CHAPTER 10

GRAPHS AND TREES

Graphs and trees have appeared previously in this book as convenient visualizations. For instance, a possibility tree shows all possible outcomes of a multistep operation with a finite number of outcomes for each step, the directed graph of a relation on a set shows which elements of the set are related to which a Hasse diagram illustrates the relations among elements in a partially ordered set, and a PERT diagram shows which tasks must precede which in executing a project.

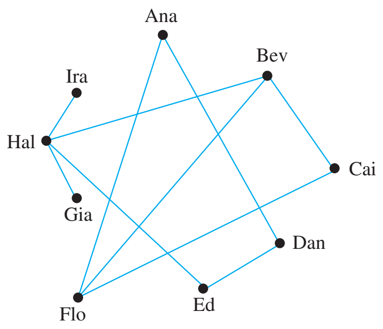
In this chapter we present some of the mathematics of graphs and trees, discussing concepts such as the degree of a vertex, connectedness, Euler and Hamiltonian circuits, representation of graphs by matrices, isomorphisms of graphs, the relation between the number of vertices and the number of edges of a tree, properties of rooted trees spanning trees, and shortest paths in graphs. Applications include uses of graphs and trees in the study of artificial intelligence, chemistry, scheduling problems, and transportation systems.

10.1 Graphs: Definitions and Basic Properties

The whole of mathematics consists in the organization of a series of aids to the imagination in the process of reasoning. — Alfred North Whitehead, 1861–1947

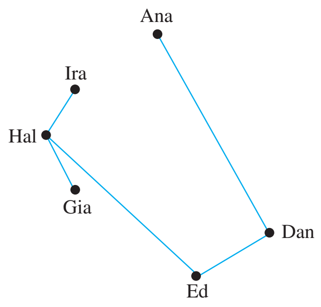
Imagine an organization that wants to set up teams of three to work on some projects. In order to maximize the number of people on each team who had previous experience working together successfully, the director asked the members to provide names of their past partners. This information is displayed below both in a table and in a diagram.

Name	Past Partners
Ana	Dan, Flo
Bev	Cai, Flo, Hal
Cai	Bev, Flo
Dan	Ana, Ed
Ed	Dan, Hal
Flo	Cai, Bev, Ana
Gia	Hal
Hal	Gia, Ed, Bev, Ira
Ira	Hal



From the diagram, it is easy to see that Bev, Cai, and Flo are a group of three past partners, and so they should form one of these teams. The figure on the next page shows the result when these three names are removed from the diagram.

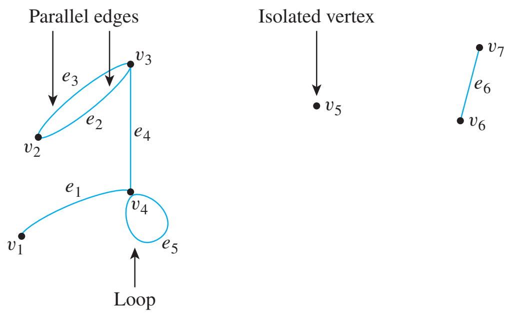
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



This drawing shows that placing Hal on the same team as Ed would leave Gia and Ira on a team containing no past partners. However, if Hal is placed on a team with Gia and Ira, then the remaining team would consist of Ana, Dan, and Ed, and both teams would contain at least one pair of past partners.

Drawings such as those shown previously are illustrations of a structure known as a *graph*. The dots are called *vertices* (plural of *vertex*) and the line segments joining vertices are called *edges*. As you can see from the drawing, it is possible for two edges to cross at a point that is not a vertex. Note also that the type of graph described here is quite different from the "graph of an equation" or the "graph of a function."

In general, a graph consists of a set of vertices and a set of edges connecting various pairs of vertices. The edges may be straight or curved and should either connect one vertex to another or a vertex to itself, as shown below.



In this drawing, the vertices have been labeled with v's and the edges with e's. When an edge connects a vertex to itself (as e_5 does), it is called a *loop*. When two edges connect the same pair of vertices (as e_2 and e_3 do), they are said to be *parallel*. It is quite possible for a vertex to be unconnected by an edge to any other vertex in the graph (as v_5 is), and in that case the vertex is said to be *isolated*. The formal definition of a graph follows.

