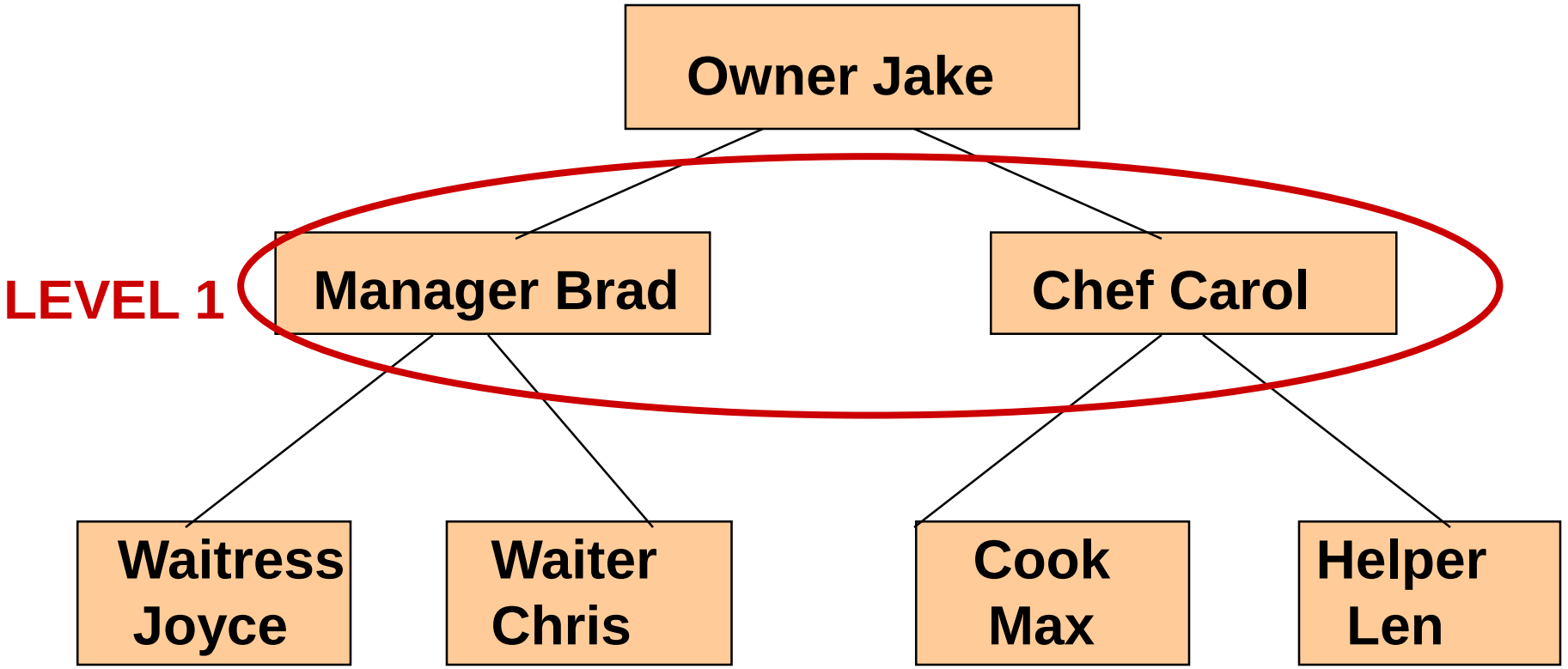
- The level of vertex v in a rooted tree is the length of the unique simple path from the root to v. 从根结点到v的唯一的简单路的长度

