Discrete Mathematics

Alexander Pasko, Evgenii Maltsev, Dmitry Popov

www.pasko.org/ap/DM

Trees

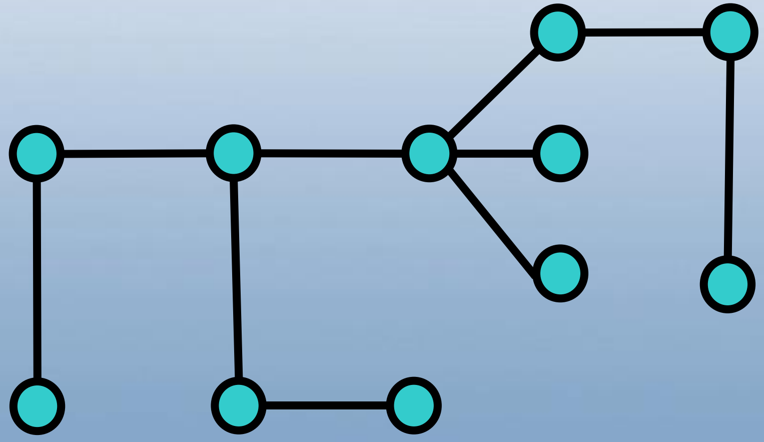
Image from "Wings of Wax“,
Nederlands Dans Theater

Contents

- Using tree structures
- Notion of tree structure
- Trees terminology
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

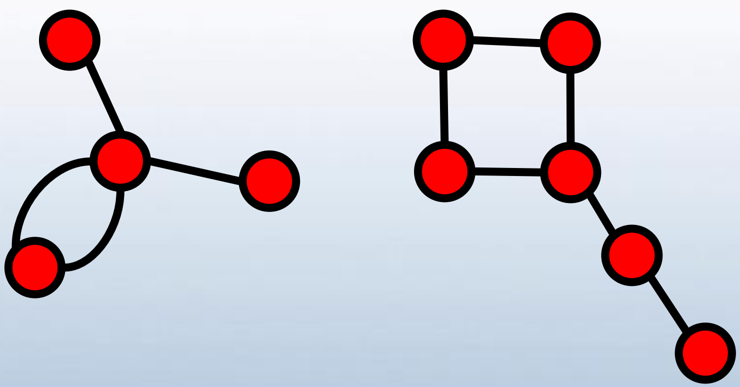
Tree Structure

- Tree is a connected undirected graph with no circuits.
- Tree is a connected graph with n-1 edges
- Tree is a graph such that there is a unique simple path between any pair of vertices

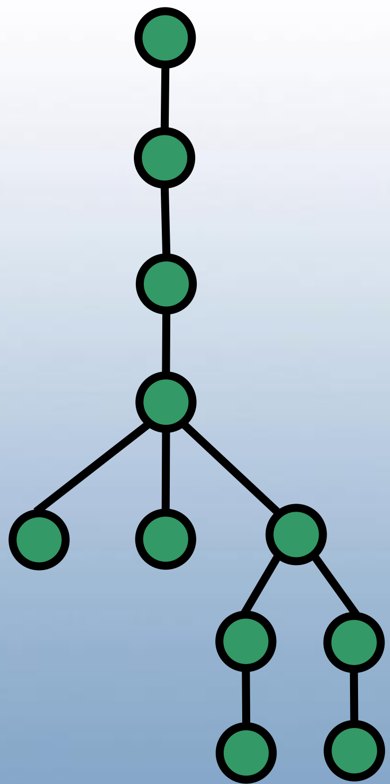


Tree structure?

Not trees

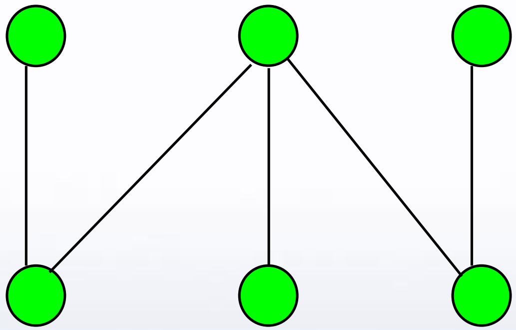


Tree

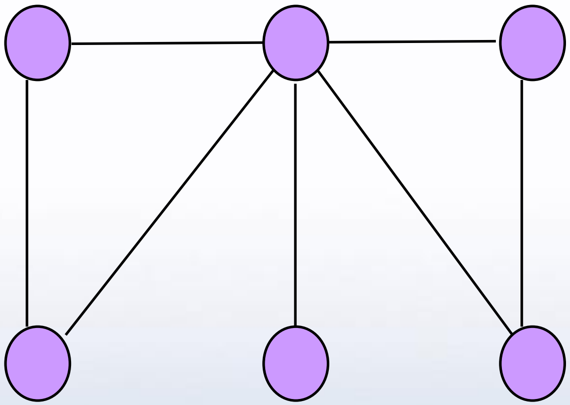


Which Graphs are Trees?

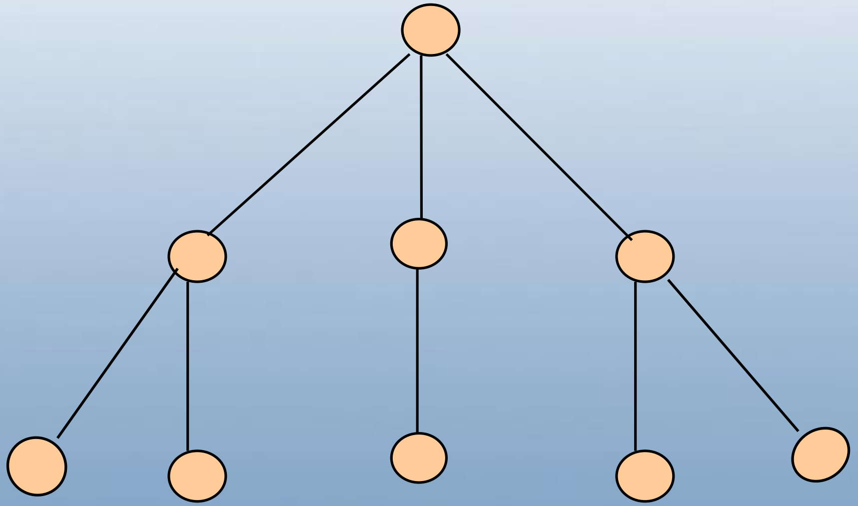
a)



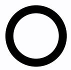
b)



c)

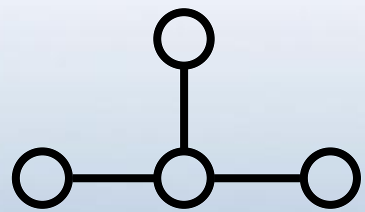


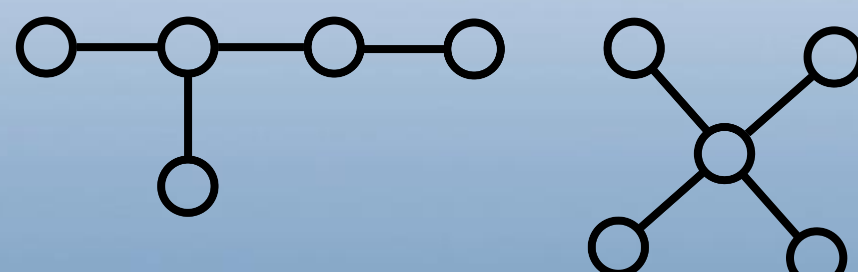
How Many n-Node Trees?

1: 

2: 

3: 

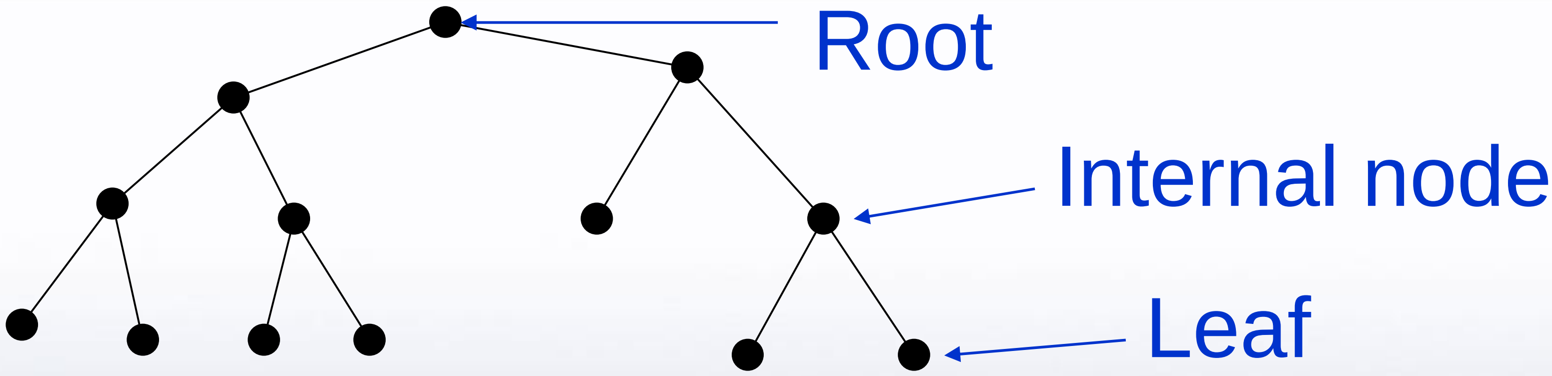
4:  

5:  

…

8: 23 of them

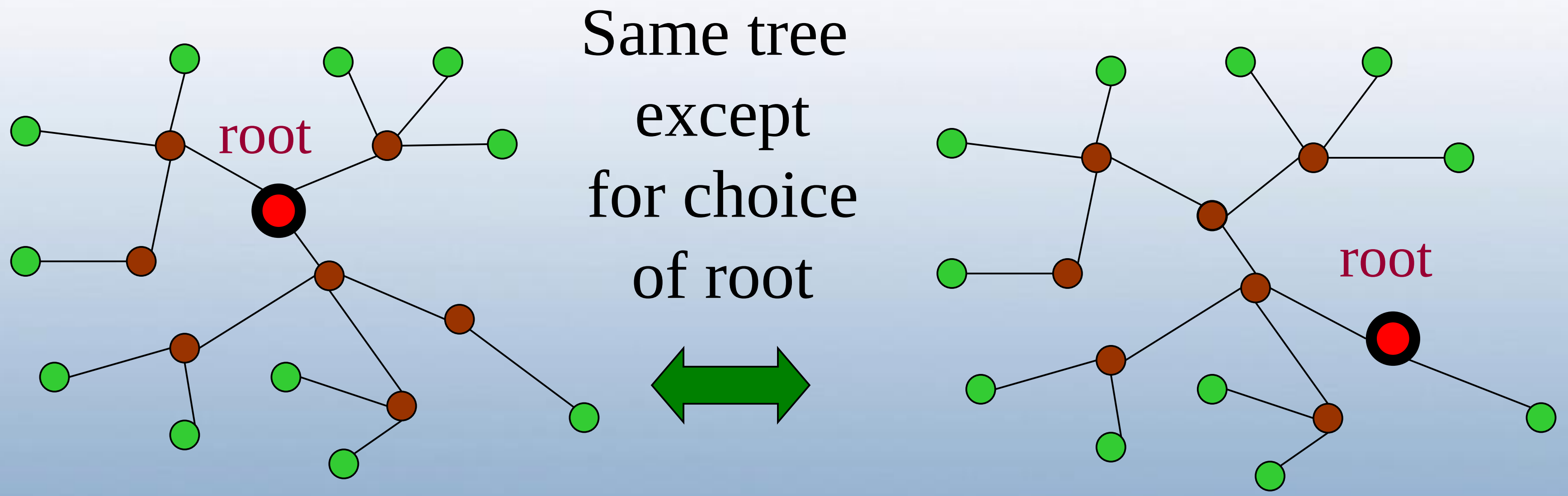
Rooted Tree



- A rooted tree has one vertex designated as root and every other edge is directed away from the root (we put the root at the top)
- Leaf node in a tree is any pendant or isolated vertex of degree 1
- Internal node is any non-leaf vertex

Number of Rooted Trees

Given unrooted tree with n nodes yields n different rooted trees.