• Definition

A **graph** G consists of two finite sets: a nonempty set $V(G)$ of **vertices** and a set $E(G)$ of **edges,** where each edge is associated with a set consisting of either one or two vertices called its **endpoints.** The correspondence from edges to endpoints is called the **edge-endpoint function.**

An edge with just one endpoint is called a **loop,** and two or more distinct edges with the same set of endpoints are said to be **parallel.** An edge is said to **connect** its endpoints; two vertices that are connected by an edge are called **adjacent;** and a vertex that is an endpoint of a loop is said to be **adjacent to itself.**

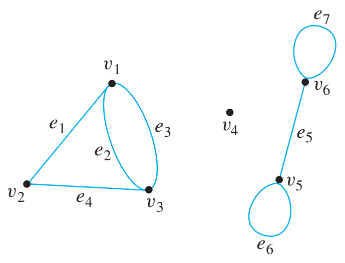
An edge is said to be **incident on** each of its endpoints, and two edges incident on the same endpoint are called **adjacent.** A vertex on which no edges are incident is called **isolated.**

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Graphs have pictorial representations in which the vertices are represented by dots and the edges by line segments. A given pictorial representation uniquely determines a graph.

Example 10.1.1 Terminology

Consider the following graph:



a. Write the vertex set and the edge set, and give a table showing the edge-endpoint function.

b. Find all edges that are incident on v_1, all vertices that are adjacent to v_1, all edges that are adjacent to e_1, all loops, all parallel edges, all vertices that are adjacent to themselves, and all isolated vertices.

Solution

a. vertex set $= \{v_1, v_2, v_3, v_4, v_5, v_6\}$
edge set $= \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$
edge-endpoint function:

Edge	**Endpoints**
e_1	$\{v_1, v_2\}$
e_2	$\{v_1, v_3\}$
e_3	$\{v_1, v_3\}$
e_4	$\{v_2, v_3\}$
e_5	$\{v_5, v_6\}$
e_6	$\{v_5\}$
e_7	$\{v_6\}$

Note that the isolated vertex v_4 does not appear in this table. Although each edge must have either one or two endpoints, a vertex need not be an endpoint of an edge.

b. e_1, e_2, and e_3 are incident on v_1.
v_2 and v_3 are adjacent to v_1.
e_2, e_3, and e_4 are adjacent to e_1.
e_6 and e_7 are loops.
e_2 and e_3 are parallel.
v_5 and v_6 are adjacent to themselves.
v_4 is an isolated vertex. ■

As noted earlier, a given pictorial representation uniquely determines a graph. However, a given graph may have more than one pictorial representation. Such things as the lengths or curvatures of the edges and the relative position of the vertices on the page may vary from one pictorial representation to another.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

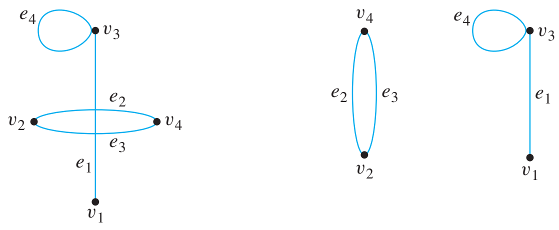
Example 10.1.2 Drawing More Than One Picture for a Graph

Consider the graph specified as follows:

$$\begin{aligned}\text{vertex set } &= \{v_1, v_2, v_3, v_4\}\\ \text{edge set } &= \{e_1, e_2, e_3, e_4\}\\ &\text{edge-endpoint function:}\end{aligned}$$

Edge	Endpoints
e_1	$\{v_1, v_3\}$
e_2	$\{v_2, v_4\}$
e_3	$\{v_2, v_4\}$
e_4	$\{v_3\}$

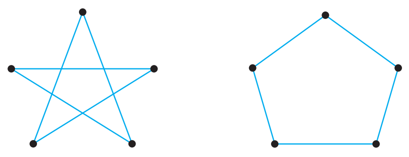
Both drawings (a) and (b) shown below are pictorial representations of this graph.



