# Key Terms and Results

## TERMS

**tree:** a connected undirected graph with no simple circuits

**forest:** an undirected graph with no simple circuits

**rooted tree:** a directed graph with a specified vertex, called the root, such that there is a unique path to every other vertex from this root

**subtree:** a subgraph of a tree that is also a tree

**parent of $v$ in a rooted tree:** the vertex $u$ such that $(u, v)$ is an edge of the rooted tree

**child of a vertex $v$ in a rooted tree:** any vertex with $v$ as its parent

**sibling of a vertex $v$ in a rooted tree:** a vertex with the same parent as $v$

**ancestor of a vertex $v$ in a rooted tree:** any vertex on the path from the root to $v$

**descendant of a vertex $v$ in a rooted tree:** any vertex that has $v$ as an ancestor

**internal vertex:** a vertex that has children

**leaf:** a vertex with no children

**level of a vertex:** the length of the path from the root to this vertex

**height of a tree:** the largest level of the vertices of a tree

**$m$-ary tree:** a tree with the property that every internal vertex has no more than $m$ children

**full $m$-ary tree:** a tree with the property that every internal vertex has exactly $m$ children

**binary tree:** an $m$-ary tree with $m = 2$ (each child may be designated as a left or a right child of its parent)

**ordered tree:** a tree in which the children of each internal vertex are linearly ordered

**balanced tree:** a tree in which every leaf is at level $h$ or $h - 1$, where $h$ is the height of the tree

**binary search tree:** a binary tree in which the vertices are labeled with items so that a label of a vertex is greater than the labels of all vertices in the left subtree of this vertex and is less than the labels of all vertices in the right subtree of this vertex

**decision tree:** a rooted tree where each vertex represents a possible outcome of a decision and the leaves represent the possible solutions of a problem

**game tree:** a rooted tree where vertices represent the possible positions of a game as it progresses and edges represent legal moves between these positions

**prefix code:** a code that has the property that the code of a character is never a prefix of the code of another character

**minmax strategy:** the strategy where the first player and second player move to positions represented by a child with maximum and minimum value, respectively

**value of a vertex in a game tree:** for a leaf, the payoff to the first player when the game terminates in the position represented by this leaf; for an internal vertex, the maximum or minimum of the values of its children, for an internal vertex at an even or odd level, respectively

**tree traversal:** a listing of the vertices of a tree

**preorder traversal:** a listing of the vertices of an ordered rooted tree defined recursively—the root is listed, followed by the first subtree, followed by the other in the order they occur from left to right

**inorder traversal:** a listing of the vertices of an ordered rooted tree defined recursively—the first subtree is listed, followed by the root, followed by the other subtrees in the order they occur from left to right

**postorder traversal:** a listing of the vertices of an ordered rooted tree defined recursively—the subtrees are listed in the order they occur from left to right, followed by the root

**infix notation:** the form of an expression (including a full set of parentheses) obtained from an inorder traversal of the binary tree representing this expression

**prefix (or Polish) notation:** the form of an expression obtained from a preorder traversal of the tree representing this expression

**postfix (or reverse Polish) notation:** the form of an expression obtained from a postorder traversal of the tree representing this expression

**spanning tree:** a tree containing all vertices of a graph

**minimum spanning tree:** a spanning tree with smallest possible sum of weights of its edges

## RESULTS

A graph is a tree if and only if there is a unique simple path between every pair of its vertices.

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

A full $m$-ary tree with $i$ internal vertices has $mi + 1$ vertices.

The relationships among the numbers of vertices, leaves, and internal vertices in a full $m$-ary tree (see Theorem 4 in Section 11.1)

There are at most $m^h$ leaves in an $m$-ary tree of height $h$.

If an $m$-ary tree has $l$ leaves, its height $h$ is at least $\lceil \log_m l \rceil$. If the tree is also full and balanced, then its height is $\lceil \log_m l \rceil$.