Contents

- Using tree structures
- Notion of tree structure
- **Trees terminology**
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

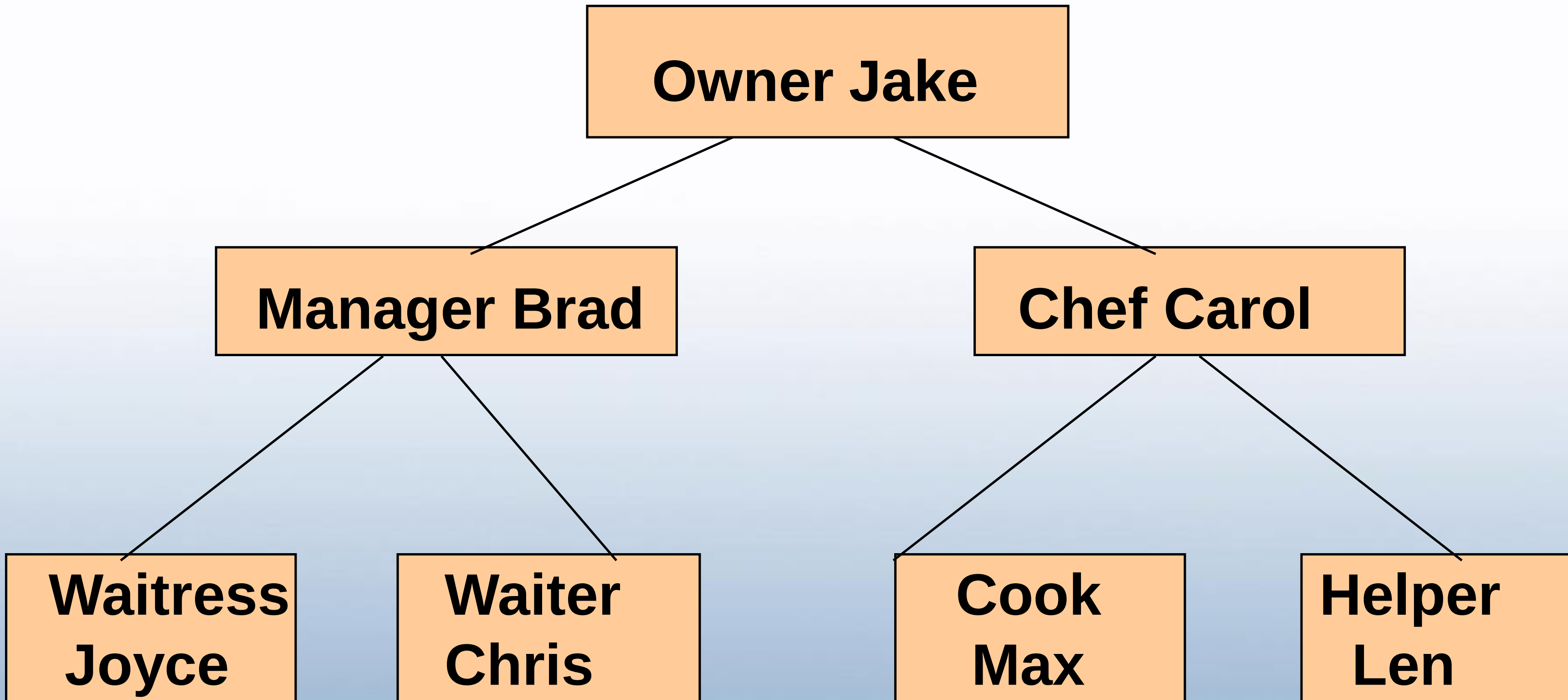
Tree Terminology

- Parent node is adjacent to the *child* node and placed above it in the rooted tree
- The ancestors of a non-root vertex are all the vertices in the path from root to this vertex
- Root has no ancestors
- The descendants of vertex v are all the vertices that have v as an ancestor
- Leaf nodes have no children
- Internal node is a node that has children
- Sibling nodes have the same parent

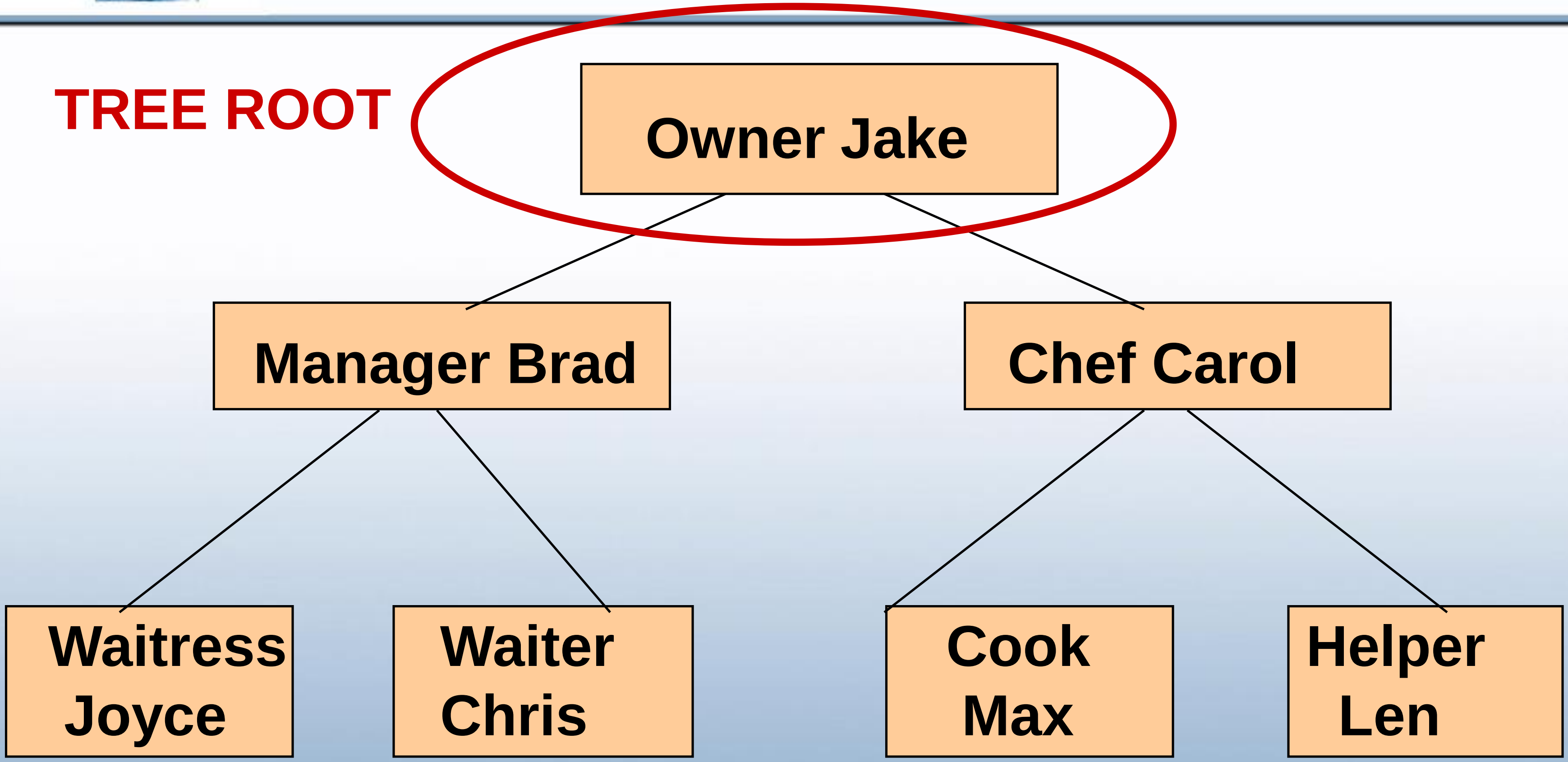
Tree Terminology

- Level of vertex v in a rooted tree is the length of the unique path from the root to v
- Height of a rooted tree is the maximum of the levels of its vertices
- Subtree at vertex v is a subgraph of the tree consisting of vertex v and its descendants and all edges incident to those descendants.

Example: Company Tree

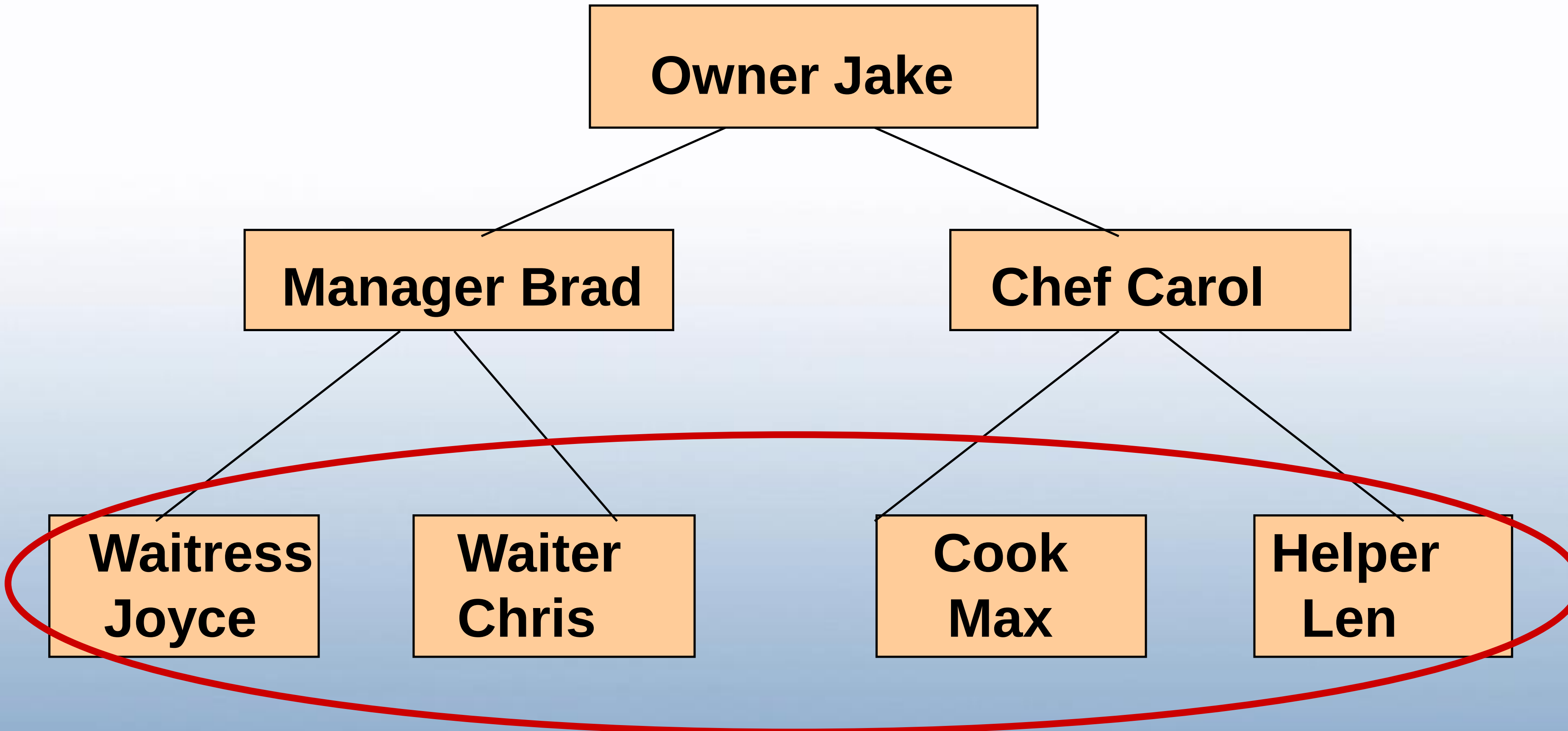


Root



Root has no ancestors.

