CHAPTER

11 Trees

A connected graph that contains no simple circuits is called a tree. Trees were used as long ago as 1857, when the English mathematician Arthur Cayley used them to count certain types of chemical compounds. Since that time, trees have been employed to solve problems in a wide variety of disciplines, as the examples in this chapter will show.

Trees are particularly useful in computer science, where they are employed in a wide range of algorithms. For instance, trees are used to construct efficient algorithms for locating items in a list. They can be used in algorithms, such as Huffman coding, that construct efficient codes saving costs in data transmission and storage. Trees can be used to study games such as checkers and chess and can help determine winning strategies for playing these games. Trees can be used to model procedures carried out using a sequence of decisions. Constructing these models can help determine the computational complexity of algorithms based on a sequence of decisions, such as sorting algorithms.

Procedures for building trees containing every vertex of a graph, including depth-first search and breadth-first search, can be used to systematically explore the vertices of a graph. Exploring the vertices of a graph via depth-first search, also known as backtracking, allows for the systematic search for solutions to a wide variety of problems, such as determining how eight queens can be placed on a chessboard so that no queen can attack another.

We can assign weights to the edges of a tree to model many problems. For example, using weighted trees we can develop algorithms to construct networks containing the least expensive set of telephone lines linking different network nodes.

11.1 Introduction to Trees

In Chapter 10 we showed how graphs can be used to model and solve many problems. In this chapter we will focus on a particular type of graph called a **tree**, so named because such graphs resemble trees. For example, *family trees* are graphs that represent genealogical charts. Family trees use vertices to represent the members of a family and edges to represent parent–child relationships. The family tree of the male members of the Bernoulli family of Swiss mathematicians is shown in Figure 1. The undirected graph representing a family tree (restricted to people of just one gender and with no inbreeding) is an example of a tree.

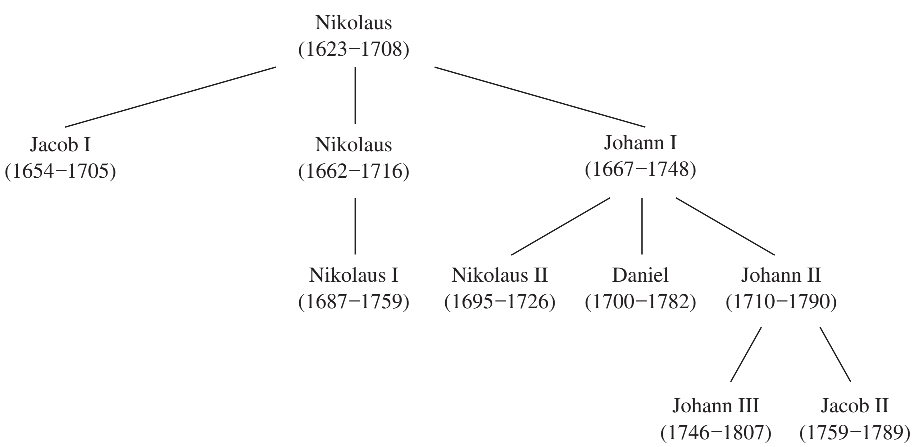


FIGURE 1 The Bernoulli Family of Mathematicians.