- 问题：这个长度是确定的吗？

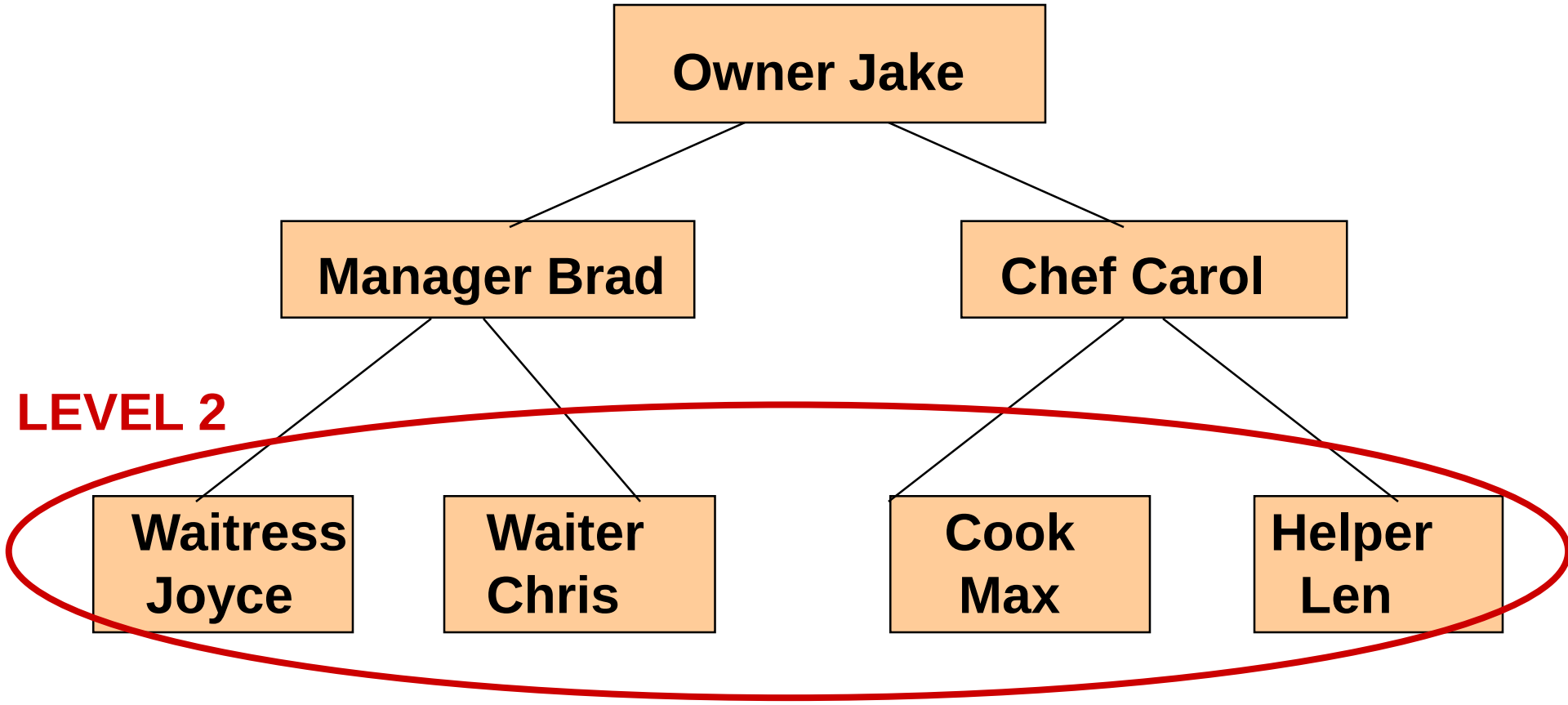
A Rooted Tree Has Levels



Level One


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

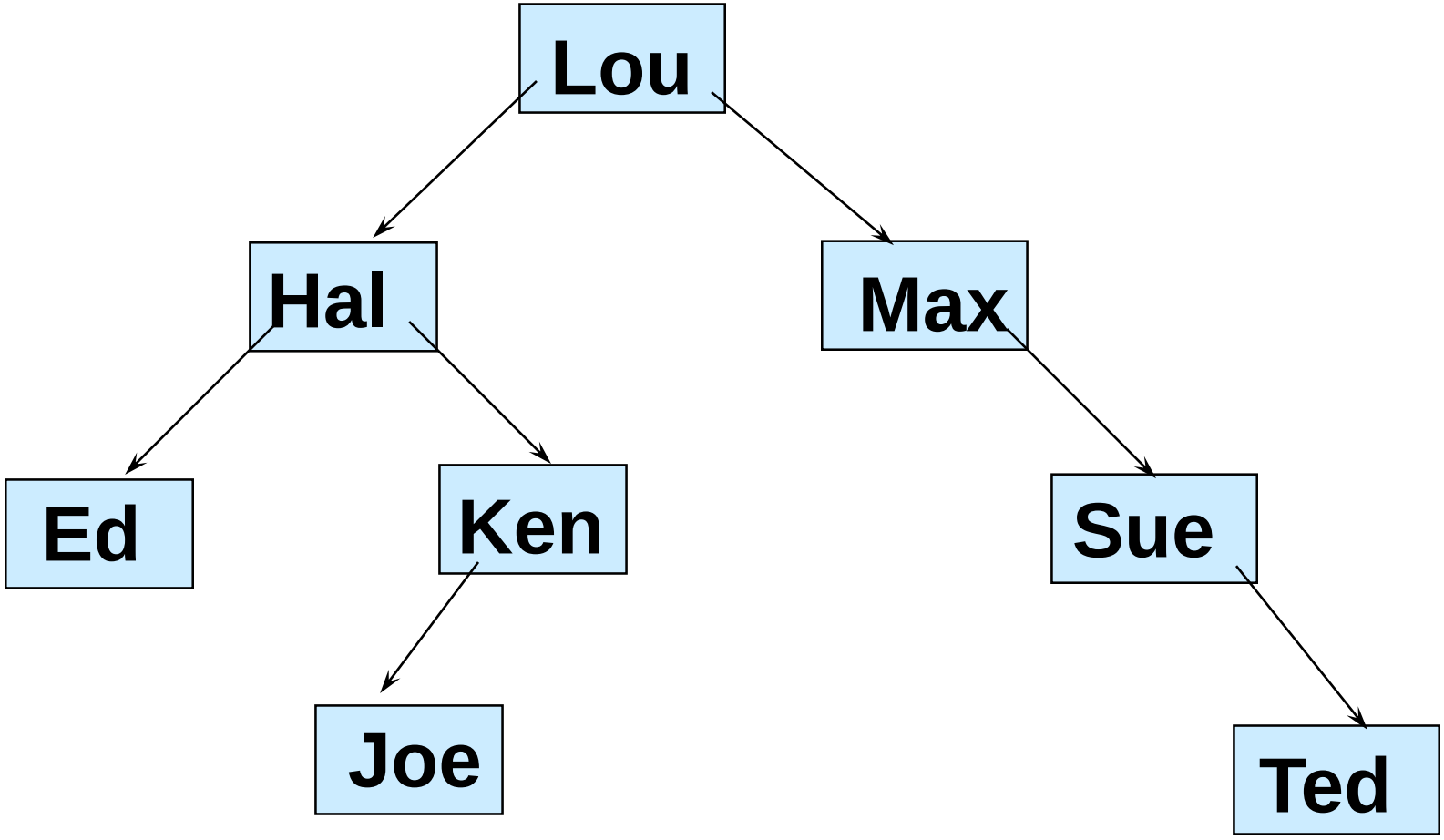
Level Two


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

Height 树高

- **The height of a rooted tree is the maximum of the levels of its vertices.**
- **最高层数**

What is the height?