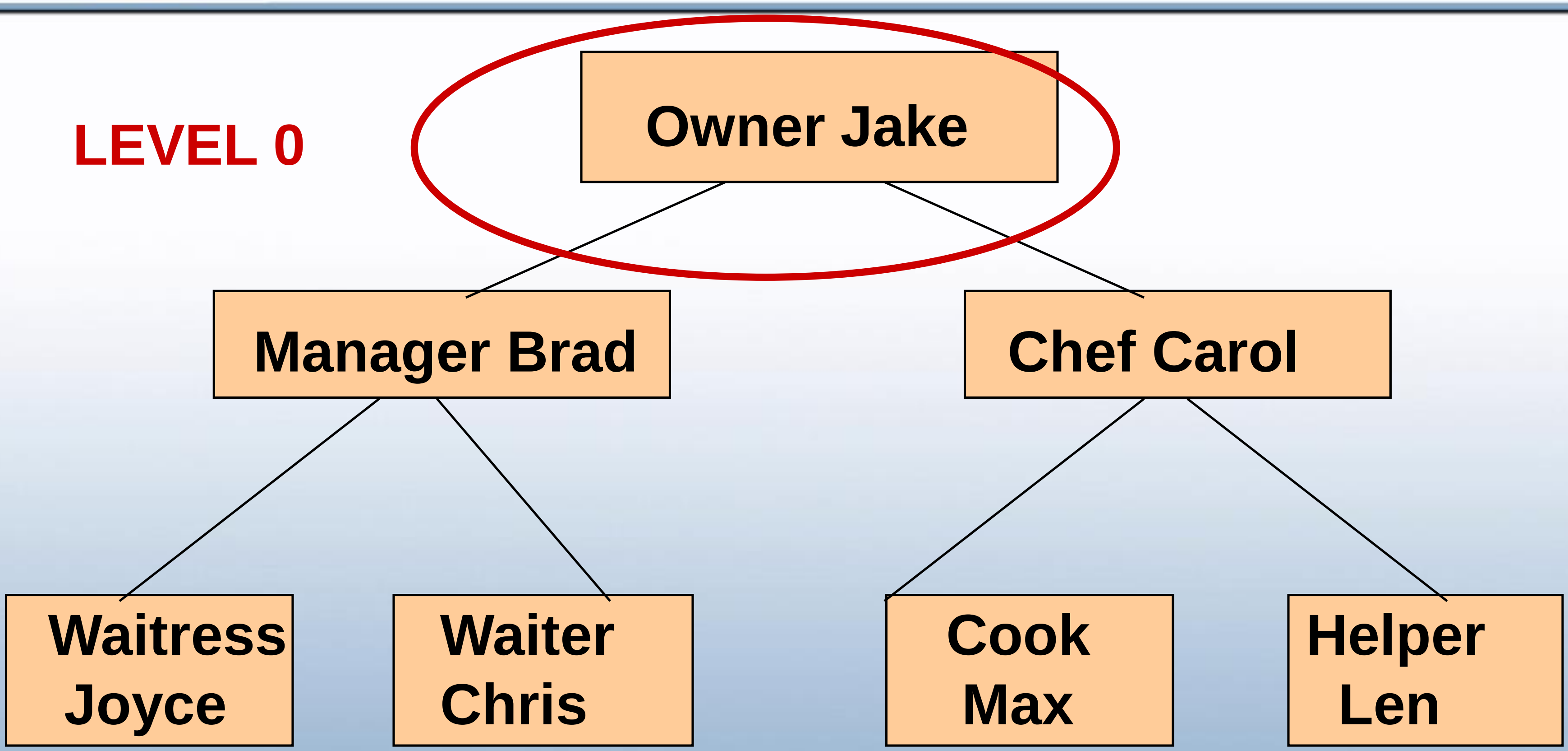
Leaves



LEAF NODES

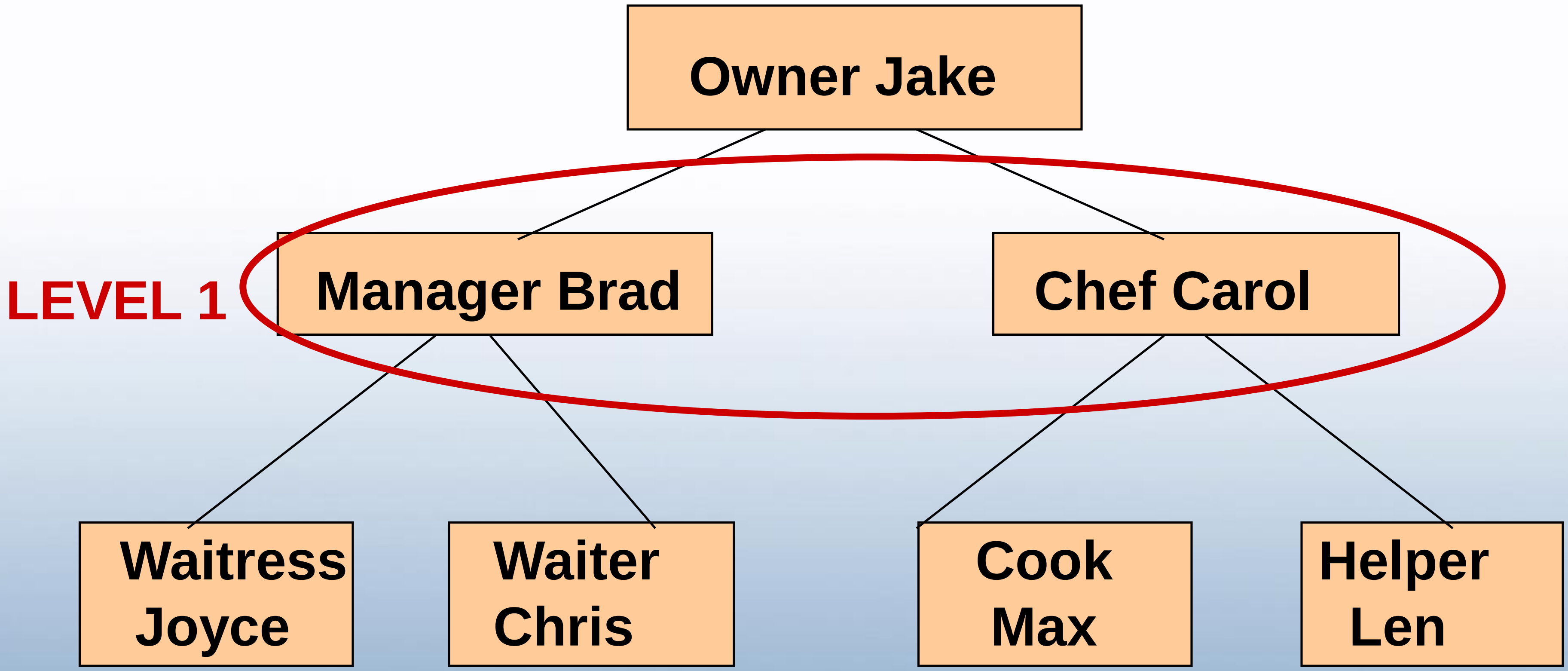
Leaf nodes have no children.

Tree Has Levels



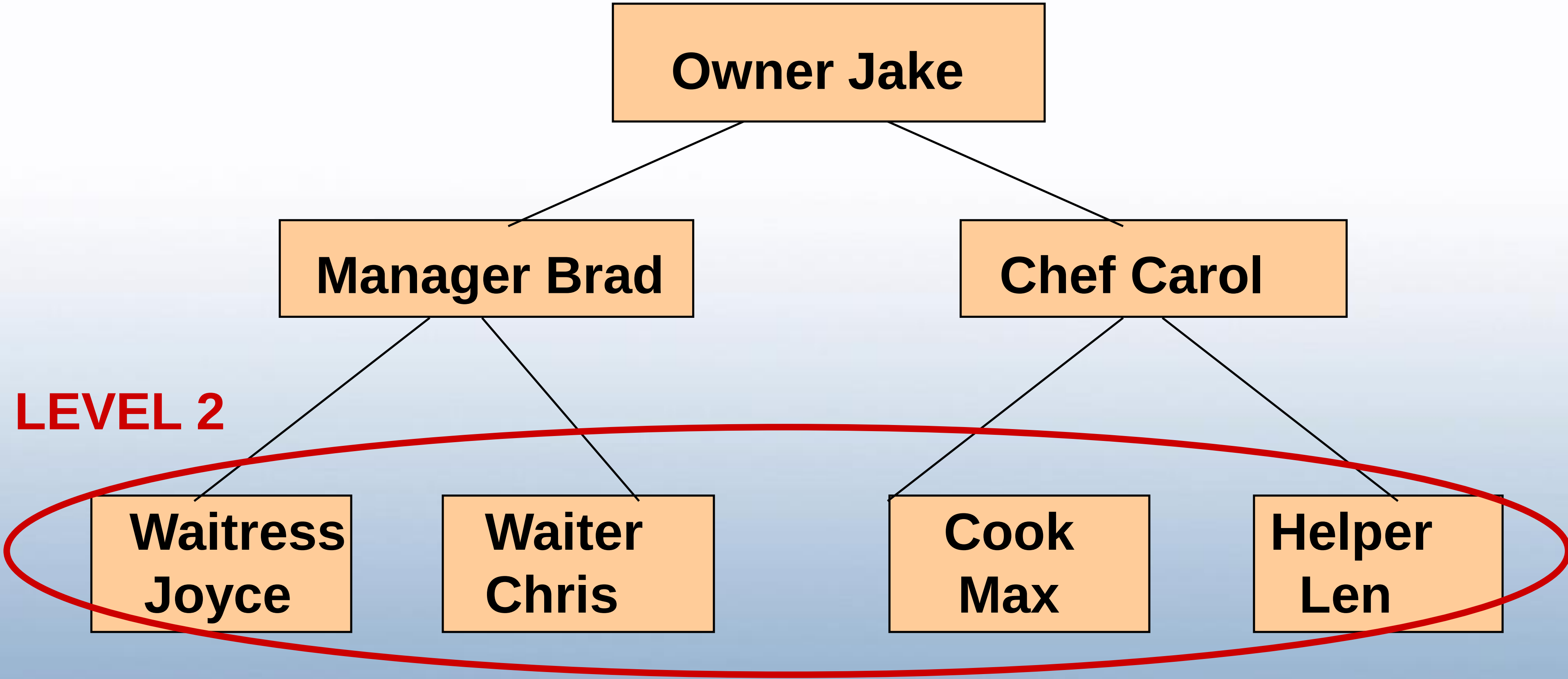
Level of vertex v in a rooted tree is the length of the unique path from the root to v