(a) (b) ■

Example 10.1.3 Labeling Drawings to Show They Represent the Same Graph

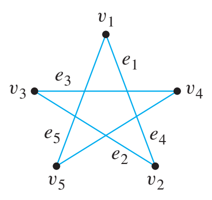
Consider the two drawings shown in Figure 10.1.1. Label vertices and edges in such a way that both drawings represent the same graph.



(a) (b)

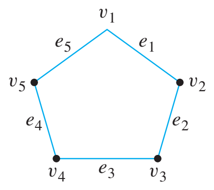
Figure 10.1.1

Solution Imagine putting one end of a piece of string at the top vertex of Figure 10.1.1(a) (call this vertex v_1), then laying the string to the next adjacent vertex on the lower right (call this vertex v_2), then laying it to the next adjacent vertex on the upper left (v_3), and so forth, returning finally to the top vertex v_1. Call the first edge e_1, the second e_2, and so forth, as shown below.



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Now imagine picking up the piece of string, together with its labels, and repositioning it as follows:



This is the same as Figure 10.1.1(b), so both drawings are representations of the graph with vertex set $\{v_1, v_2, v_3, v_4, v_5\}$, edge set $\{e_1, e_2, e_3, e_4, e_5\}$, and edge-endpoint function as follows:

Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_2, v_3\}$
e_3	$\{v_3, v_4\}$
e_4	$\{v_4, v_5\}$
e_5	$\{v_5, v_1\}$

■

In Chapter 8 we discussed the directed graph of a binary relation on a set. The general definition of directed graph is similar to the definition of graph, except that one associates an *ordered pair* of vertices with each edge instead of a *set* of vertices. Thus each edge of a directed graph can be drawn as an arrow going from the first vertex to the second vertex of the ordered pair.

• Definition

A **directed graph,** or **digraph,** consists of two finite sets: a nonempty set $V(G)$ of vertices and a set $D(G)$ of directed edges, where each is associated with an ordered pair of vertices called its **endpoints.** If edge e is associated with the pair (v, w) of vertices, then e is said to be the (**directed**) **edge** from v to w.

Note that each directed graph has an associated ordinary (undirected) graph, which is obtained by ignoring the directions of the edges.

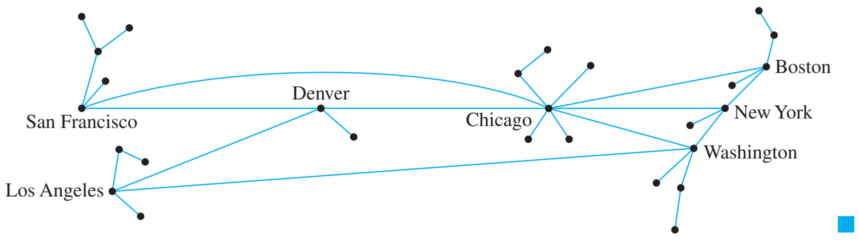
Examples of Graphs

Graphs are a powerful problem-solving tool because they enable us to represent a complex situation with a single image that can be analyzed both visually and with the aid of a computer. A few examples follow, and others are included in the exercises.

Example 10.1.4 Using a Graph to Represent a Network

Telephone, electric power, gas pipeline, and air transport systems can all be represented by graphs, as can computer networks—from small local area networks to the global Internet system that connects millions of computers worldwide. Questions that arise in the design of such systems involve choosing connecting edges to minimize cost, optimize a certain type of service, and so forth. A typical network, called a hub and spoke model, is shown on the next page.