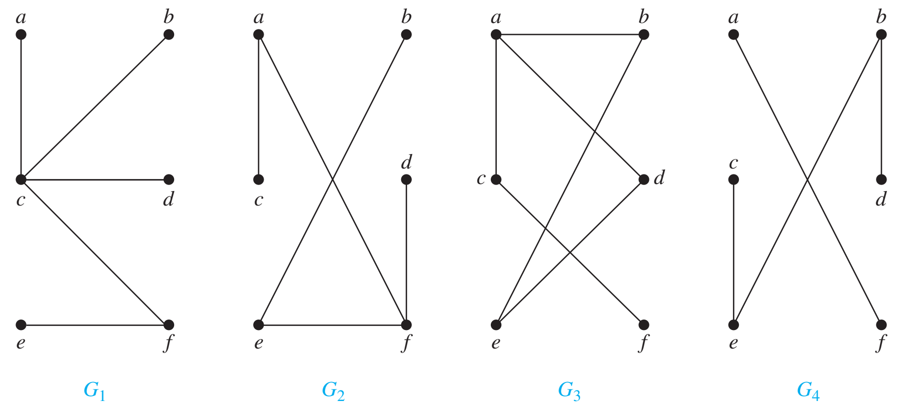


FIGURE 2 Examples of Trees and Graphs That Are Not Trees.

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

Because a tree cannot have a simple circuit, a tree cannot contain multiple edges or loops. Therefore any tree must be a simple graph.

EXAMPLE 1 Which of the graphs shown in Figure 2 are trees?

Solution: G_1 and G_2 are trees, because both are connected graphs with no simple circuits. G_3 is not a tree because e, b, a, d, e is a simple circuit in this graph. Finally, G_4 is not a tree because it is not connected. ◂

Any connected graph that contains no simple circuits is a tree. What about graphs containing no simple circuits that are not necessarily connected? These graphs are called **forests** and have the property that each of their connected components is a tree. Figure 3 displays a forest.

Trees are often defined as undirected graphs with the property that there is a unique simple path between every pair of vertices. Theorem 1 shows that this alternative definition is equivalent to our definition.

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

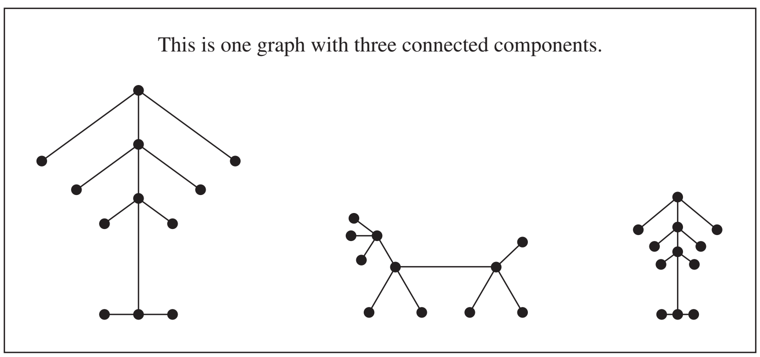


FIGURE 3 Example of a Forest.

Proof: First assume that T is a tree. Then T is a connected graph with no simple circuits. Let x and y be two vertices of T. Because T is connected, by Theorem 1 of Section 10.4 there is a simple path between x and y. Moreover, this path must be unique, for if there were a second such path, the path formed by combining the first path from x to y followed by the path from y to x obtained by reversing the order of the second path from x to y would form a circuit. This implies, using Exercise 59 of Section 10.4, that there is a simple circuit in T. Hence, there is a unique simple path between any two vertices of a tree.

Now assume that there is a unique simple path between any two vertices of a graph T. Then T is connected, because there is a path between any two of its vertices. Furthermore, T can have no simple circuits. To see that this is true, suppose T had a simple circuit that contained the vertices x and y. Then there would be two simple paths between x and y, because the simple circuit is made up of a simple path from x to y and a second simple path from y to x. Hence, a graph with a unique simple path between any two vertices is a tree. ◁

Rooted Trees

In many applications of trees, a particular vertex of a tree is designated as the **root**. Once we specify a root, we can assign a direction to each edge as follows. Because there is a unique path from the root to each vertex of the graph (by Theorem 1), we direct each edge away from the root. Thus, a tree together with its root produces a directed graph called a **rooted tree**.

DEFINITION 2 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.

Rooted trees can also be defined recursively. Refer to Section 5.3 to see how this can be done. We can change an unrooted tree into a rooted tree by choosing any vertex as the root. Note that different choices of the root produce different rooted trees. For instance, Figure 4 displays the rooted trees formed by designating a to be the root and c to be the root, respectively, in the tree T. We usually draw a rooted tree with its root at the top of the graph. The arrows indicating the directions of the edges in a rooted tree can be omitted, because the choice of root determines the directions of the edges.