Level One



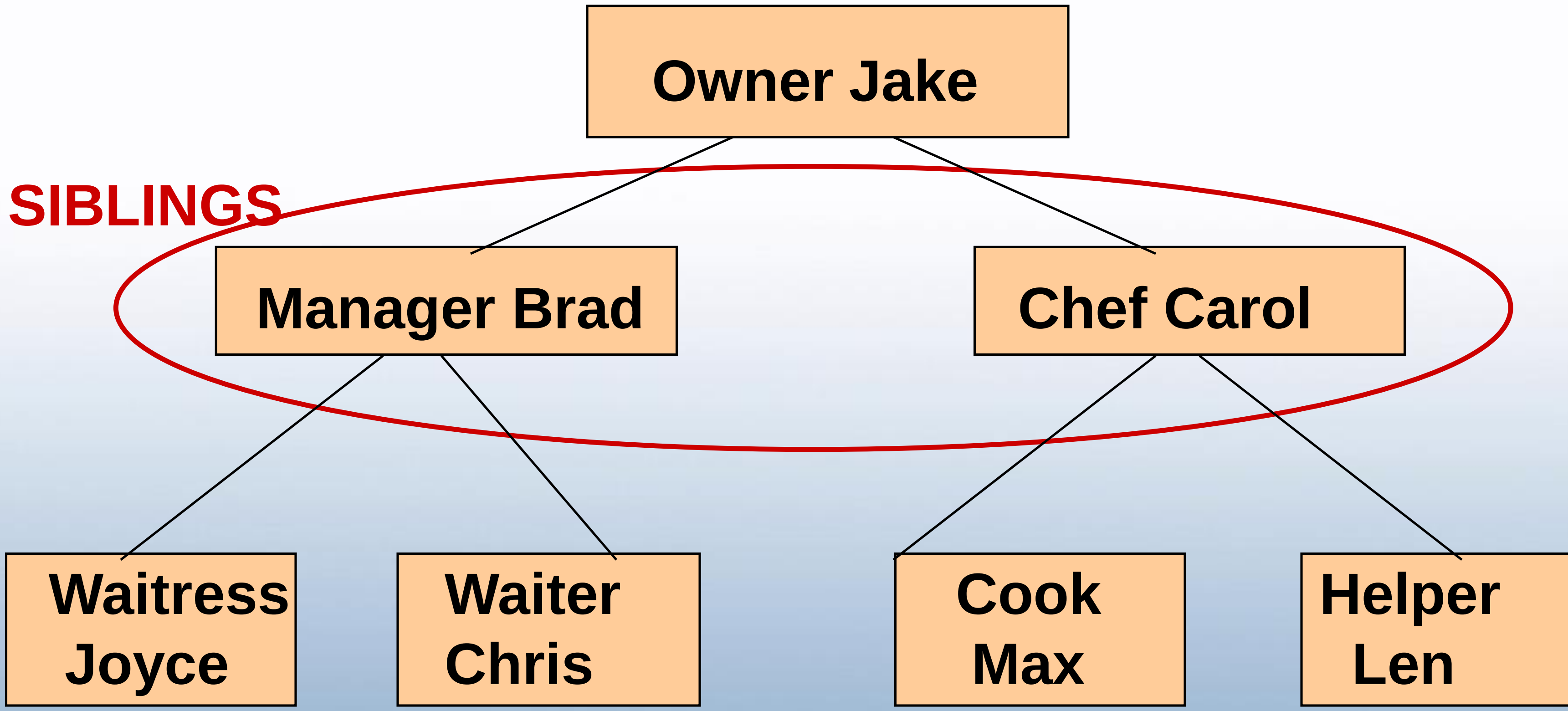
Level of vertex v in a rooted tree is the length of the unique path from the root to v

Level Two



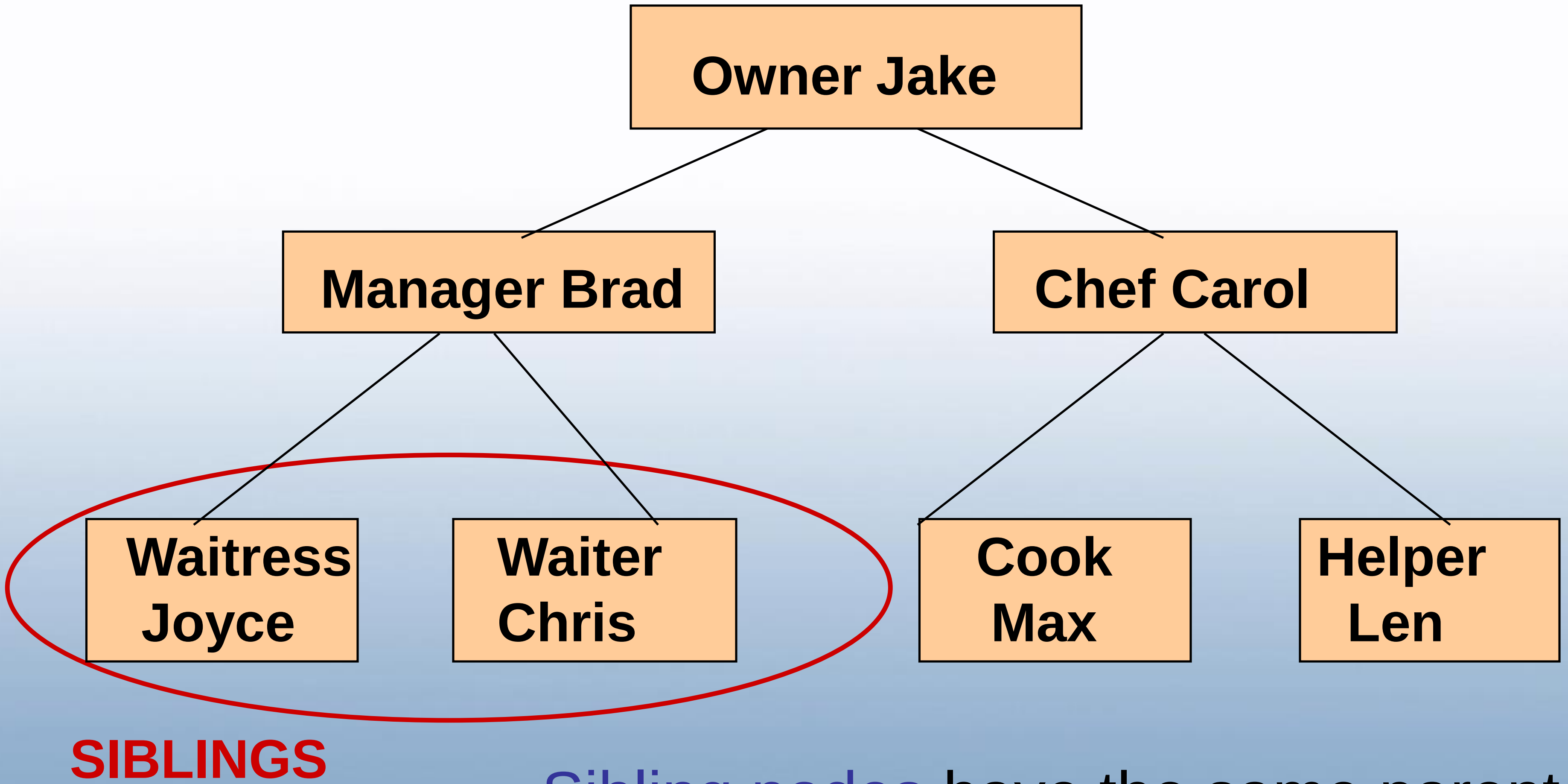
Level of vertex v in a rooted tree is the length of the unique path from the root to v

Siblings



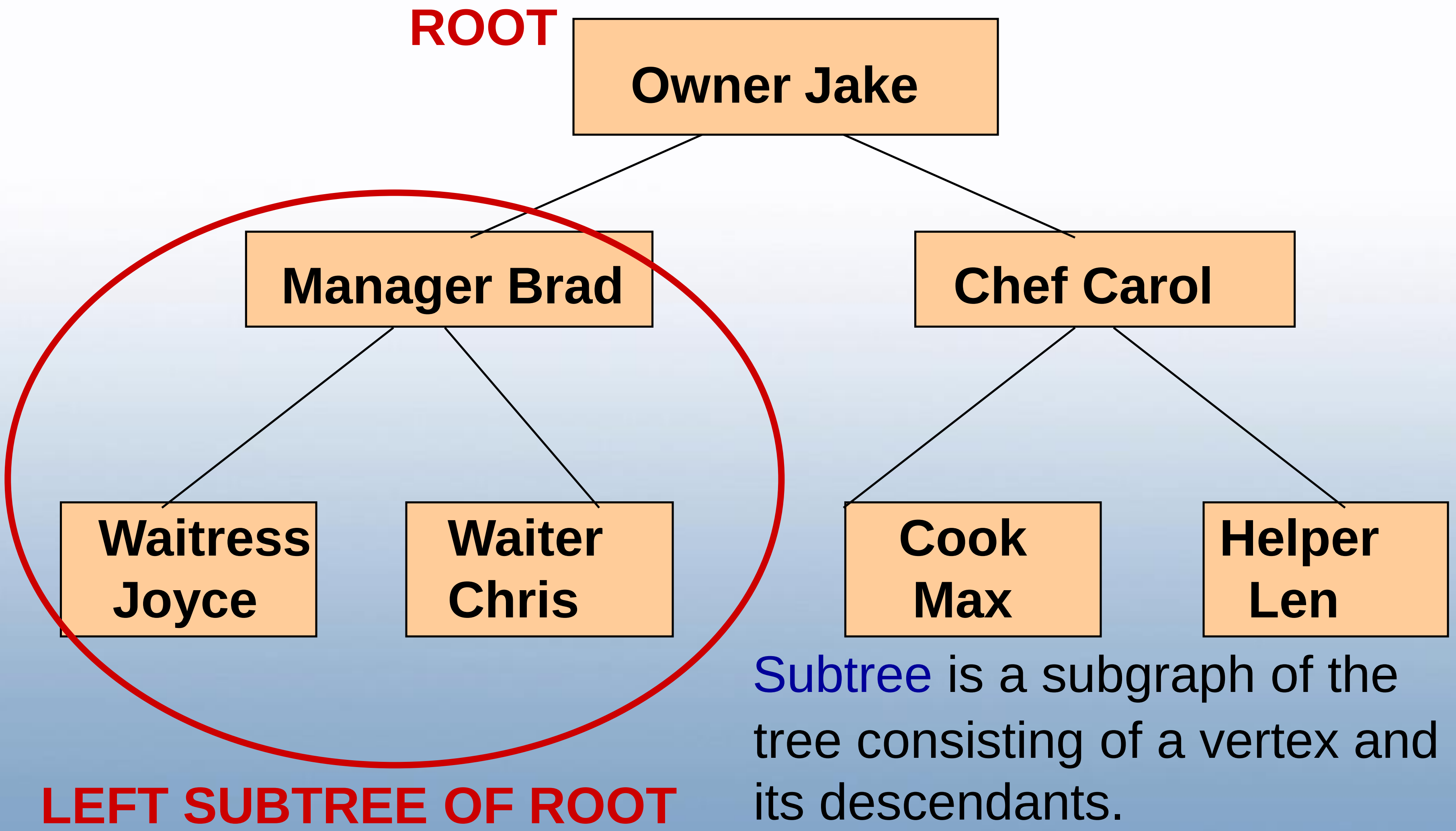
Sibling nodes have the same parent.

Siblings



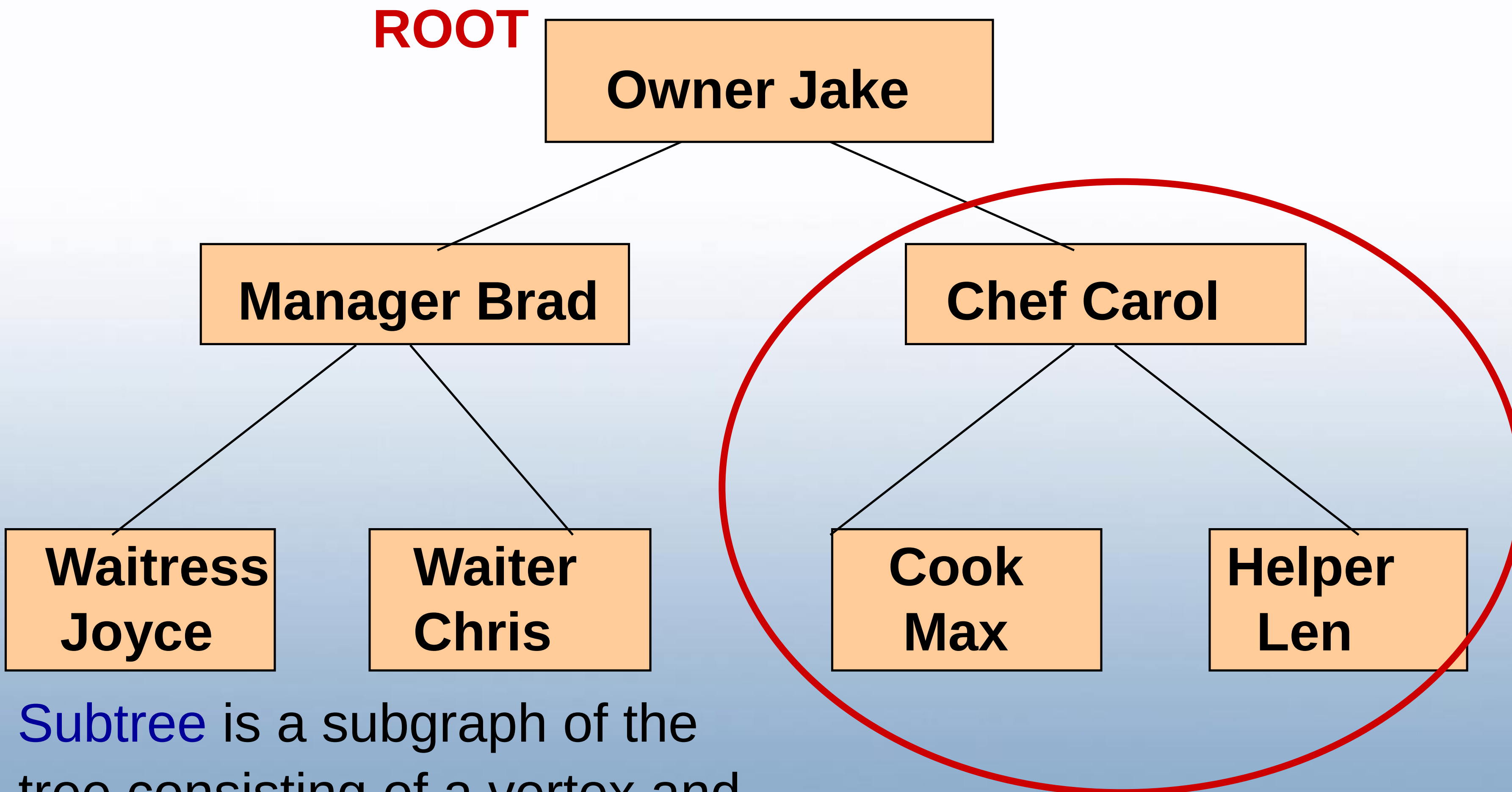
Sibling nodes have the same parent.