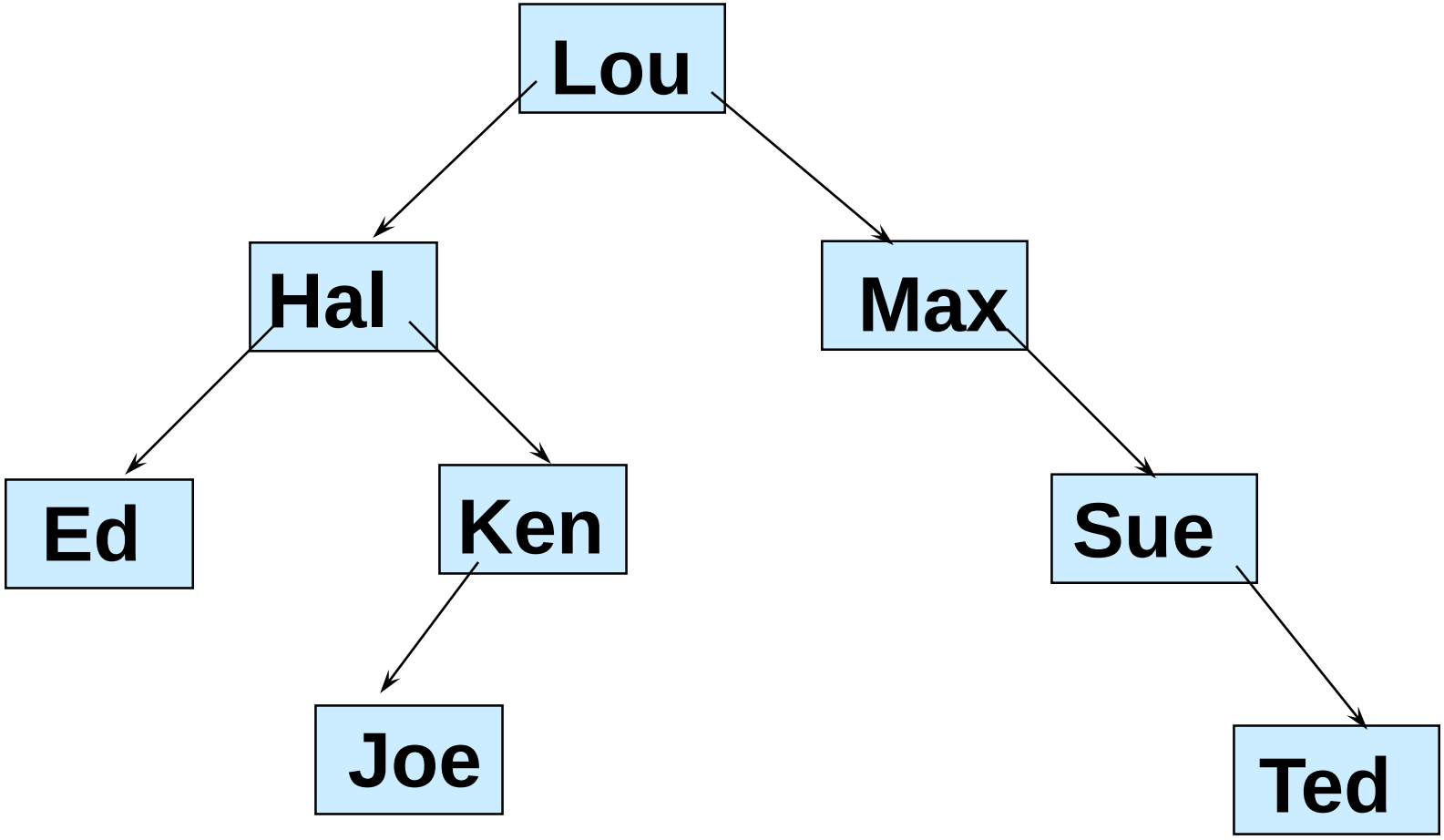


Parent 父结点、子结点

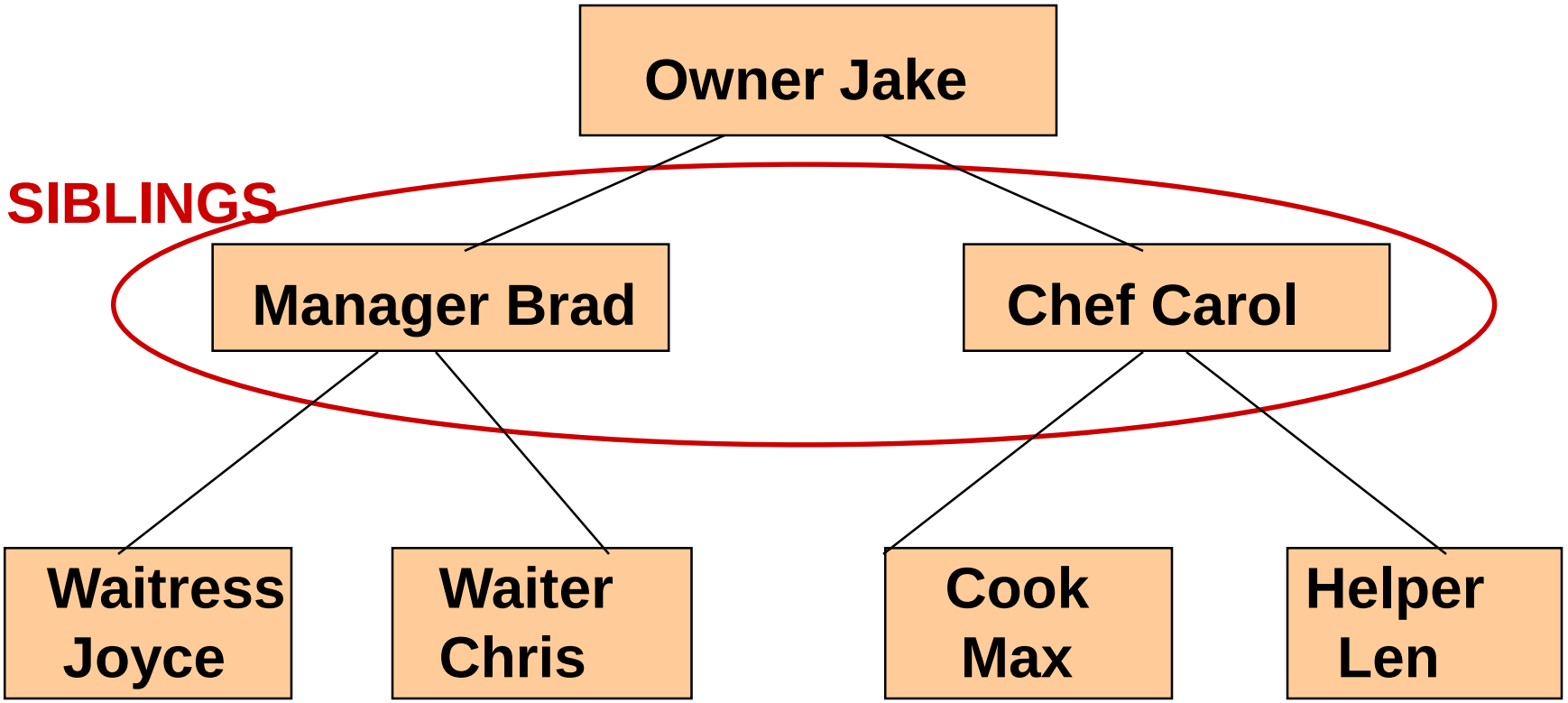
- **The parent of a non-root vertex v （非根结点）is the unique vertex u with a directed edge （有向边）from u to v； V称为u的子结点。**

- 问题：根树的任一个结点是否存在父结点？如果存在是否唯一？
- 任一结点是否存在子结点，子结点是否唯一？ 为什么？
- 根树中任一条边的起点与终点的关系？

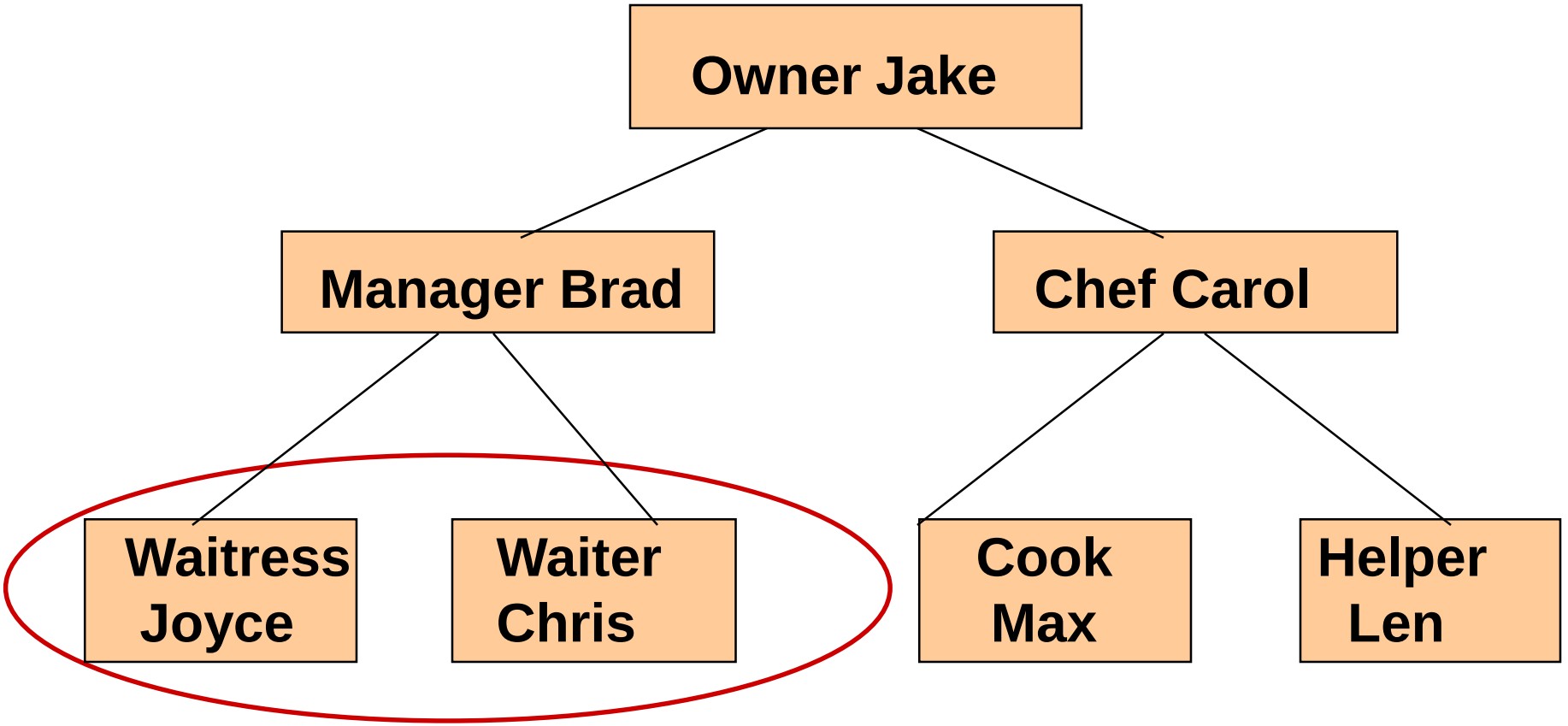
What is the parent of Ed?