**Huffman coding:** a procedure for constructing an optimal binary code for a set of symbols, given the frequencies of these symbols

**depth-first search, or backtracking:** a procedure for constructing a spanning tree by adding edges that form a path until this is not possible, and then moving back up the path until a vertex is found where a new path can be formed

**breadth-first search:** a procedure for constructing a spanning tree that successively adds all edges incident to the last set of edges added, unless a simple circuit is formed

**Prim's algorithm:** a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges with minimal weight among all edges incident to a vertex already in the tree so that no edge produces a simple circuit when it is added

**Kruskal's algorithm:** a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges of least weight that are not already in the tree such that no edge produces a simple circuit when it is added

## Review Questions

**1. a)** Define a tree. **b)** Define a forest.

**2.** Can there be two different simple paths between the vertices of a tree?

**3.** Give at least three examples of how trees are used in modeling.

**4. a)** Define a rooted tree and the root of such a tree.
**b)** Define the parent of a vertex and a child of a vertex in a rooted tree.
**c)** What are an internal vertex, a leaf, and a subtree in a rooted tree?
**d)** Draw a rooted tree with at least 10 vertices, where the degree of each vertex does not exceed 3. Identify the root, the parent of each vertex, the children of each vertex, the internal vertices, and the leaves.

**5. a)** How many edges does a tree with $n$ vertices have?
**b)** What do you need to know to determine the number of edges in a forest with $n$ vertices?

**6. a)** Define a full $m$-ary tree.
**b)** How many vertices does a full $m$-ary tree have if it has $i$ internal vertices? How many leaves does the tree have?

**7. a)** What is the height of a rooted tree?
**b)** What is a balanced tree?
**c)** How many leaves can an $m$-ary tree of height $h$ have?

**8. a)** What is a binary search tree?
**b)** Describe an algorithm for constructing a binary search tree.
**c)** Form a binary search tree for the words *vireo, warbler, egret, grosbeak, nuthatch*, and *kingfisher*.

**9. a)** What is a prefix code?
**b)** How can a prefix code be represented by a binary tree?

**10. a)** Define preorder, inorder, and postorder tree traversal.
**b)** Give an example of preorder, postorder, and inorder traversal of a binary tree of your choice with at least 12 vertices.

**11. a)** Explain how to use preorder, inorder, and postorder traversals to find the prefix, infix, and postfix forms of an arithmetic expression.
**b)** Draw the ordered rooted tree that represents $((x - 3) + ((x/4) + (x - y) \uparrow 3))$.
**c)** Find the prefix and postfix forms of the expression in part (b).

**12.** Show that the number of comparisons used by a sorting algorithm to sort a list of $n$ elements is at least $\lceil \log n! \rceil$.

**13. a)** Describe the Huffman coding algorithm for constructing an optimal code for a set of symbols, given the frequency of these symbols.
**b)** Use Huffman coding to find an optimal code for these symbols and frequencies: A: 0.2, B: 0.1, C: 0.3, D: 0.4.

**14.** Draw the game tree for nim if the starting position consists of two piles with one and four stones, respectively. Who wins the game if both players follow an optimal strategy?

**15. a)** What is a spanning tree of a simple graph?
**b)** Which simple graphs have spanning trees?
**c)** Describe at least two different applications that require that a spanning tree of a simple graph be found.

**16. a)** Describe two different algorithms for finding a spanning tree in a simple graph.
**b)** Illustrate how the two algorithms you described in part (a) can be used to find the spanning tree of a simple graph, using a graph of your choice with at least eight vertices and 15 edges.

**17. a)** Explain how backtracking can be used to determine whether a simple graph can be colored using $n$ colors.
**b)** Show, with an example, how backtracking can be used to show that a graph with a chromatic number equal to 4 cannot be colored with three colors, but can be colored with four colors.

**18. a)** What is a minimum spanning tree of a connected weighted graph?
**b)** Describe at least two different applications that require that a minimum spanning tree of a connected weighted graph be found.

**19. a)** Describe Kruskal's algorithm and Prim's algorithm for finding minimum spanning trees.
**b)** Illustrate how Kruskal's algorithm and Prim's algorithm are used to find a minimum spanning tree, using a weighted graph with at least eight vertices and 15 edges.

## Supplementary Exercises

*1. Show that a simple graph is a tree if and only if it contains no simple circuits and the addition of an edge connecting two nonadjacent vertices produces a new graph that has exactly one simple circuit (where circuits that contain the same edges are not considered different).