The terminology for trees has botanical and genealogical origins. Suppose that T is a rooted tree. If v is a vertex in T other than the root, the **parent** of v is the unique vertex u such that there is a directed edge from u to v (the reader should show that such a vertex is unique). When u is the parent of v, v is called a **child** of u. Vertices with the same parent are called **siblings**. The **ancestors** of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root (that is, its parent, its parent's parent, and so on, until the root is reached). The **descendants** of a vertex v are those vertices that have v as

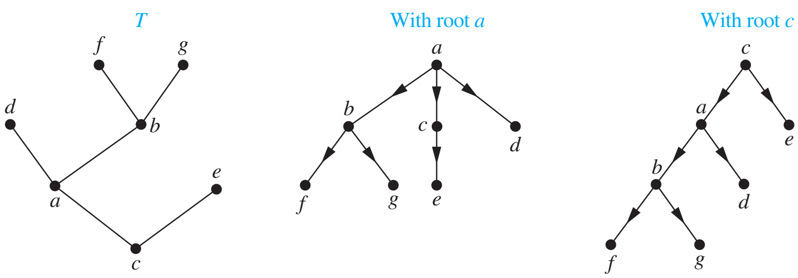


FIGURE 4 **A Tree and Rooted Trees Formed by Designating Two Different Roots.**

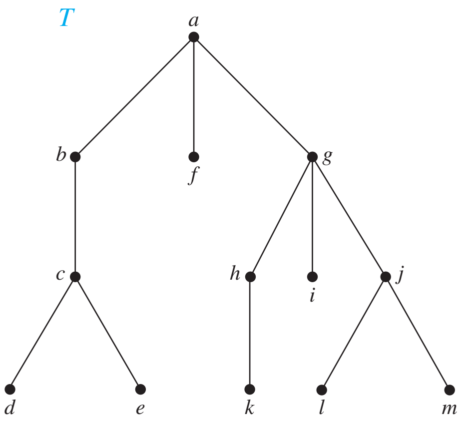


FIGURE 5 **A Rooted Tree T.**

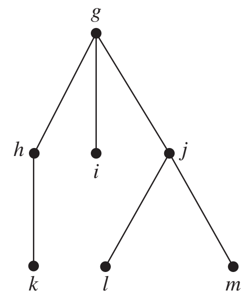


FIGURE 6 **The Subtree Rooted at g.**

an ancestor. A vertex of a rooted tree is called a **leaf** if it has no children. Vertices that have children are called **internal vertices**. The root is an internal vertex unless it is the only vertex in the graph, in which case it is a leaf.

If a is a vertex in a tree, the **subtree** with a as its root is the subgraph of the tree consisting of a and its descendants and all edges incident to these descendants.

EXAMPLE 2 In the rooted tree T (with root a) shown in Figure 5, find the parent of c, the children of g, the siblings of h, all ancestors of e, all descendants of b, all internal vertices, and all leaves. What is the subtree rooted at g?

Extra Examples

Solution: The parent of c is b. The children of g are h, i, and j. The siblings of h are i and j. The ancestors of e are c, b, and a. The descendants of b are c, d, and e. The internal vertices are a, b, c, g, h, and j. The leaves are d, e, f, i, k, l, and m. The subtree rooted at g is shown in Figure 6. ◀

Rooted trees with the property that all of their internal vertices have the same number of children are used in many different applications. Later in this chapter we will use such trees to study problems involving searching, sorting, and coding.

DEFINITION 3 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*.

Links

EXAMPLE 3 Are the rooted trees in Figure 7 full m-ary trees for some positive integer m?