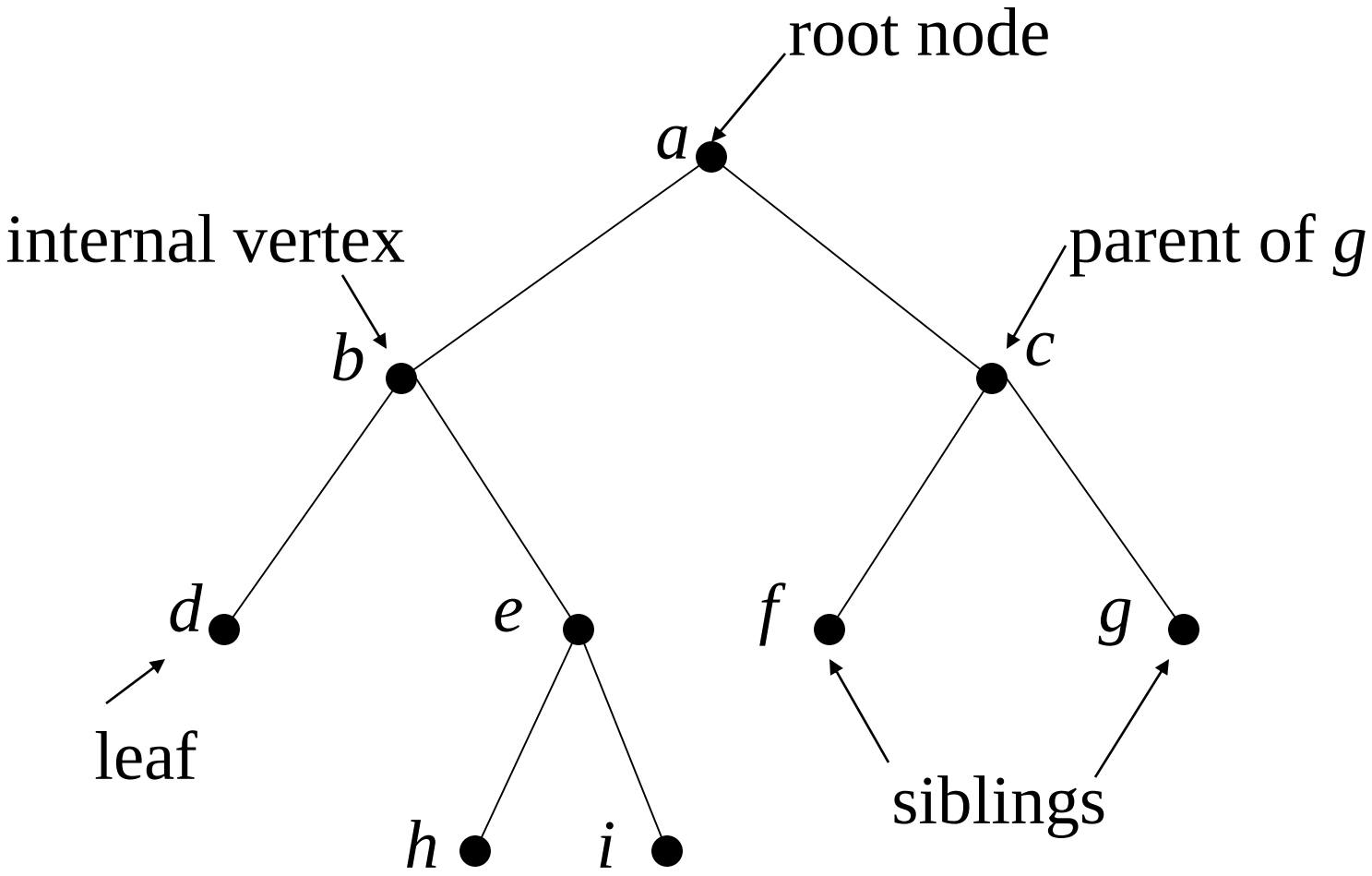


Sibling nodes （兄弟结点）
have same parent



Sibling nodes have same parent



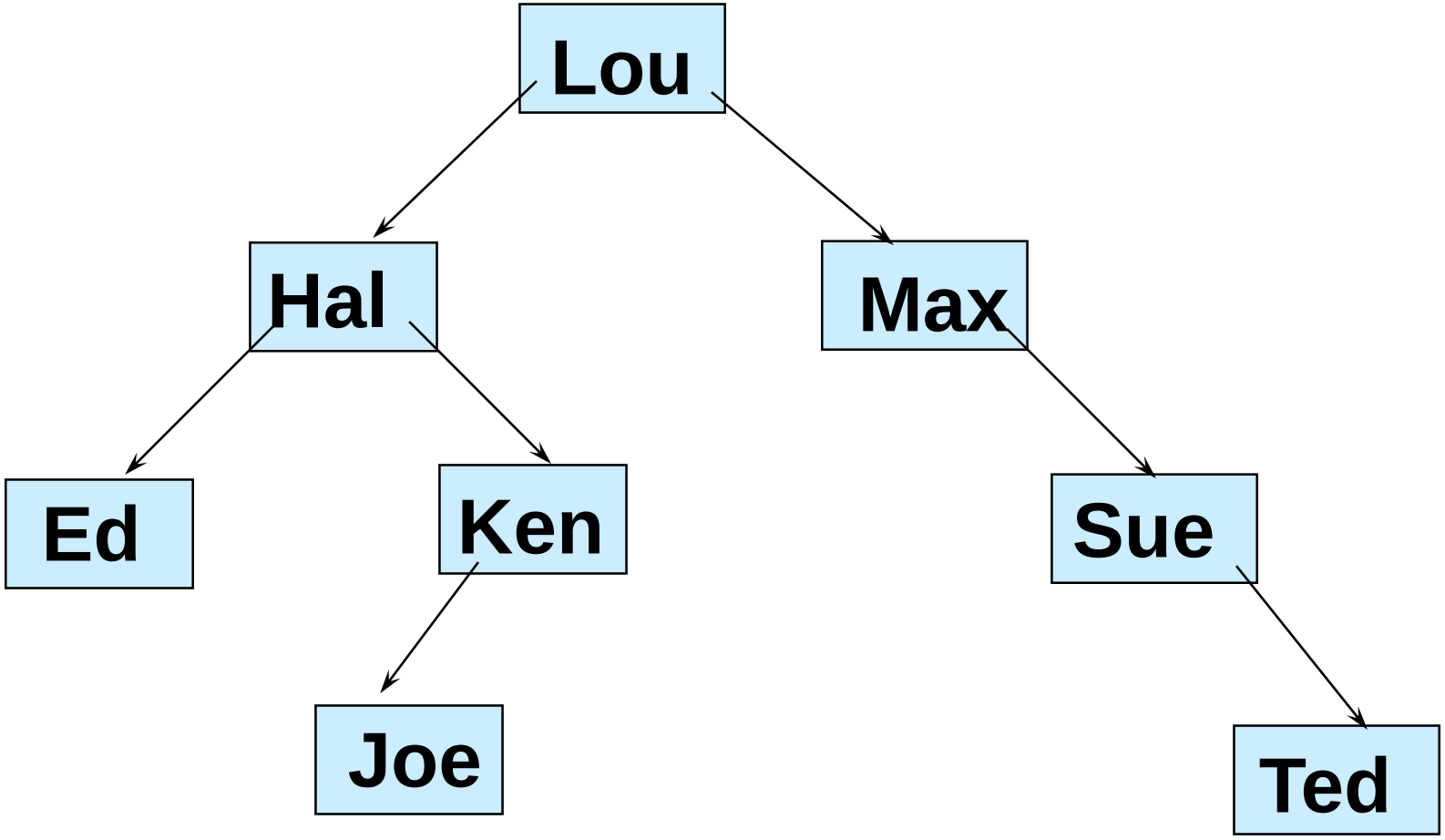

root node
a
internal vertex
parent of g
b
c
d
e
f
g
leaf
siblings
h
i

Leaf 叶结点

- A vertex is called a leaf if it has no children.