*2. How many nonisomorphic rooted trees are there with six vertices?

3. Show that every tree with at least one edge must have at least two pendant vertices.

4. Show that a tree with $n$ vertices that has $n - 1$ pendant vertices must be isomorphic to $K_{1,n-1}$.

5. What is the sum of the degrees of the vertices of a tree with $n$ vertices?

*6. Suppose that $d_1, d_2, \ldots, d_n$ are $n$ positive integers with sum $2n - 2$. Show that there is a tree that has $n$ vertices such that the degrees of these vertices are $d_1, d_2, \ldots, d_n$.

7. Show that every tree is a planar graph.

8. Show that every tree is bipartite.

9. Show that every forest can be colored using two colors.

A **B-tree of degree *k*** is a rooted tree such that all its leaves are at the same level, its root has at least two and at most $k$ children unless it is a leaf, and every internal vertex other than the root has at least $\lceil k/2 \rceil$, but no more than $k$, children. Computer files can be accessed efficiently when B-trees are used to represent them.

10. Draw three different B-trees of degree 3 with height 4.

*11. Give an upper bound and a lower bound for the number of leaves in a B-tree of degree $k$ with height $h$.

*12. Give an upper bound and a lower bound for the height of a B-tree of degree $k$ with $n$ leaves.

The **binomial trees** $B_i$, $i = 0, 1, 2, \ldots$, are ordered rooted trees defined recursively:

*Basis step:* The binomial tree $B_0$ is the tree with a single vertex.

*Recursive step:* Let $k$ be a nonnegative integer. To construct the binomial tree $B_{k+1}$, add a copy of $B_k$ to a second copy of $B_k$ by adding an edge that makes the root of the first copy of $B_k$ the leftmost child of the root of the second copy of $B_k$.

13. Draw $B_k$ for $k = 0, 1, 2, 3, 4$.

14. How many vertices does $B_k$ have? Prove that your answer is correct.

15. Find the height of $B_k$. Prove that your answer is correct.

16. How many vertices are there in $B_k$ at depth $j$, where $0 \le j \le k$? Justify your answer.

17. What is the degree of the root of $B_k$? Prove that your answer is correct.

18. Show that the vertex of largest degree in $B_k$ is the root.

A rooted tree $T$ is called an ***$S_k$*-tree** if it satisfies this recursive definition. It is an $S_0$-tree if it has one vertex. For $k > 0$, $T$ is an $S_k$-tree if it can be built from two $S_{k-1}$-trees by making the root of one the root of the $S_k$-tree and making the root of the other the child of the root of the first $S_{k-1}$-tree.

19. Draw an $S_k$-tree for $k = 0, 1, 2, 3, 4$.

20. Show that an $S_k$-tree has $2^k$ vertices and a unique vertex at level $k$. This vertex at level $k$ is called the **handle**.

*21. Suppose that $T$ is an $S_k$-tree with handle $v$. Show that $T$ can be obtained from disjoint trees $T_0, T_1, \ldots, T_{k-1}$, with roots $r_0, r_1, \ldots, r_{k-1}$, respectively, where $v$ is not in any of these trees, where $T_i$ is an $S_i$-tree for $i = 0, 1, \ldots, k - 1$, by connecting $v$ to $r_0$ and $r_i$ to $r_{i+1}$ for $i = 0, 1, \ldots, k - 2$.

The listing of the vertices of an ordered rooted tree in **level order** begins with the root, followed by the vertices at level 1 from left to right, followed by the vertices at level 2 from left to right, and so on.

22. List the vertices of the ordered rooted trees in Figures 3 and 9 of Section 11.3 in level order.

23. Devise an algorithm for listing the vertices of an ordered rooted tree in level order.

*24. Devise an algorithm for determining if a set of universal addresses can be the addresses of the leaves of a rooted tree.

25. Devise an algorithm for constructing a rooted tree from the universal addresses of its leaves.

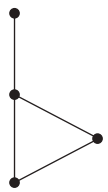
A **cut set** of a graph is a set of edges such that the removal of these edges produces a subgraph with more connected components than in the original graph, but no proper subset of this set of edges has this property.