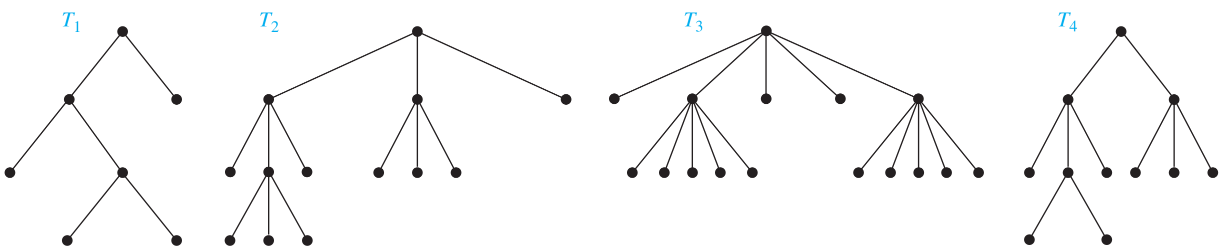


FIGURE 7 **Four Rooted Trees.**

Solution: T_1 is a full binary tree because each of its internal vertices has two children. T_2 is a full 3-ary tree because each of its internal vertices has three children. In T_3 each internal vertex has five children, so T_3 is a full 5-ary tree. T_4 is not a full m-ary tree for any m because some of its internal vertices have two children and others have three children. ◀

ORDERED ROOTED TREES An **ordered rooted tree** is a rooted tree where the children of each internal vertex are ordered. Ordered rooted trees are drawn so that the children of each internal vertex are shown in order from left to right. Note that a representation of a rooted tree in the conventional way determines an ordering for its edges. We will use such orderings of edges in drawings without explicitly mentioning that we are considering a rooted tree to be ordered.

In an ordered binary tree (usually called just a **binary tree**), if an internal vertex has two children, the first child is called the **left child** and the second child is called the **right child**. The tree rooted at the left child of a vertex is called the **left subtree** of this vertex, and the tree rooted at the right child of a vertex is called the **right subtree** of the vertex. The reader should note that for some applications every vertex of a binary tree, other than the root, is designated as a right or a left child of its parent. This is done even when some vertices have only one child. We will make such designations whenever it is necessary, but not otherwise.

Ordered rooted trees can be defined recursively. Binary trees, a type of ordered rooted trees, were defined this way in Section 5.3.

EXAMPLE 4 What are the left and right children of d in the binary tree T shown in Figure 8(a) (where the order is that implied by the drawing)? What are the left and right subtrees of c?

Solution: The left child of d is f and the right child is g. We show the left and right subtrees of c in Figures 8(b) and 8(c), respectively. ◀

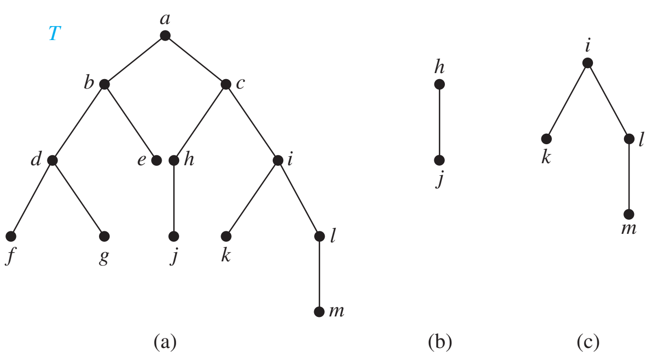


FIGURE 8 **A Binary Tree T and Left and Right Subtrees of the Vertex c.**

Just as in the case of graphs, there is no standard terminology used to describe trees, rooted trees, ordered rooted trees, and binary trees. This nonstandard terminology occurs because trees are used extensively throughout computer science, which is a relatively young field. The reader should carefully check meanings given to terms dealing with trees whenever they occur.