Left Subtree



Subtree is a subgraph of the tree consisting of a vertex and its descendants.

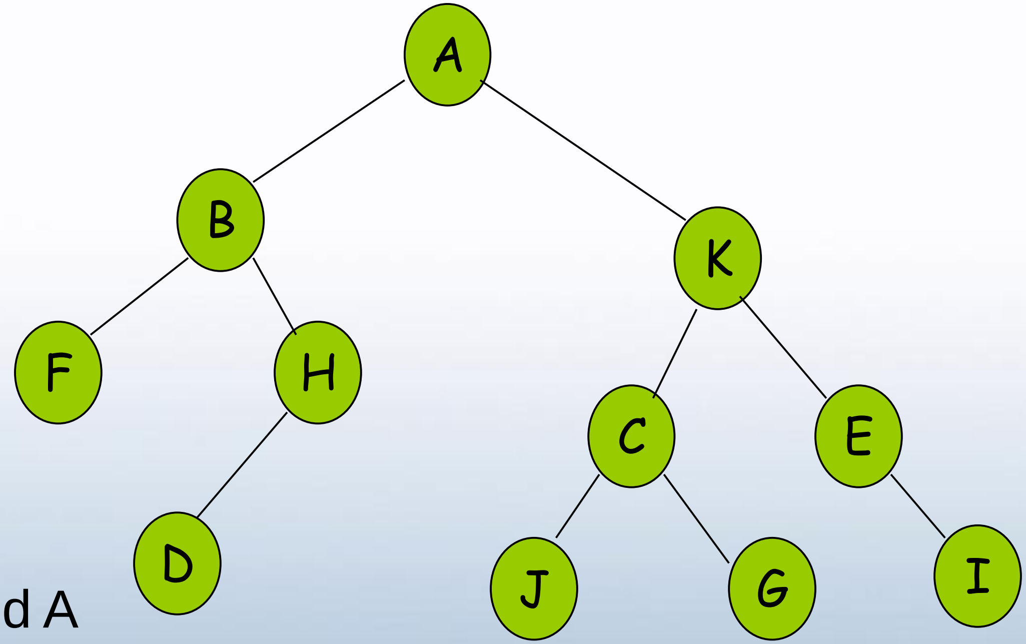
Right Subtree



RIGHT SUBTREE OF ROOT

Subtree is a subgraph of the tree consisting of a vertex and its descendants.

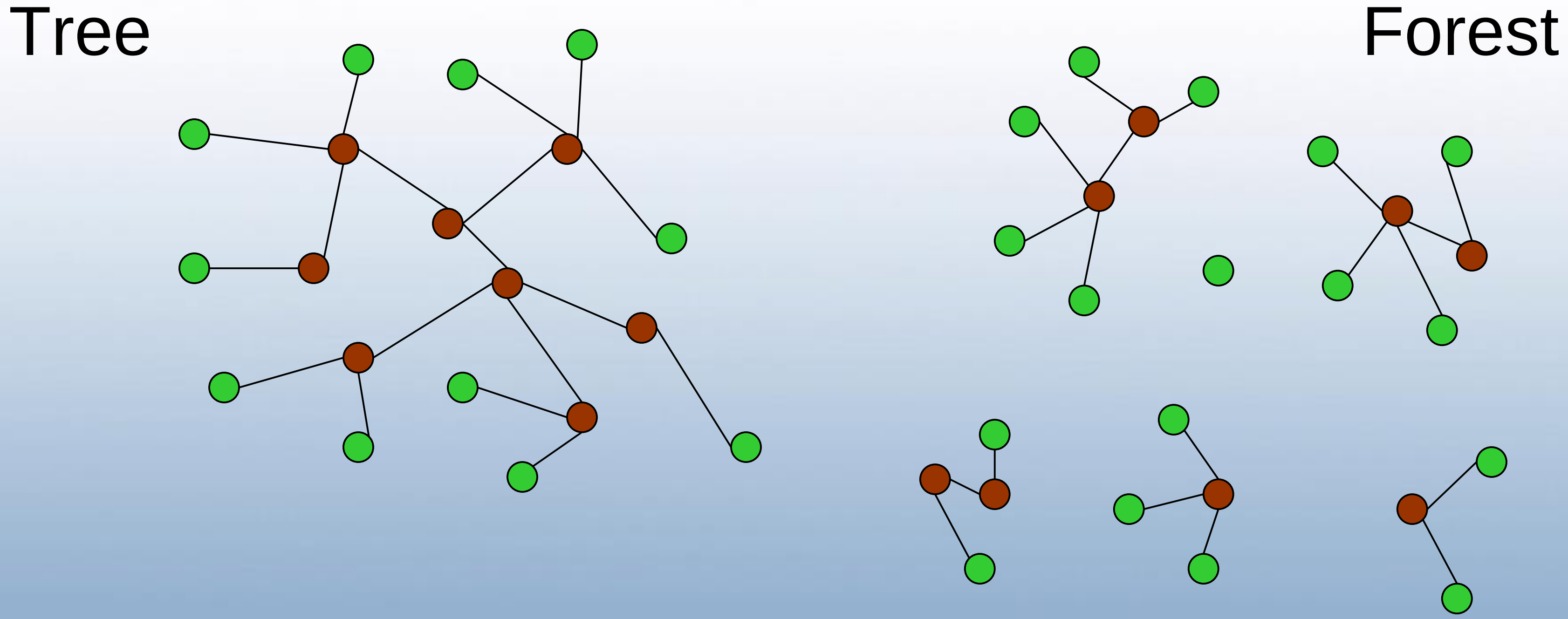
Tree Terminology Summary



- The parent of H is B
- C is a child of K
- The ancestors of I are E, K, and A
- The descendants of B are F, H, and D
- A is the root, and has no ancestors
- K and its descendants make a subtree
- The sibling of G is J
- The leaf nodes have no children: F,D,J,G,I

Tree and Forest

A not-necessarily-connected undirected graph without simple circuits is called a forest.



Leaves in green, internal nodes in brown.

Contents

- Using tree structures
- Notion of tree structure
- Trees terminology
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

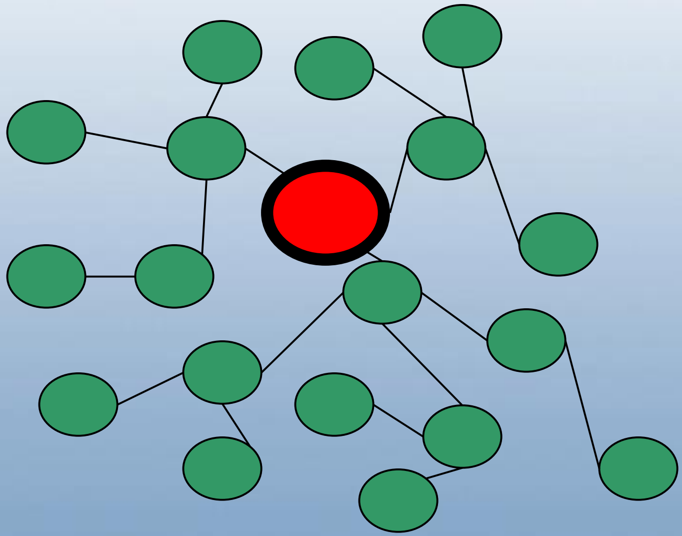
n-ary Trees

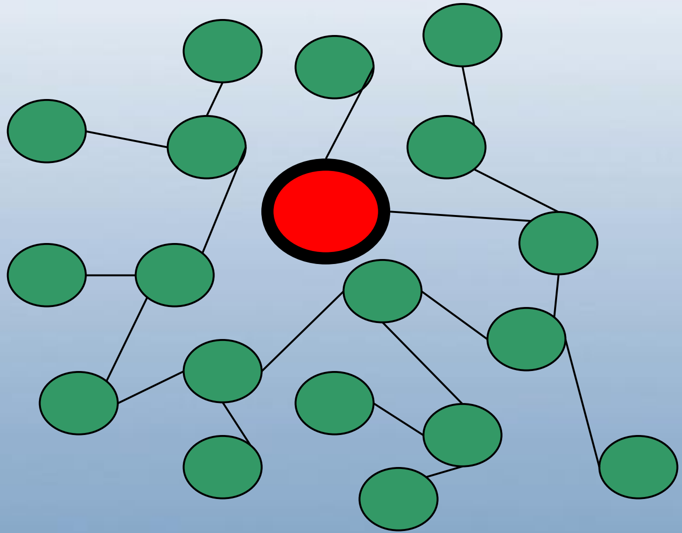
- Rooted tree is called n-ary if every vertex has no more than n children.
- n-ary tree is called full if every internal (non-leaf) vertex has exactly n children.
- 2-ary tree is called binary tree.

  These are handy for describing sequences of yes-no decisions.
- Tree is called full binary tree if every internal vertex has exactly 2 children.

Which Tree is Binary?