  没有子结点的结点称为叶结点
- 也是出度为0的点

- 根树的叶结点的度肯定为1， 是否度为1的结点就一定是叶结点？

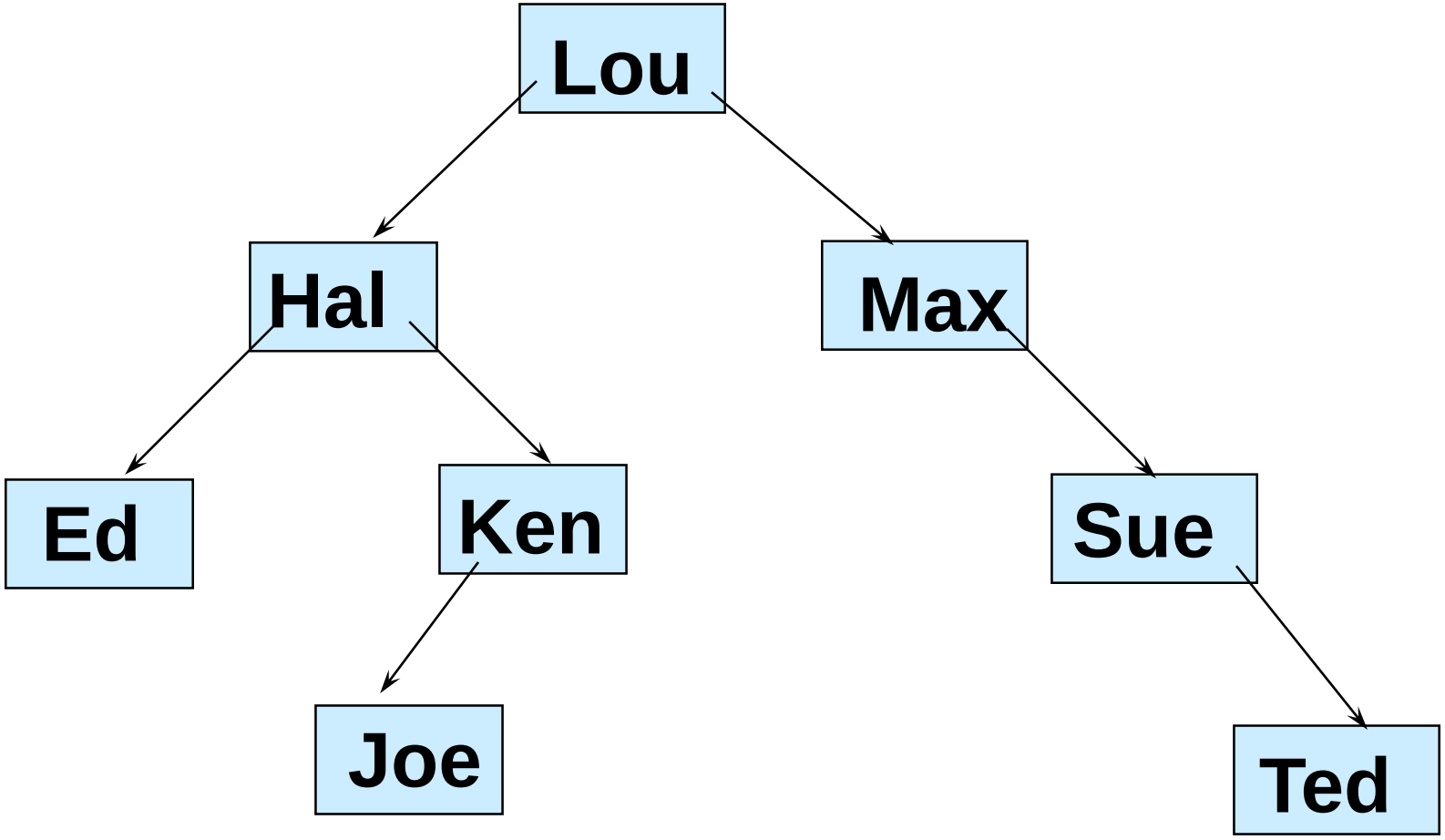
How many leaves?



Ancestors 祖先结点

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

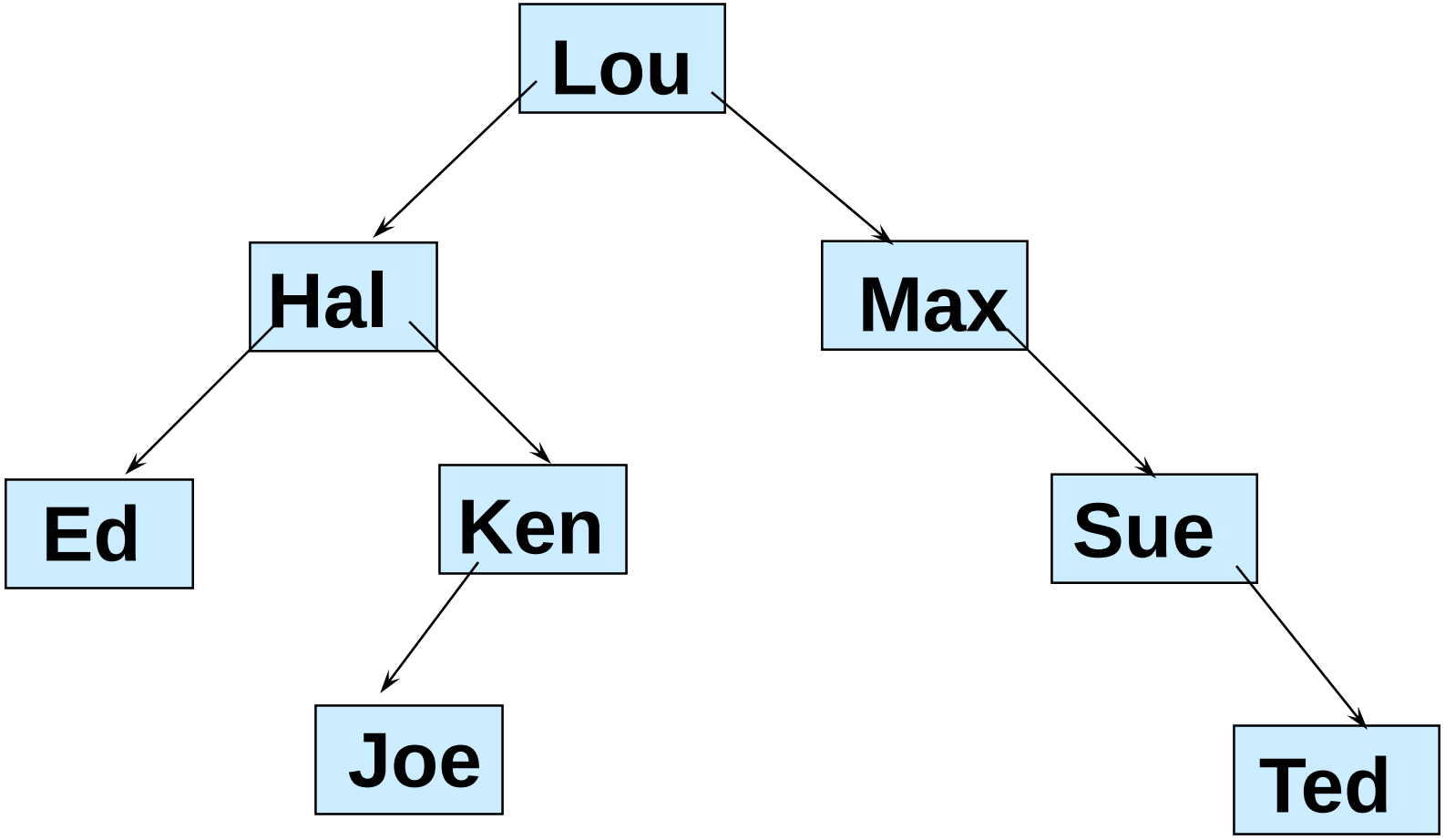
How many ancestors of Ken?