Trees as Models

Trees are used as models in such diverse areas as computer science, chemistry, geology, botany, and psychology. We will describe a variety of such models based on trees.

and x_7 and x_8 using P_7. In the second step, we add $x_1 + x_2$ and $x_3 + x_4$ using P_2 and $x_5 + x_6$ and $x_7 + x_8$ using P_3. Finally, in the third step, we add $x_1 + x_2 + x_3 + x_4$ and $x_5 + x_6 + x_7 + x_8$ using P_1. The three steps used to add eight numbers compares favorably to the seven steps required to add eight numbers serially, where the steps are the addition of one number to the sum of the previous numbers in the list. ◀

Properties of Trees

We will often need results relating the numbers of edges and vertices of various types in trees.

THEOREM 2 A tree with n vertices has $n - 1$ edges.

Proof: We will use mathematical induction to prove this theorem. Note that for all the trees here we can choose a root and consider the tree rooted.

BASIS STEP: When $n = 1$, a tree with $n = 1$ vertex has no edges. It follows that the theorem is true for $n = 1$.

INDUCTIVE STEP: The inductive hypothesis states that every tree with k vertices has $k - 1$ edges, where k is a positive integer. Suppose that a tree T has $k + 1$ vertices and that v is a leaf of T (which must exist because the tree is finite), and let w be the parent of v. Removing from T the vertex v and the edge connecting w to v produces a tree T' with k vertices, because the resulting graph is still connected and has no simple circuits. By the inductive hypothesis, T' has $k - 1$ edges. It follows that T has k edges because it has one more edge than T', the edge connecting v and w. This completes the inductive step. ◁

Recall that a tree is a connected undirected graph with no simple circuits. So, when G is an undirected graph with n vertices, Theorem 2 tells us that the two conditions *(i)* G is connected and *(ii)* G has no simple circuits, imply *(iii)* G has $n - 1$ edges. Also, when *(i)* and *(iii)* hold, then *(ii)* must also hold, and when *(ii)* and *(iii)* hold, *(i)* must also hold. That is, if G is connected and G has $n - 1$ edges, then G has no simple circuits, so that G is a tree (see Exercise 15(a)), and if G has no simple circuits and G has $n - 1$ edges, then G is connected, and so is a tree (see Exercise 15(b)). Consequently, when two of *(i)*, *(ii)*, and *(iii)* hold, the third condition must also hold, and G must be a tree.

COUNTING VERTICES IN FULL m-ARY TREES The number of vertices in a full m-ary tree with a specified number of internal vertices is determined, as Theorem 3 shows. As in Theorem 2, we will use n to denote the number of vertices in a tree.

THEOREM 3 A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.

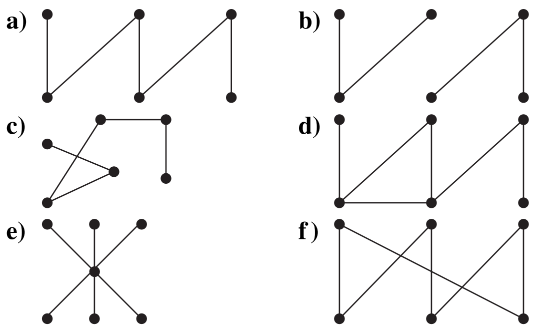
Proof: Every vertex, except the root, is the child of an internal vertex. Because each of the i internal vertices has m children, there are mi vertices in the tree other than the root. Therefore, the tree contains $n = mi + 1$ vertices. ◁

Suppose that T is a full m-ary tree. Let i be the number of internal vertices and l the number of leaves in this tree. Once one of n, i, and l is known, the other two quantities are determined. Theorem 4 explains how to find the other two quantities from the one that is known.