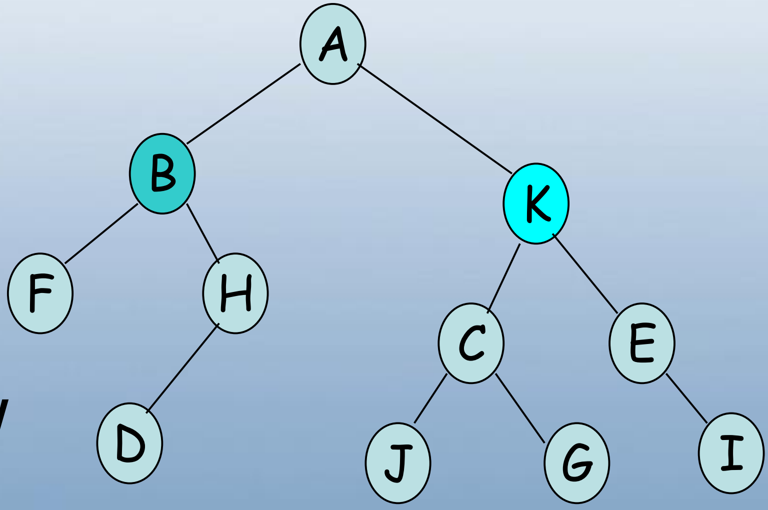
Theorem: A given rooted tree is a binary tree if
every node has degree ≤ 3, and
the root has degree ≤ 2





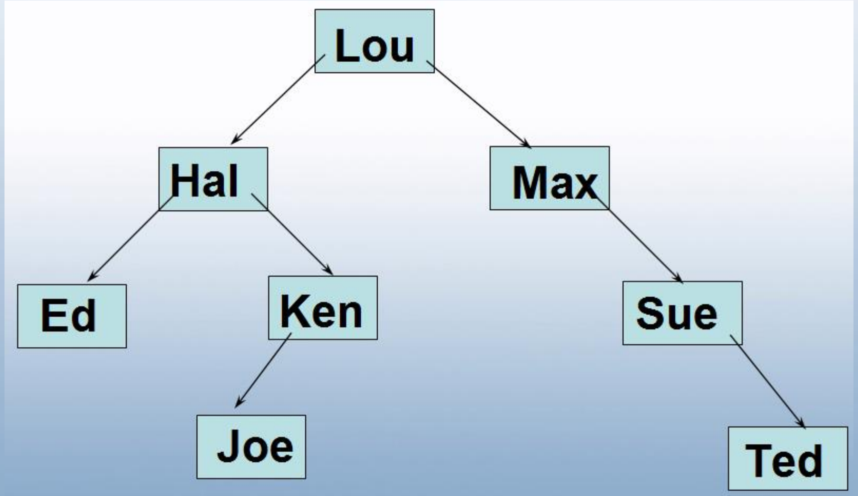
Ordered Rooted Tree

- This is just a rooted tree in which the children of each internal node are ordered.
- In ordered binary trees, we can define:
  - left child, right child
  - left subtree, right subtree
- Example:
  - left subtree: B and below
  - right subtree: K and below



Balanced Binary Tree

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



Binary Tree Properties

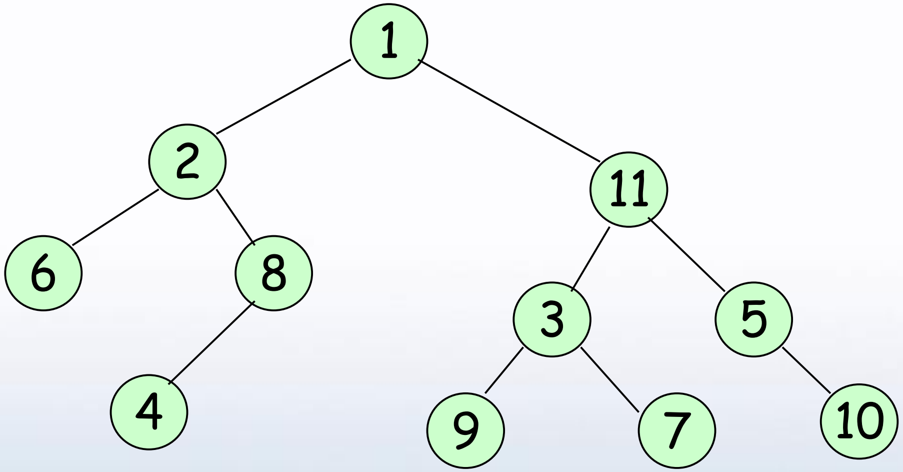
- A tree with N vertices has N-1 edges.
- There are at most 2^H leaves in a binary tree of height H.
- If a binary tree with L leaves is full and balanced, then its height is

$$\mathbf{H = \lceil \log_2 L \rceil}$$

Contents

- Using tree structures
- Notion of tree structure
- Trees terminology
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

Computer Representation of Trees

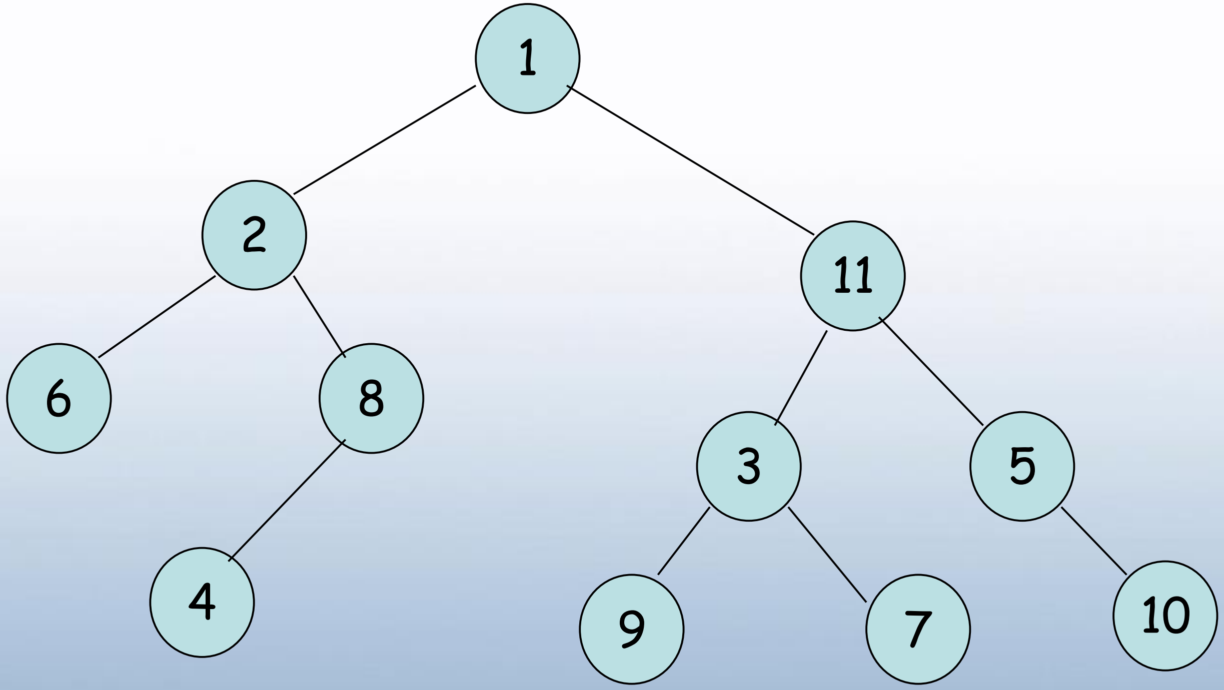


Binary tree data structure with attributes

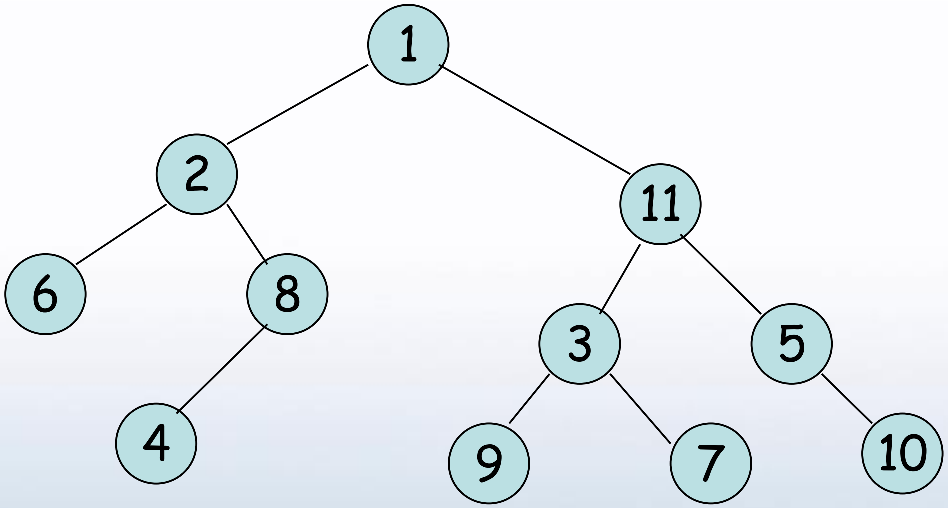
- data
  - the actual information in a node
- left
  - the binary tree to the left, or nil
- right
  - the binary tree to the right, or nil

Computer Representation of Trees

	data	*left*	*right*
1	*d*1	2	11
2	*d*2	6	8
3	*d*3	9	7
4	*d*4	–1	–1
5	*d*5	–1	10
6	*d*6	–1	–1
7	*d*7	–1	–1
8	*d*8	4	–1
9	*d*9	–1	–1
10	*d*10	–1	–1
11	*d*11	3	5



Computer Representation of Trees



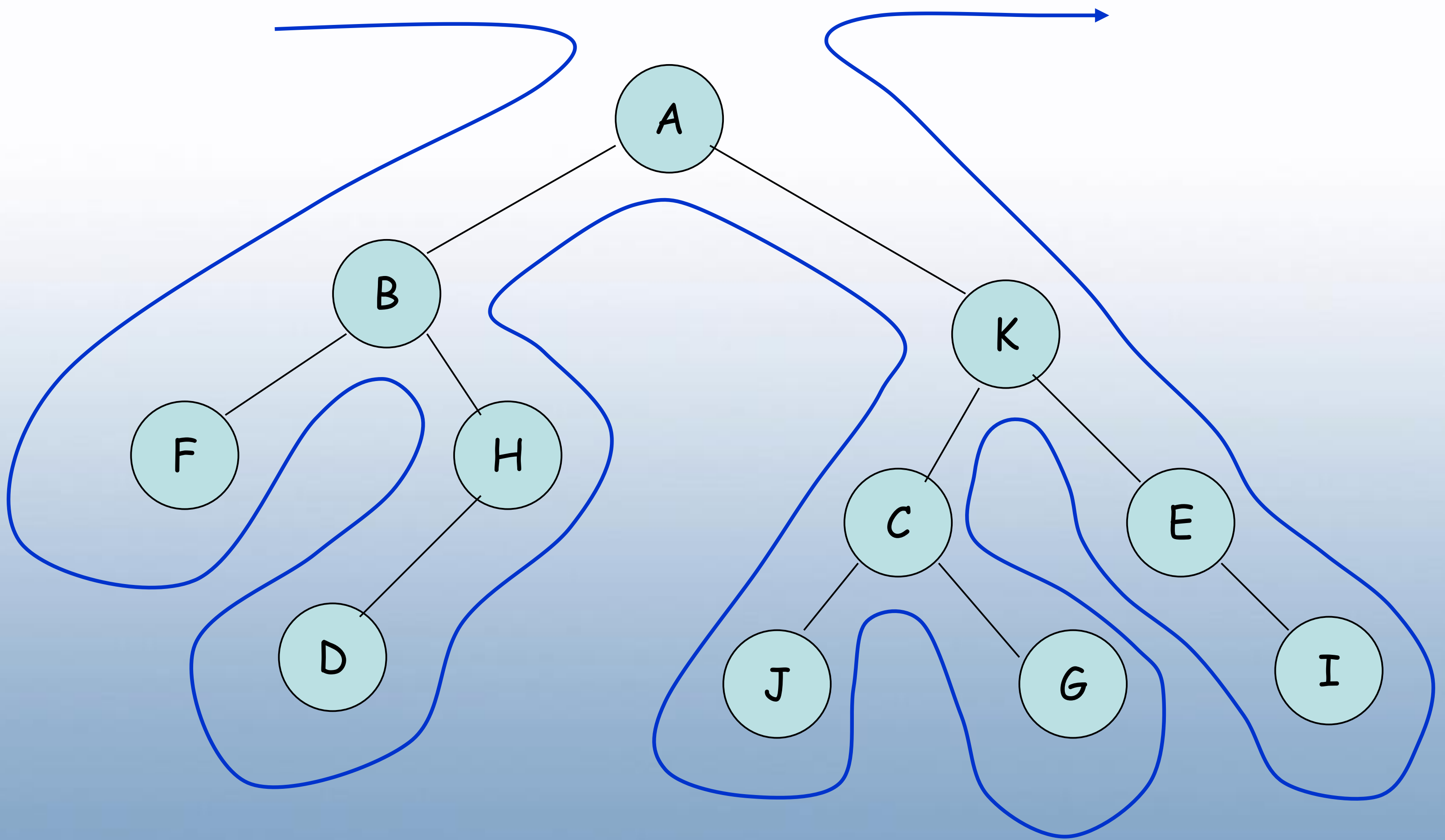
Compact representation by a 1d array, giving parent of a node:

1	2	3	4	5	6	7	8	9	10	11
0	1	11	8	11	2	3	2	3	5	1

Contents

- Using tree structures
- Notion of tree structure
- Trees terminology
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

Tree Traversal



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

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.