Descendants子孙结点

- **The descendants of vertex v are all the vertices that have v as an ancestor.**
- 子孙后代结点

How many descendants of Hal?



Internal Vertex(内结点)

A vertex that has children is called an internal vertex. 有子结点的结点称为内结点

以v为根的子树：The subtree with vertex v as its root is the subgraph of the tree consisting of vertex v and its descendants and all edges incident to those descendants.

包含v及其所有子孙结点以及相关的边的子树

Examples…

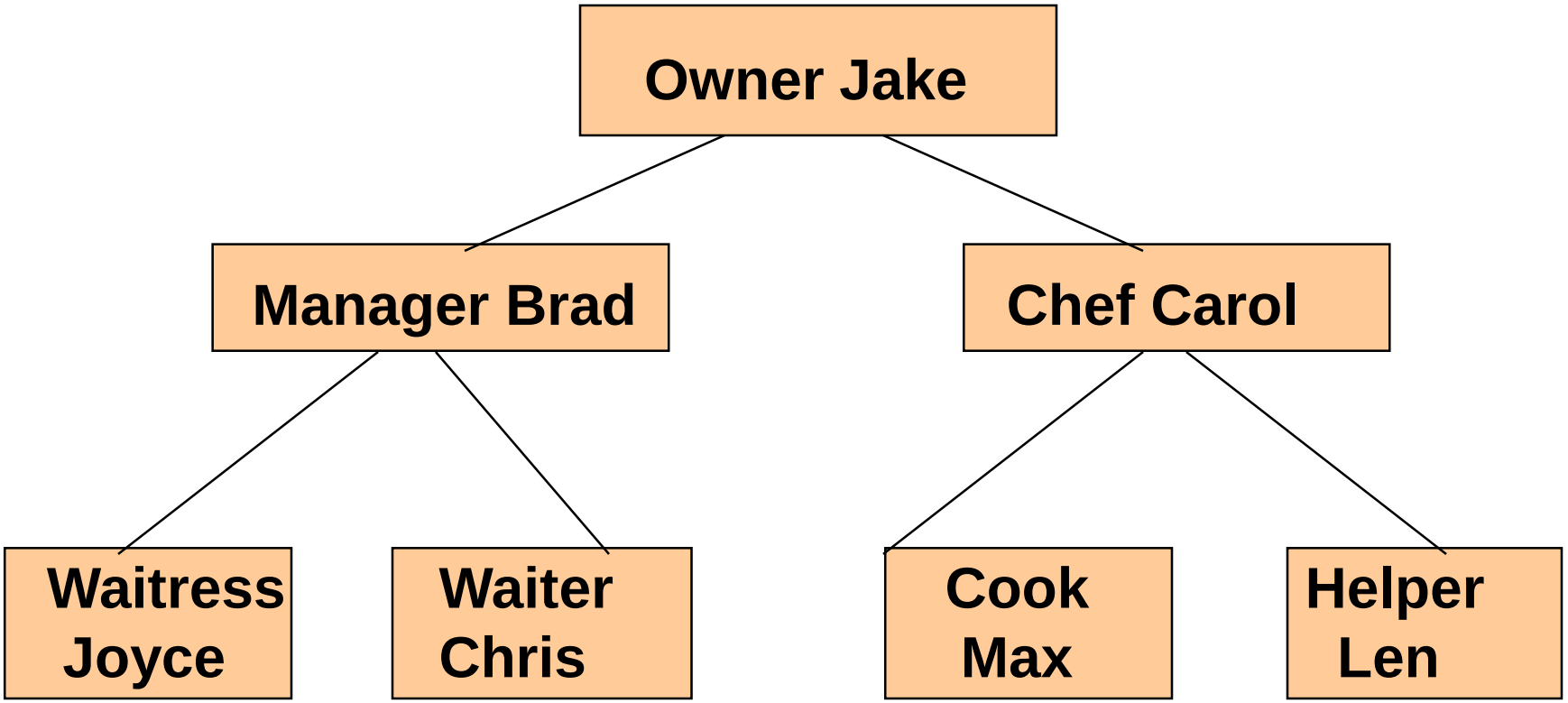
有序根树 Ordered Rooted Tree

- **Definition** （有序根树）: An *ordered rooted tree* is a rooted tree where the children of each internal vertex are ordered.