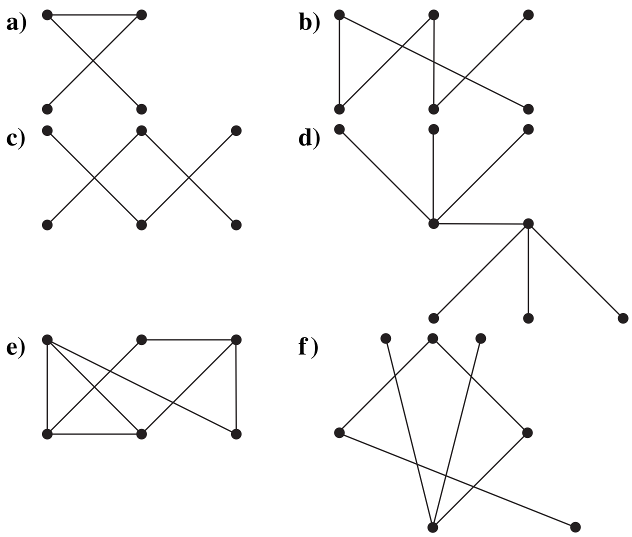
Then each leaf is at level h or $h-1$, and because the height is h, there is at least one leaf at level h. It follows that there must be more than m^{h-1} leaves (see Exercise 30). Because $l \le m^h$, we have $m^{h-1} < l \le m^h$. Taking logarithms to the base m in this inequality gives $h - 1 < \log_m l \le h$. Hence, $h = \lceil \log_m l \rceil$. ◁

Exercises

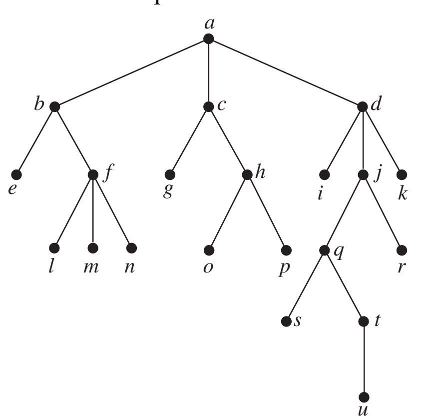
1. Which of these graphs are trees?



2. Which of these graphs are trees?

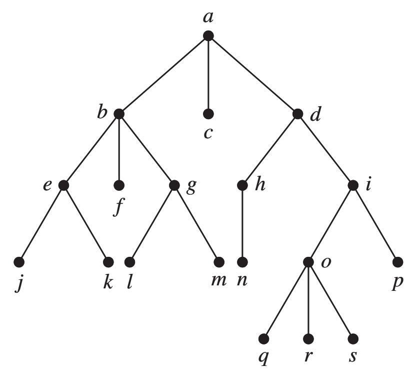


3. Answer these questions about the rooted tree illustrated.



a) Which vertex is the root?
b) Which vertices are internal?
c) Which vertices are leaves?
d) Which vertices are children of j?
e) Which vertex is the parent of h?
f) Which vertices are siblings of o?
g) Which vertices are ancestors of m?
h) Which vertices are descendants of b?

4. Answer the same questions as listed in Exercise 3 for the rooted tree illustrated.



5. Is the rooted tree in Exercise 3 a full m-ary tree for some positive integer m?

6. Is the rooted tree in Exercise 4 a full m-ary tree for some positive integer m?

7. What is the level of each vertex of the rooted tree in Exercise 3?

8. What is the level of each vertex of the rooted tree in Exercise 4?

9. Draw the subtree of the tree in Exercise 3 that is rooted at

a) a. **b)** c. **c)** e.

10. Draw the subtree of the tree in Exercise 4 that is rooted at

a) a. **b)** c. **c)** e.

11. a) How many nonisomorphic unrooted trees are there with three vertices?
b) How many nonisomorphic rooted trees are there with three vertices (using isomorphism for directed graphs)?

***12. a)** How many nonisomorphic unrooted trees are there with four vertices?
b) How many nonisomorphic rooted trees are there with four vertices (using isomorphism for directed graphs)?

***13. a)** How many nonisomorphic unrooted trees are there with five vertices?

b) How many nonisomorphic rooted trees are there with five vertices (using isomorphism for directed graphs)?