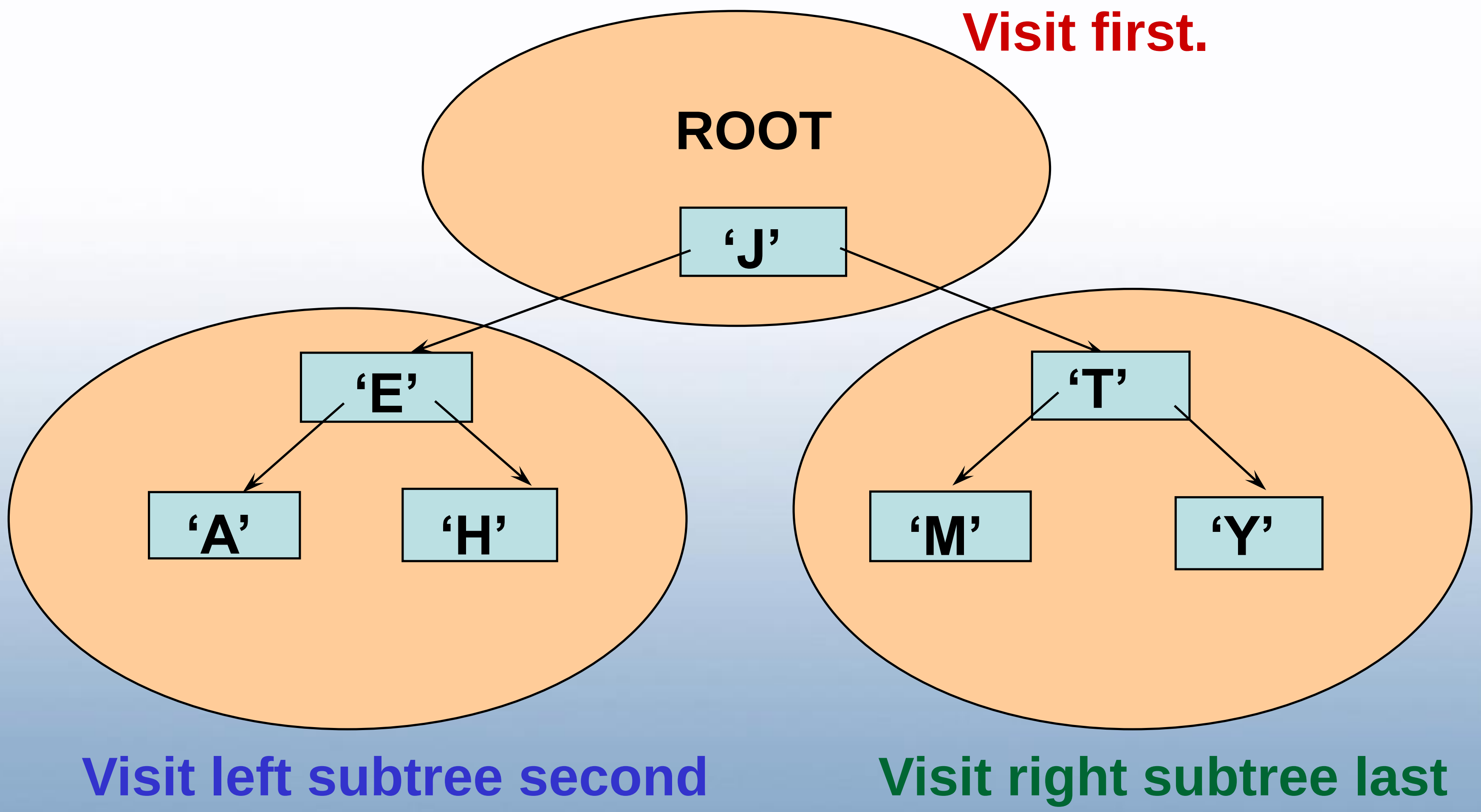
The preorder traversal

1) begins by visiting r

2) traverses T_1 in preorder

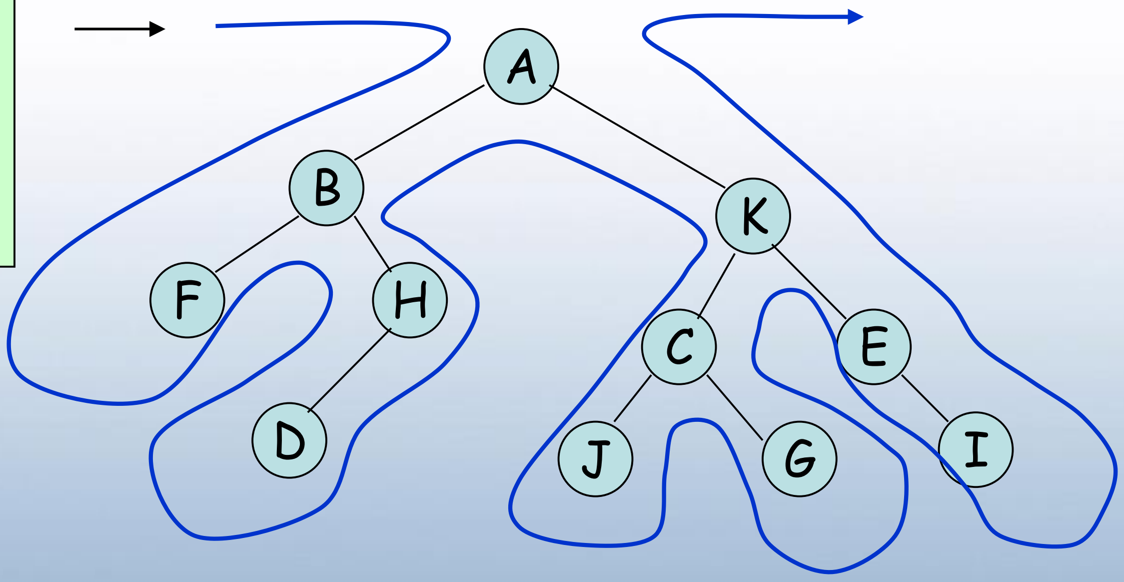
3) traverses T_2 in preorder.

Preorder Traversal:
J E A H T M Y



Preorder Traversal

```
preorder(x)
   if isNode(x)
   then print(x)
 preorder(left(x))
 preorder(right(x))
```



Preorder traversal:

A, B, F, H, D, K, C, J, G, E, I

Inorder Traversal

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.

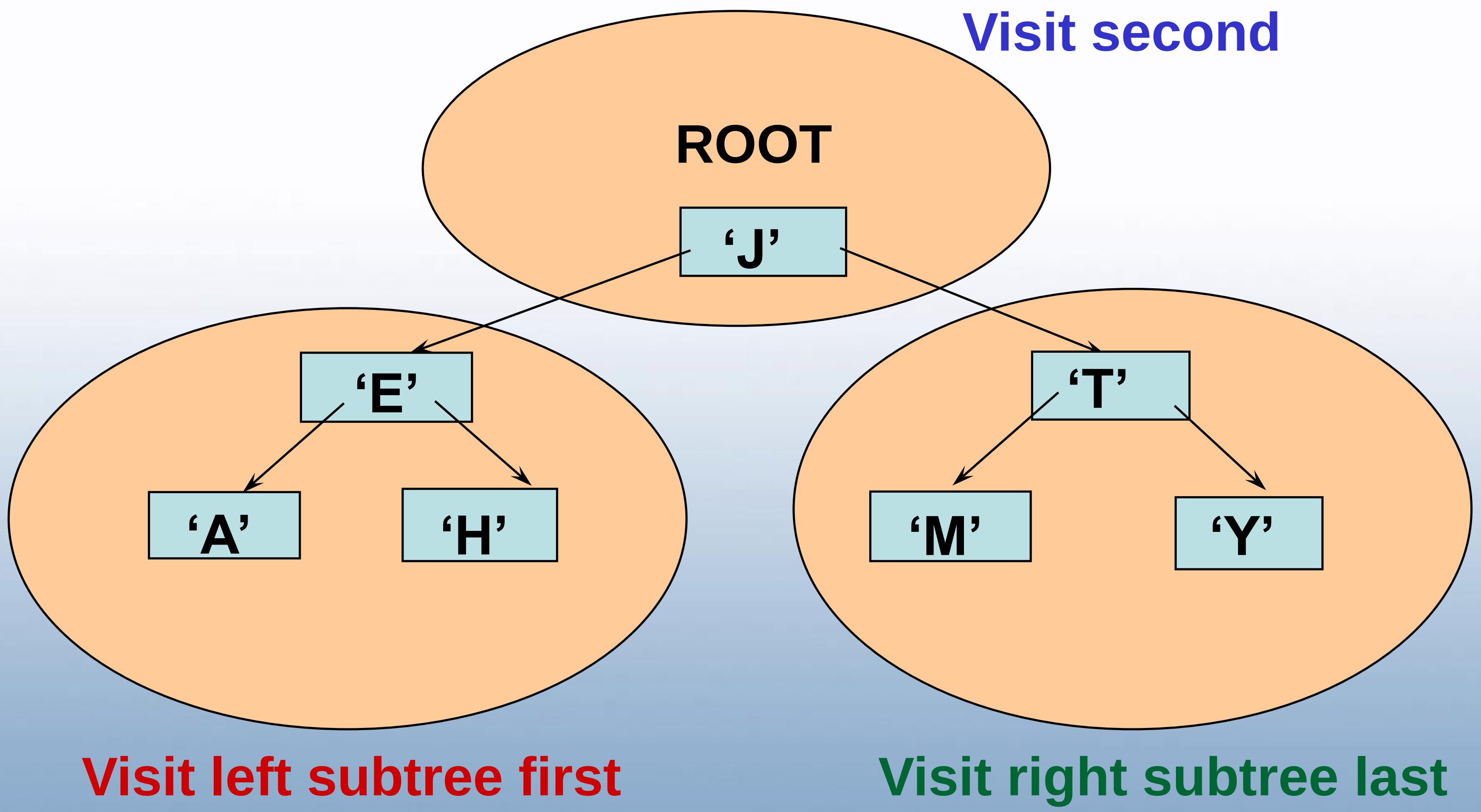
The inorder traversal

1) begins by traversing T_1 in inorder

2) visits r inbetween

3) traverses T_2 in inorder.