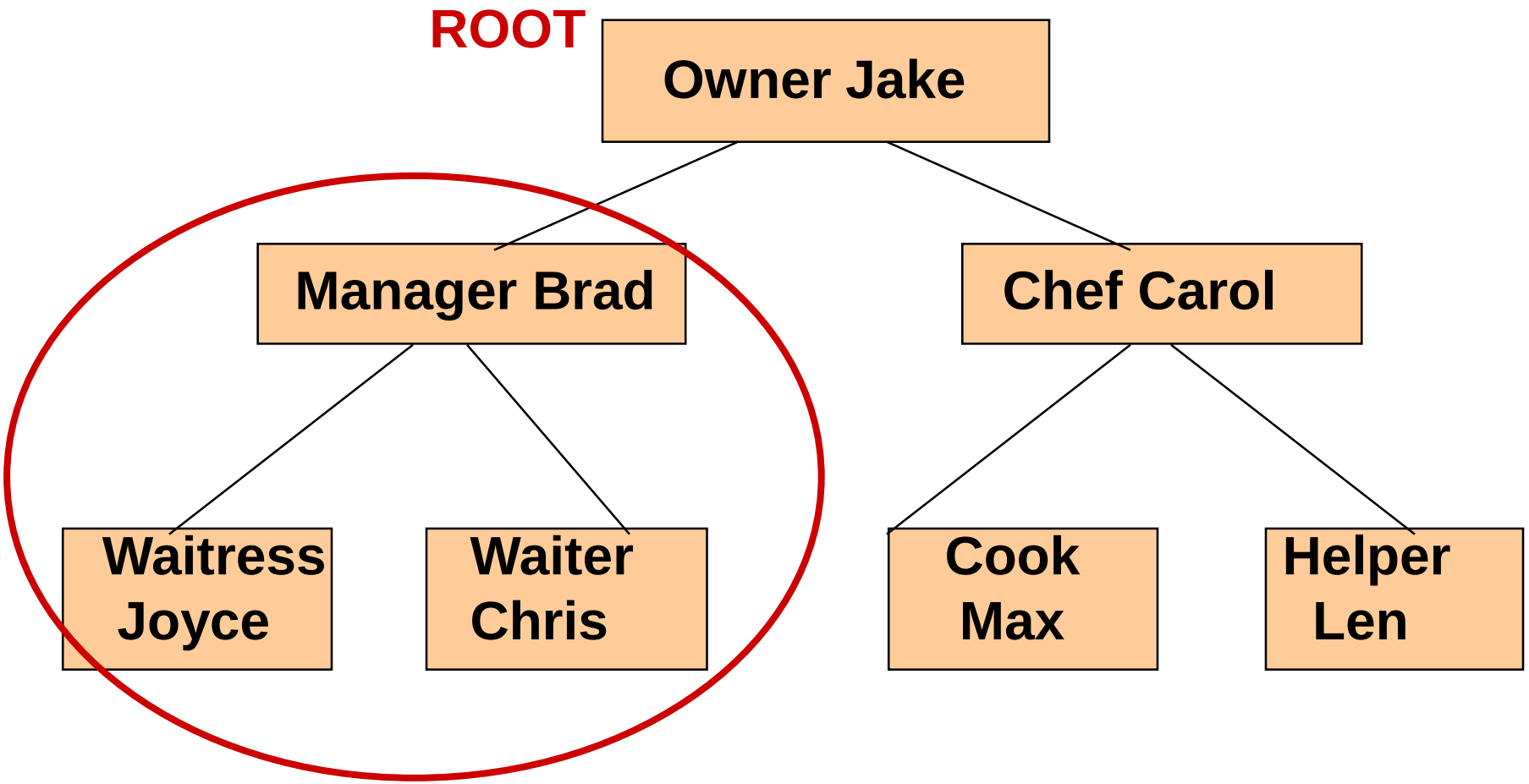
如果将根树的每个内结点的所有子结点都排个序，那这样的根树称为有序根树。

- 数据结构里讲的搜素树都是有序根树
- 例子：家族成员形成的根树（不算配偶）

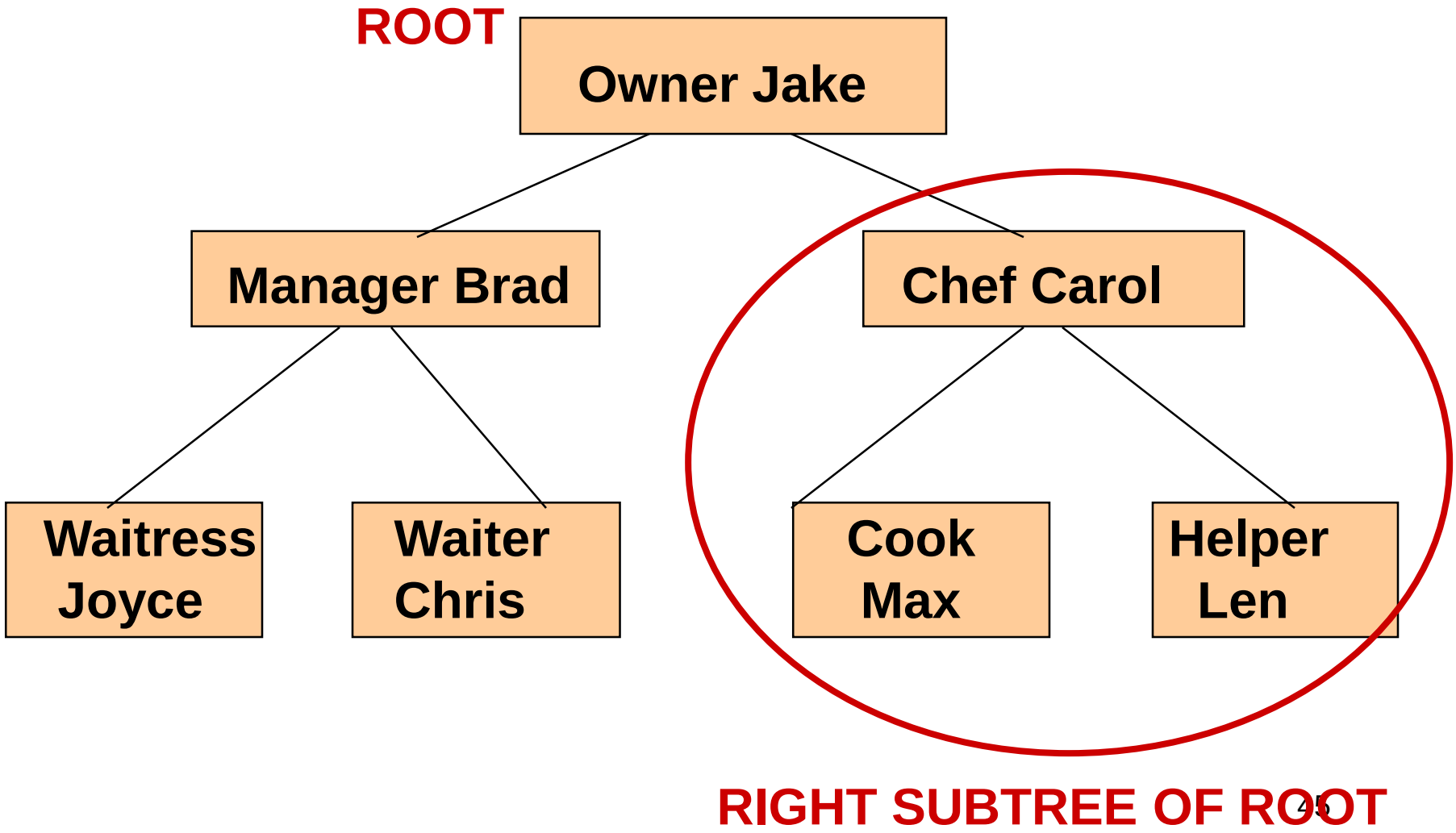
How many internal vertices?



A Subtree of a Ordered Rooted Binary Tree



Another Subtree



m-ary trees m-元树

m-元树(m叉树)：A rooted tree is called a *m-ary tree* if every internal vertex has no more than *m* children.

正则m-元树：The tree is called a *full m-ary tree* if every internal vertex has exactly *m* children.（正则m-元树，教材里翻译**成满m-元树**）

(complete m-ary tree 完全正则m元树)：如果一颗正则m-元根树的所有叶结点都处于同一层。（教材中完全m元树）

二元树（二叉树）： An *m*-ary tree with *m*=2 is called a *binary tree*.