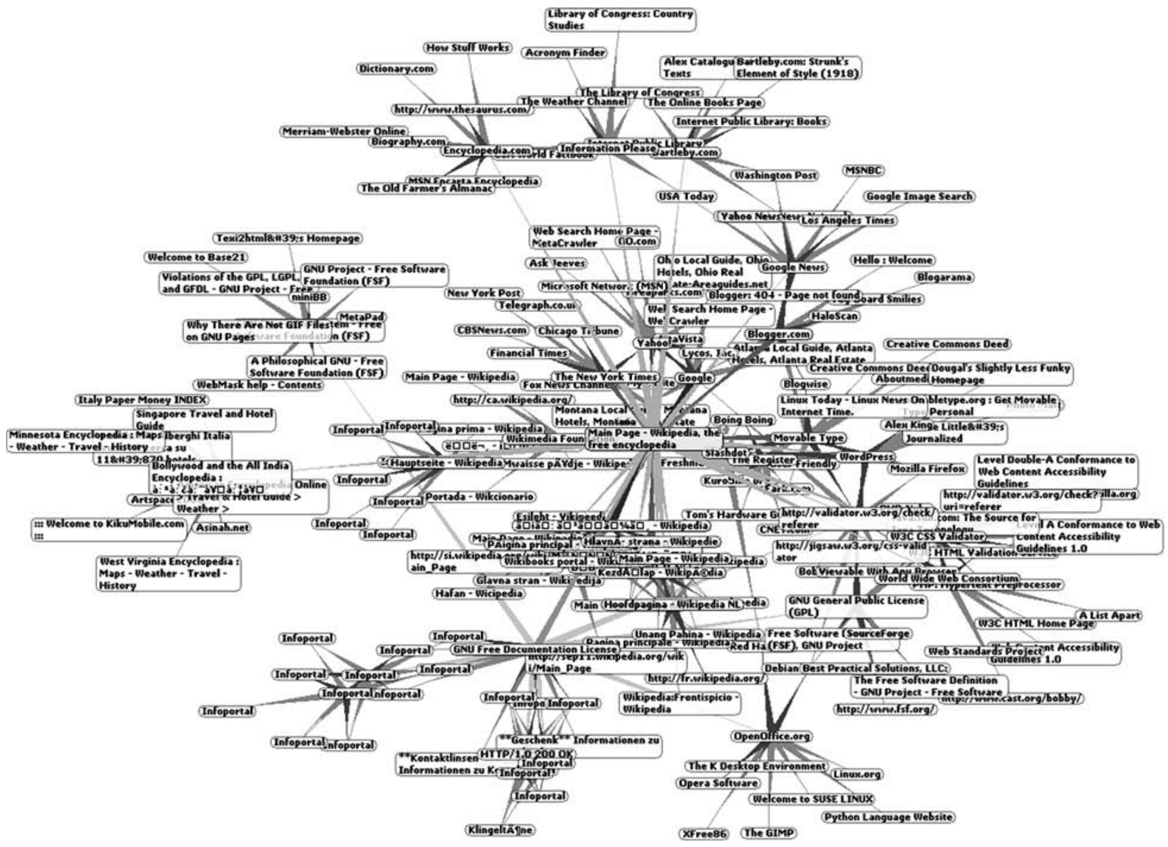
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



Example 10.1.5 Using a Graph to Represent the World Wide Web

The World Wide Web, or Web, is a system of interlinked documents, or webpages, contained on the Internet. Users employing Web browsers, such as Internet Explorer, Google Chrome, Apple Safari, and Opera, can move quickly from one webpage to another by clicking on hyperlinks, which use versions of software called hypertext transfer protocols (HTTPs). Individuals and individual companies create the pages, which they transmit to servers that contain software capable of delivering them to those who request them through a Web browser. Because the amount of information currently on the Web is so vast, search engines, such as Google, Yahoo, and Bing, have algorithms for finding information very efficiently.

The picture below shows a minute fraction of the hyperlink connections on the Internet that radiate in and out from the Wikipedia main page.



Wikipedia/Chris 73

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.1.6 Using a Graph to Represent Knowledge

In many applications of artifical intelligence, a knowledge base of information is collected and represented inside a computer. Because of the way the knowledge is represented and because of the properties that govern the artificial intelligence program, the computer is not limited to retrieving data in the same form as it was entered; it can also derive new facts from the knowledge base by using certain built-in rules of inference. For example, from the knowledge that the *Los Angeles Times* is a big-city daily and that a big-city daily contains national news, an artifical intelligence program could infer that the *Los Angeles Times* contains national news. The directed graph shown in Figure 10.1.2 is a pictorial representation for a simplified knowledge base about periodical publications.