Inorder Traversal:
A E H J M T Y



Postorder Traversal

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

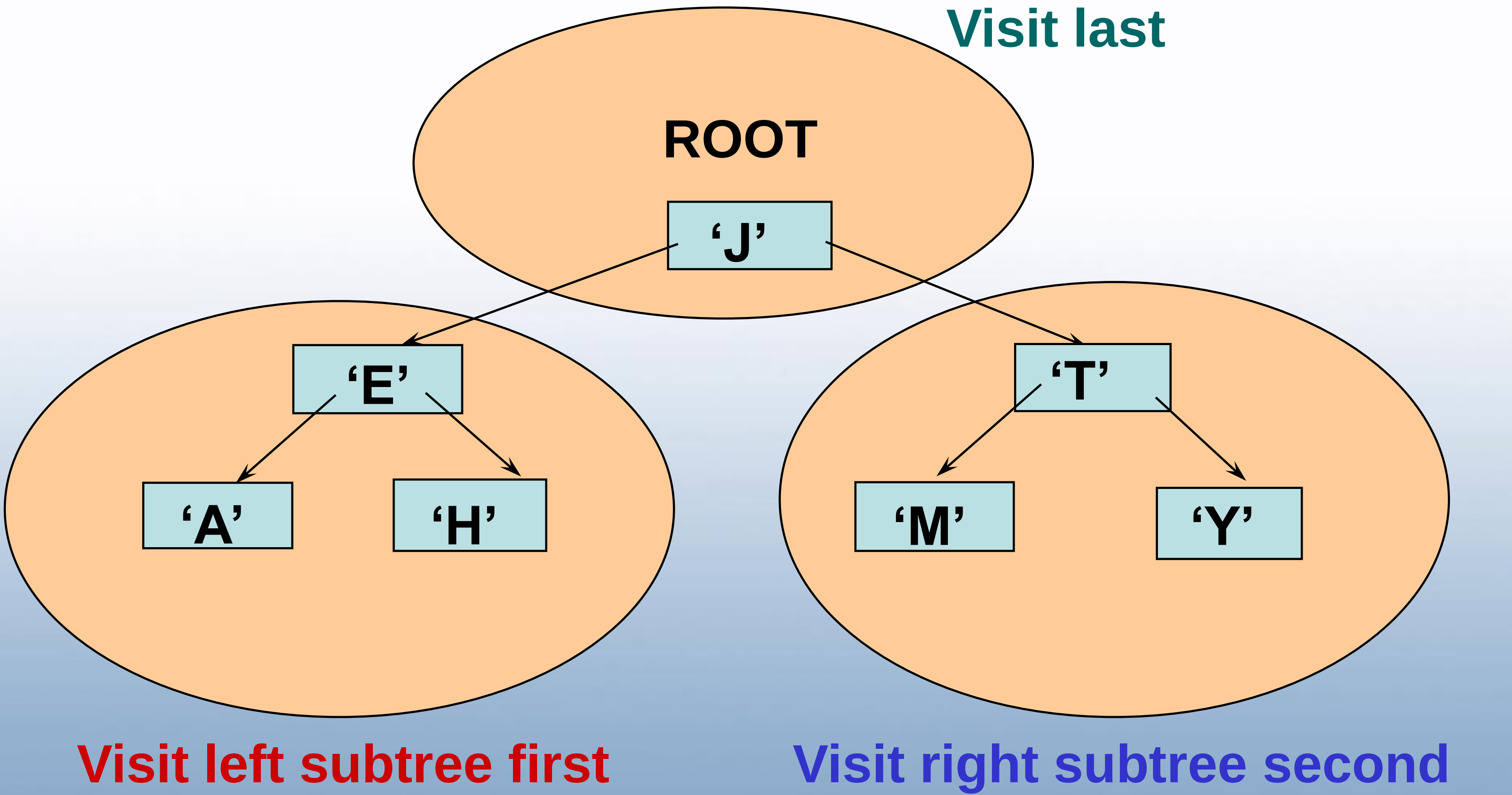
1) begins by traversing T_1 in postorder

2) traverses T_2 in postorder

3) ends by visiting r

Postorder Traversal:

A H E M Y T J



Contents

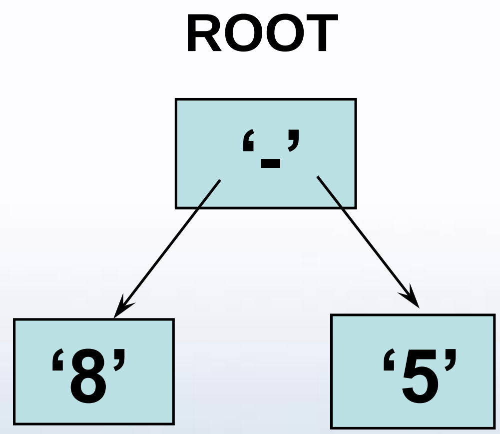
- Using tree structures
- Notion of tree structure
- Trees terminology
- N-ary and binary trees
- Computer representation of trees
- Binary search tree
- Decision tree
- Tree traversal
- Binary expression tree

Binary Expression Tree

Special kind of binary tree for arithmetic expressions:

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

Binary Expression Tree Traversal



INORDER TRAVERSAL : 8 – 5

PREORDER TRAVERSAL: – 8 5

POSTORDER TRAVERSAL: 8 5 –

Levels and 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.

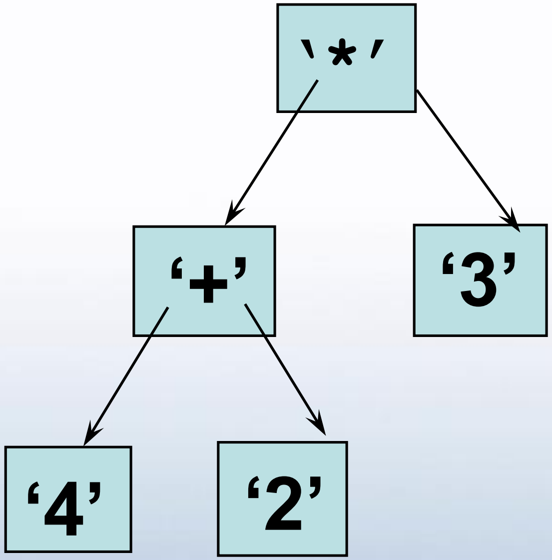
Operations at higher levels of the tree are evaluated later than those below them. The operation at the root is always the last operation performed.

Infix, Postfix and Prefix Expressions

Infix, Postfix and Prefix notations are three different but equivalent ways of writing expressions:

- Infix notation: X + Y
  - Operators are written in-between their operands.
  - Produced by inorder traversal
- Postfix notation (also known as "Reverse Polish notation"): X Y +
  - Operators are written after their operands.
  - Produced by postorder traversal
- Prefix notation (also known as "Polish notation"): + X Y
  - Operators are written before their operands.
  - Produced by preorder traversal

Infix, Postfix and Prefix

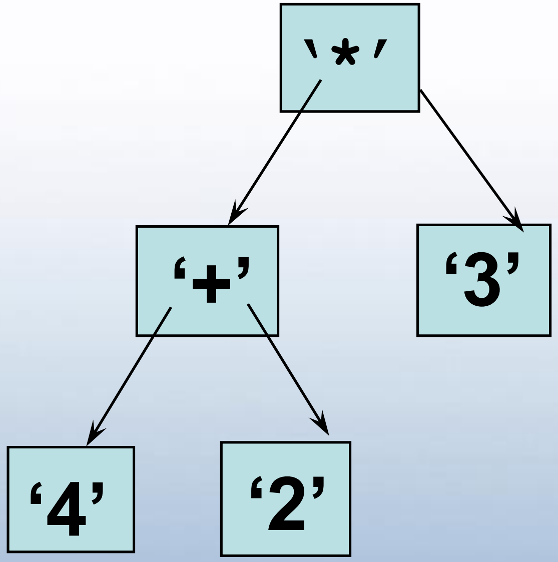


Infix: ((4 + 2) * 3) Needs extra information to define the order of evaluation

Prefix: * + 4 2 3 Operators act on the two nearest values on the right.

Postfix: 4 2 + 3 * Operators act on values immediately to the left

Binary Expression Tree Evaluation



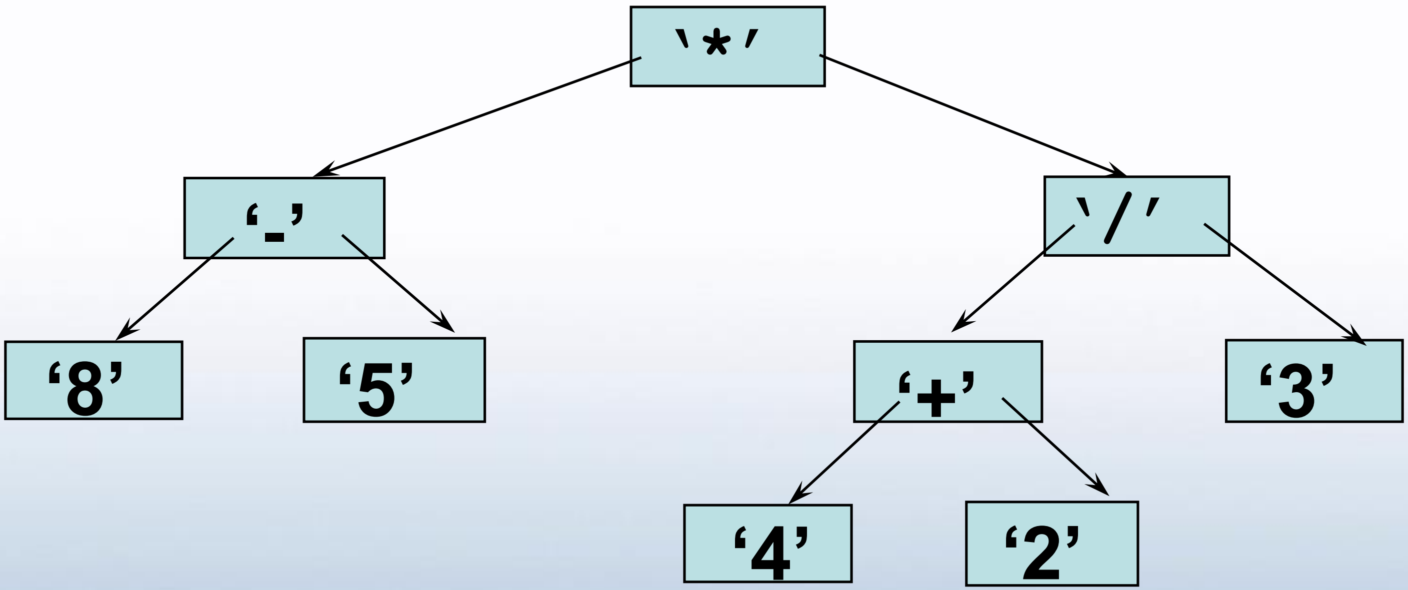
What value does it have?

(4 + 2) * 3 = 18

PREORDER TRAVERSAL: `* + 4 2 3`

Evaluation can be done in this order, but the operation is applied only when both operands are defined

Evaluation

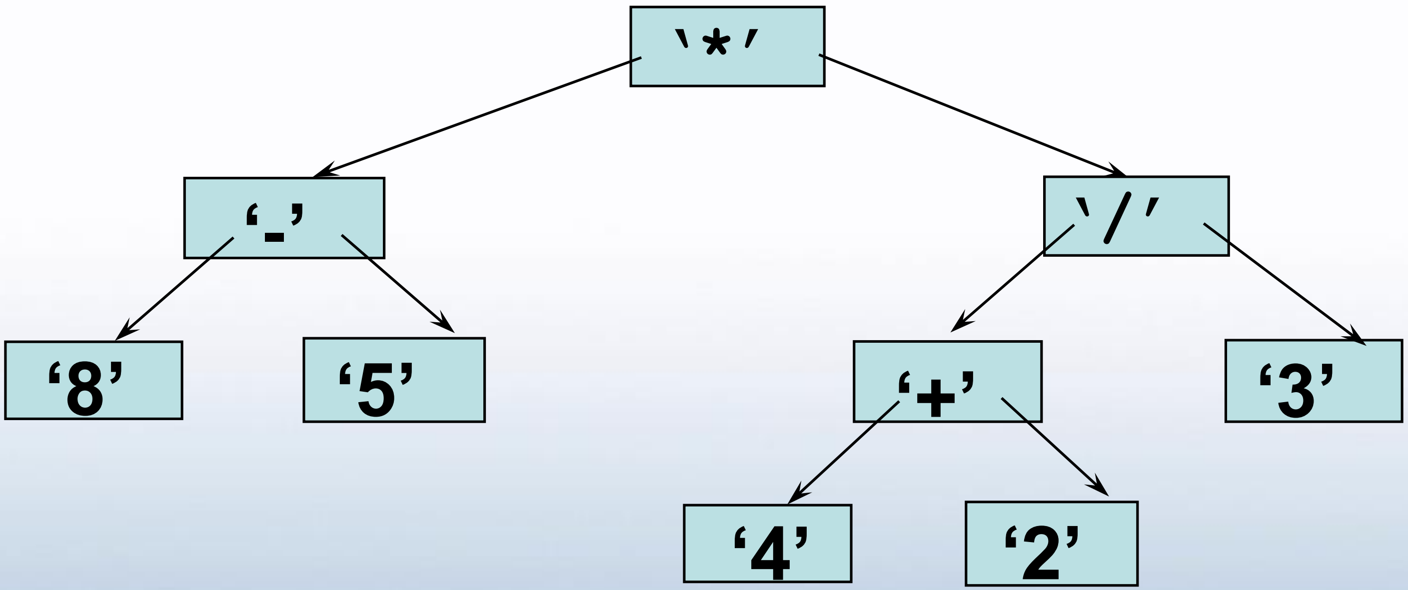


Infix: ((8 - 5) * ((4 + 2) / 3)) *easy to read, hard to evaluate*

Prefix: * - 8 5 / + 4 2 3 *complex to evaluate*

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

Evaluation



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

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

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

“Cat Tree”