注意：“满m-树、完全m-树”这几个名词在不同书之间的差异

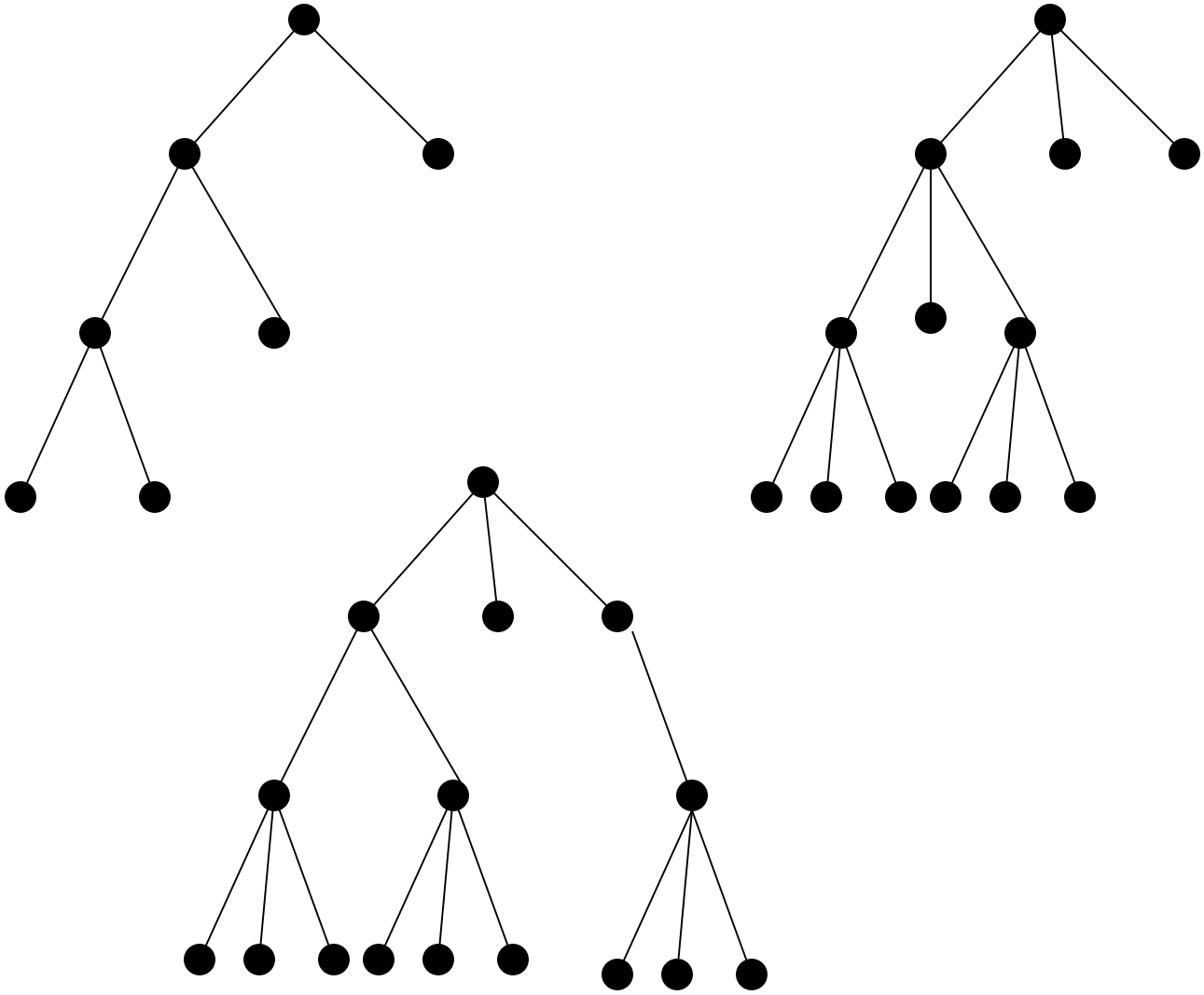
Binary Tree(二元树,二叉树)

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(正则二元树) if every internal vertex has exactly 2 children.

- Ordered Binary Tree:有序二叉树的每个内结点的子结点分为左子结点和右子结点

m元树的例子



Tree Properties 根树的一些数量特征

Theorem 3：A full m-ary tree with k internal vertices contains n=mk+1 vertices.（正则m元树，满m元树）

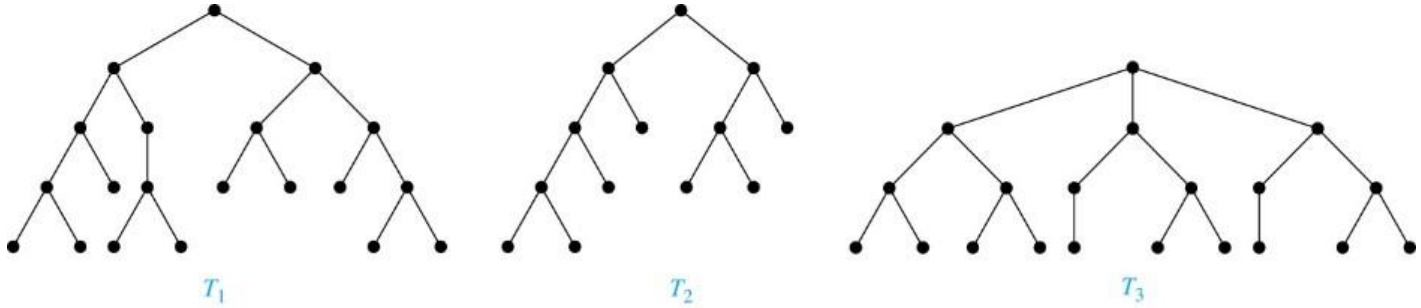
Proof: Every vertex (but root) is a sub vertex of an internal vertex. There are mk sub-vertices (total), plus the root which is not a sub vertex of any vertex.

Theorem 4. There are at most 2^H leaves in a binary tree of height H. 高度为H的m元树顶多有m^H个叶结点。

Balanced *m*-Ary Trees

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

Example: Which of the rooted trees shown below is balanced?



Solution: T_1 and T_3 are balanced, but T_2 is not because it has leaves at levels 2, 3, and 4.

推论： If a m-ary tree with L leaves, then its height $h \geq \lceil \log_m L \rceil$. If it is full and balanced(正则且平衡的), then its height is $H = \lceil \log_m L \rceil$.

Summary—the most important properties
树的重要性质总结

- 1. Connected 连通
- 2. no simple circuit 无简单回路
- 3. n=m+1 (or m=n-1), where n is the number of vertices, m number of edges
- 4. There is an unique simple path between any two distinct vertices. 任意两个不同的结点之间存在唯一真路（简单路）
- 5. Start from any vertex of a tree, a rooted (directed) tree can be constructed 从任何一点出发都可以形成一颗根树
- 6. 根树是有向树，根到任一个非根结点有唯一的一条有向路。
- 7. 根树的内结点到其任一子孙结点都有唯一的有向路

思考问题

- 如果一个n个结点的连通图，恰好有n-1条边，是否这个图一定是树？
- 定理：一个(n,m)连通图是树当且仅当n=m+1
- 如果是，能否以这两个条件作为树的定义？也就是说与树的定义的两个条件等价？
- 如果是n个结点的树林，那应该会有多少条边？

More questions about tree

- Question 1: what is the minimum number of edges of a connected undirected simple graph with n vertices?
  n个结点的一个连通简单无向图的至少有多少条边？

- Question 2: if A is the adjacency matrix of an undirected simple graph G, can you work out a way to know whether G is a tree or not? How? 如何利用邻接矩阵判断一个简单无向图是否是一颗树？

- Question 3: which vertices in a tree are cut vertices? Which edges are cut edges? 树中哪些边是割边，哪些是割点？

- Question 4: is there any circuit in a tree? 树中有回路吗？

Exercises

- 7.1节 T11(1)
- 7.1节 T19，T21 , T31