According to this knowledge base, what paper finish does the *New York Times* use?

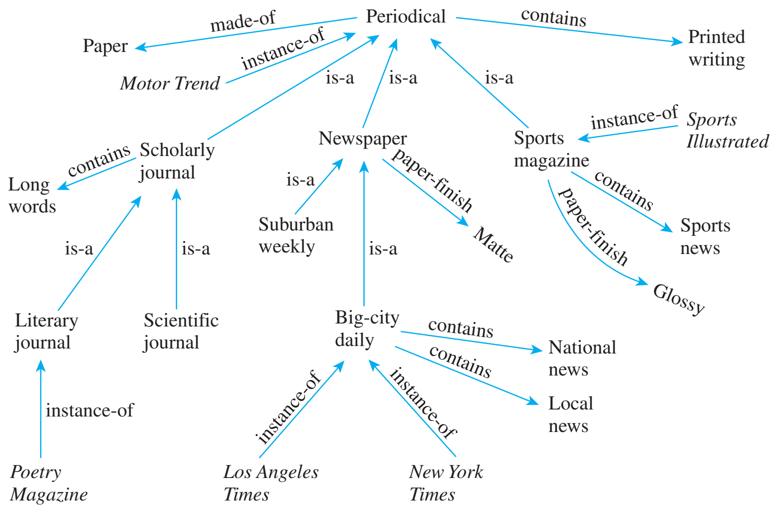


Figure 10.1.2

Solution The arrow going from *New York Times* to big-city daily (labeled "instance-of") shows that the *New York Times* is a big-city daily. The arrow going from big-city daily to newspaper (labeled "is-a") shows that a big-city daily is a newspaper. The arrow going from newspaper to matte (labeled "paper-finish") indicates that the paper finish on a newspaper is matte. Hence it can be inferred that the paper finish on the *New York Times* is matte. ■

Example 10.1.7 Using a Graph to Solve a Problem: Vegetarians and Cannibals

The following is a variation of a famous puzzle often used as an example in the study of artificial intelligence. It concerns an island on which all the people are of one of two types, either vegetarians or cannibals. Initially, two vegetarians and two cannibals are on the left bank of a river. With them is a boat that can hold a maximum of two people. The aim of the puzzle is to find a way to transport all the vegetarians and cannibals to the right bank of the river. What makes this difficult is that at no time can the number of cannibals on either bank outnumber the number of vegetarians. Otherwise, disaster befalls the vegetarians!

Solution A systematic way to approach this problem is to introduce a notation that can indicate all possible arrangements of vegetarians, cannibals, and the boat on the banks of

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

the river. For example, you could write (vvc/Bc) to indicate that there are two vegetarians and one cannibal on the left bank and one cannibal and the boat on the right bank. Then $(vvccB/)$ would indicate the initial position in which both vegetarians, both cannibals, and the boat are on the left bank of the river. The aim of the puzzle is to figure out a sequence of moves to reach the position $(/Bvvcc)$ in which both vegetarians, both cannibals, and the boat are on the right bank of the river.

Construct a graph whose vertices are the various arrangements that can be reached in a sequence of legal moves starting from the initial position. Connect vertex x to vertex y if it is possible to reach vertex y in one legal move from vertex x. For instance, from the initial position there are four legal moves: one vegetarian and one cannibal can take the boat to the right bank; two cannibals can take the boat to the right bank; one cannibal can take the boat to the right bank; or two vegetarians can take the boat to the right bank. You can show these by drawing edges connecting vertex $(vvccB/)$ to vertices (vc/Bvc), (vv/Bcc), $(vvcBc)$, and (cc/Bvv). (It might seem natural to draw directed edges rather than undirected edges from one vertex to another. The rationale for drawing undirected edges is that each legal move is reversible.) From the position (vc/Bvc), the only legal moves are to go back to $(vvccB/)$ or to go to $(vvcB/c)$. You can also show these by drawing in edges. Continue this process until finally you reach $(/Bvvcc)$. From Figure 10.1.3 it is apparent that one successful sequence of moves is $(vvccB/) \to (vc/Bvc) \to (vvcB/c) \to (c/Bvvc) \to (ccB/vv) \to (/Bvvcc)$.