26. Show that a cut set of a graph must have at least one edge in common with any spanning tree of this graph.

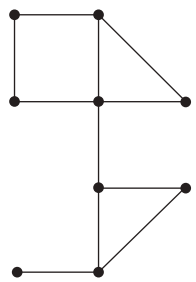
A **cactus** is a connected graph in which no edge is in more than one simple circuit not passing through any vertex other than its initial vertex more than once or its initial vertex other than at its terminal vertex (where two circuits that contain the same edges are not considered different).

**27.** Which of these graphs are cacti?

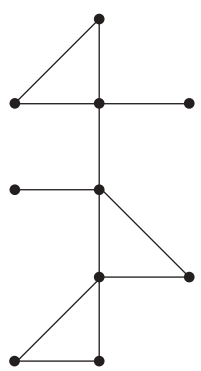
**a)**



**b)**



**c)**



**28.** Is a tree necessarily a cactus?

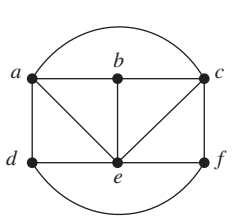
**29.** Show that a cactus is formed if we add a circuit containing new edges beginning and ending at a vertex of a tree.

***30.** Show that if every circuit not passing through any vertex other than its initial vertex more than once in a connected graph contains an odd number of edges, then this graph must be a cactus.

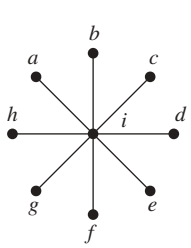
A **degree-constrained spanning tree** of a simple graph $G$ is a spanning tree with the property that the degree of a vertex in this tree cannot exceed some specified bound. Degree-constrained spanning trees are useful in models of transportation systems where the number of roads at an intersection is limited, models of communications networks where the number of links entering a node is limited, and so on.

In Exercises 31–33 find a degree-constrained spanning tree of the given graph where each vertex has degree less than or equal to 3, or show that such a spanning tree does not exist.

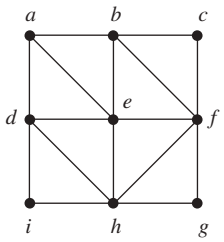
**31.**




**32.**




**33.**



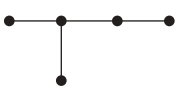

**34.** Show that a degree-constrained spanning tree of a simple graph in which each vertex has degree not exceeding 2 consists of a single Hamilton path in the graph.

**35.** A tree with $n$ vertices is called **graceful** if its vertices can be labeled with the integers $1, 2, \ldots, n$ such that the absolute values of the difference of the labels of adjacent vertices are all different. Show that these trees are graceful.

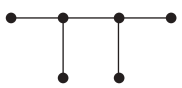
**a)**



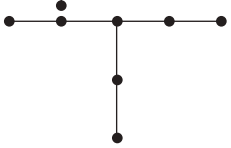
**b)**



**c)**



**d)**



A **caterpillar** is a tree that contains a simple path such that every vertex not contained in this path is adjacent to a vertex in the path.

**36.** Which of the graphs in Exercise 35 are caterpillars?

**37.** How many nonisomorphic caterpillars are there with six vertices?

****38.** **a)** Prove or disprove that all trees whose edges form a single path are graceful.

**b)** Prove or disprove that all caterpillars are graceful.

**39.** Suppose that in a long bit string the frequency of occurrence of a 0 bit is 0.9 and the frequency of a 1 bit is 0.1 and bits occur independently.

**a)** Construct a Huffman code for the four blocks of two bits, 00, 01, 10, and 11. What is the average number of bits required to encode a bit string using this code?

**b)** Construct a Huffman code for the eight blocks of three bits. What is the average number of bits required to encode a bit string using this code?

**40.** Suppose that $G$ is a directed graph with no circuits. Describe how depth-first search can be used to carry out a topological sort of the vertices of $G$.

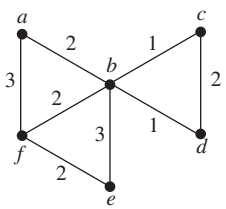
***41.** Suppose that $e$ is an edge in a weighted graph that is incident to a vertex $v$ such that the weight of $e$ does not exceed the weight of any other edge incident to $v$. Show that there exists a minimum spanning tree containing this edge.