***14.** Show that a simple graph is a tree if and only if it is connected but the deletion of any of its edges produces a graph that is not connected.

☞*** 15.** Let G be a simple graph with n vertices. Show that

a) G is a tree if and only if it is connected and has $n - 1$ edges.

b) G is a tree if and only if G has no simple circuits and has $n - 1$ edges. [*Hint:* To show that G is connected if it has no simple circuits and $n - 1$ edges, show that G cannot have more than one connected component.]

16. Which complete bipartite graphs $K_{m,n}$, where m and n are positive integers, are trees?

17. How many edges does a tree with 10,000 vertices have?

18. How many vertices does a full 5-ary tree with 100 internal vertices have?

19. How many edges does a full binary tree with 1000 internal vertices have?

20. How many leaves does a full 3-ary tree with 100 vertices have?

21. Suppose 1000 people enter a chess tournament. Use a rooted tree model of the tournament to determine how many games must be played to determine a champion, if a player is eliminated after one loss and games are played until only one entrant has not lost. (Assume there are no ties.)

22. A chain letter starts when a person sends a letter to five others. Each person who receives the letter either sends it to five other people who have never received it or does not send it to anyone. Suppose that 10,000 people send out the letter before the chain ends and that no one receives more than one letter. How many people receive the letter, and how many do not send it out?

23. A chain letter starts with a person sending a letter out to 10 others. Each person is asked to send the letter out to 10 others, and each letter contains a list of the previous six people in the chain. Unless there are fewer than six names in the list, each person sends one dollar to the first person in this list, removes the name of this person from the list, moves up each of the other five names one position, and inserts his or her name at the end of this list. If no person breaks the chain and no one receives more than one letter, how much money will a person in the chain ultimately receive?

***24.** Either draw a full m-ary tree with 76 leaves and height 3, where m is a positive integer, or show that no such tree exists.

***25.** Either draw a full m-ary tree with 84 leaves and height 3, where m is a positive integer, or show that no such tree exists.

***26.** A full m-ary tree T has 81 leaves and height 4.

a) Give the upper and lower bounds for m.

b) What is m if T is also balanced?

A **complete m-ary tree** is a full m-ary tree in which every leaf is at the same level.

27. Construct a complete binary tree of height 4 and a complete 3-ary tree of height 3.

28. How many vertices and how many leaves does a complete m-ary tree of height h have?

29. Prove

a) part (*ii*) of Theorem 4.

b) part (*iii*) of Theorem 4.

☞**30.** Show that a full m-ary balanced tree of height h has more than m^{h-1} leaves.

31. How many edges are there in a forest of t trees containing a total of n vertices?

32. Explain how a tree can be used to represent the table of contents of a book organized into chapters, where each chapter is organized into sections, and each section is organized into subsections.

33. How many different isomers do these saturated hydrocarbons have?

a) C_3H_8 **b)** C_5H_{12} **c)** C_6H_{14}

34. What does each of these represent in an organizational tree?

a) the parent of a vertex

b) a child of a vertex

c) a sibling of a vertex

d) the ancestors of a vertex

e) the descendants of a vertex

f) the level of a vertex

g) the height of the tree

35. Answer the same questions as those given in Exercise 34 for a rooted tree representing a computer file system.

36. a) Draw the complete binary tree with 15 vertices that represents a tree-connected network of 15 processors.

b) Show how 16 numbers can be added using the 15 processors in part (a) using four steps.

37. Let n be a power of 2. Show that n numbers can be added in $\log n$ steps using a tree-connected network of $n - 1$ processors.

***38.** A **labeled tree** is a tree where each vertex is assigned a label. Two labeled trees are considered isomorphic when there is an isomorphism between them that preserves the labels of vertices. How many nonisomorphic trees are there with three vertices labeled with different integers from the set $\{1, 2, 3\}$? How many nonisomorphic trees are there with four vertices labeled with different integers from the set $\{1, 2, 3, 4\}$?