**42.** Three couples arrive at the bank of a river. Each of the wives is jealous and does not trust her husband when he is with one of the other wives (and perhaps with other people), but not with her. How can six people cross to the other side of the river using a boat that can hold no more than two people so that no husband is alone with a woman other than his wife? Use a graph theory model.

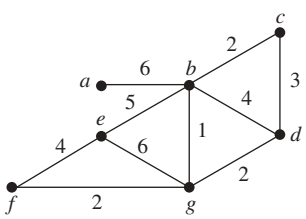
*43. Show that if no two edges in a weighted graph have the same weight, then the edge with least weight incident to a vertex $v$ is included in every minimum spanning tree.

44. Find a minimum spanning tree of each of these graphs where the degree of each vertex in the spanning tree does not exceed 2.

**a)**




**b)**




Let $G = (V, E)$ be a directed graph and let $r$ be a vertex in $G$. An **arborescence** of $G$ rooted at $r$ is a subgraph $T = (V, F)$ of $G$ such that the underlying undirected graph of $T$ is a spanning tree of the underlying undirected graph of $G$ and for every vertex $v \in V$ there is a path from $r$ to $v$ in $T$ (with directions taken into account).

45. Show that a subgraph $T = (V, F)$ of the graph $G = (V, E)$ is an arborescence of $G$ rooted at $r$ if and only if $T$ contains $r$, $T$ has no simple circuits, and for every vertex $v \in V$ other than $r$, $\deg^-(v) = 1$ in $T$.

46. Show that a directed graph $G = (V, E)$ has an arborescence rooted at the vertex $r$ if and only if for every vertex $v \in V$, there is a directed path from $r$ to $v$.

47. In this exercise we will develop an algorithm to find the strong components of a directed graph $G = (V, E)$. Recall that a vertex $w \in V$ is **reachable** from a vertex $v \in V$ if there is a directed path from $v$ to $w$.

**a)** Explain how to use breadth-first search in the directed graph $G$ to find all the vertices reachable from a vertex $v \in G$.

**b)** Explain how to use breadth-first search in $G^{conv}$ to find all the vertices from which a vertex $v \in G$ is reachable. (Recall that $G^{conv}$ is the directed graph obtained from $G$ by reversing the direction of all its edges.)

**c)** Explain how to use parts (a) and (b) to construct an algorithm that finds the strong components of a directed graph $G$, and explain why your algorithm is correct.

## Computer Projects

### Write programs with these input and output.

1. Given the adjacency matrix of an undirected simple graph, determine whether the graph is a tree.
2. Given the adjacency matrix of a rooted tree and a vertex in the tree, find the parent, children, ancestors, descendants, and level of this vertex.
3. Given the list of edges of a rooted tree and a vertex in the tree, find the parent, children, ancestors, descendants, and level of this vertex.
4. Given a list of items, construct a binary search tree containing these items.
5. Given a binary search tree and an item, locate or add this item to the binary search tree.
6. Given the ordered list of edges of an ordered rooted tree, find the universal addresses of its vertices.
7. Given the ordered list of edges of an ordered rooted tree, list its vertices in preorder, inorder, and postorder.
8. Given an arithmetic expression in prefix form, find its value.
9. Given an arithmetic expression in postfix form, find its value.
10. Given the frequency of symbols, use Huffman coding to find an optimal code for these symbols.
11. Given an initial position in the game of nim, determine an optimal strategy for the first player.
12. Given the adjacency matrix of a connected undirected simple graph, find a spanning tree for this graph using depth-first search.
13. Given the adjacency matrix of a connected undirected simple graph, find a spanning tree for this graph using breadth-first search.
14. Given a set of positive integers and a positive integer $N$, use backtracking to find a subset of these integers that have $N$ as their sum.
*15. Given the adjacency matrix of an undirected simple graph, use backtracking to color the graph with three colors, if this is possible.
*16. Given a positive integer $n$, solve the $n$-queens problem using backtracking.
17. Given the list of edges and their weights of a weighted undirected connected graph, use Prim's algorithm to find a minimum spanning tree of this graph.
18. Given the list of edges and their weights of a weighted undirected connected graph, use Kruskal's algorithm to find a minimum spanning tree of this graph.