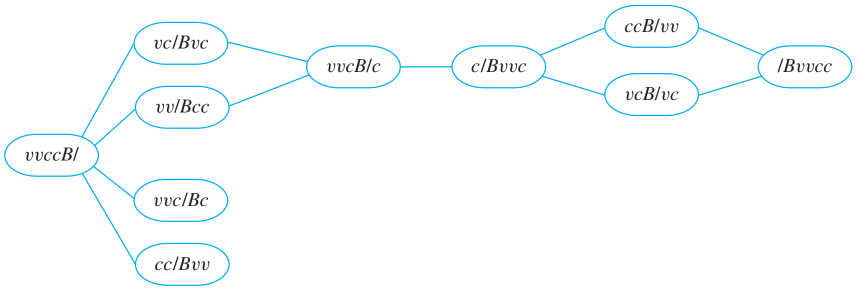


Figure 10.1.3 ■

Special Graphs

One important class of graphs consists of those that do not have any loops or parallel edges. Such graphs are called *simple*. In a simple graph, no two edges share the same set of endpoints, so specifying two endpoints is sufficient to determine an edge.

• Definition and Notation

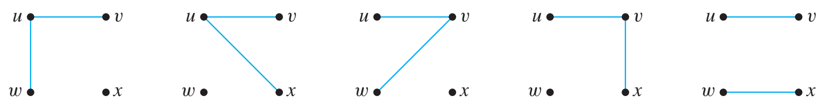
A **simple graph** is a graph that does not have any loops or parallel edges. In a simple graph, an edge with endpoints v and w is denoted $\{v, w\}$.

Example 10.1.8 A Simple Graph

Draw all simple graphs with the four vertices $\{u, v, w, x\}$ and two edges, one of which is $\{u, v\}$.

Solution Each possible edge of a simple graph corresponds to a subset of two vertices. Given four vertices, there are $\binom{4}{2} = 6$ such subsets in all: $\{u, v\}$, $\{u, w\}$, $\{u, x\}$, $\{v, w\}$, $\{v, x\}$, and $\{w, x\}$. Now one edge of the graph is specified to be $\{u, v\}$, so any of the remaining five from this list can be chosen to be the second edge. The possibilities are shown on the next page.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



Another important class of graphs consists of those that are "complete" in the sense that all pairs of vertices are connected by edges.

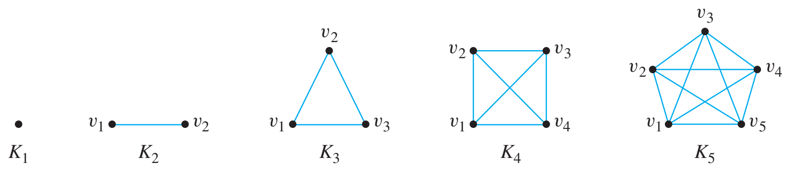
Note The K stands for the German word *komplett*, which means "complete."

• Definition

Let n be a positive integer. A **complete graph on n vertices,** denoted K_n, is a simple graph with n vertices and exactly one edge connecting each pair of distinct vertices.

Example 10.1.9 Complete Graphs on n Vertices: K_1, K_2, K_3, K_4, K_5

The complete graphs K_1, K_2, K_3, K_4, and K_5 can be drawn as follows:



In yet another class of graphs, the vertex set can be separated into two subsets: Each vertex in one of the subsets is connected by exactly one edge to each vertex in the other subset, but not to any vertices in its own subset. Such a graph is called *complete bipartite.*

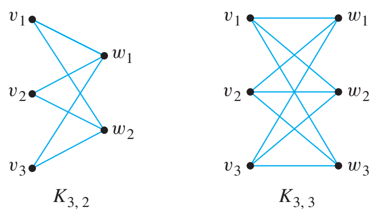
• Definition

Let m and n be positive integers. A **complete bipartite graph on (m, n) vertices,** denoted $K_{m,n}$, is a simple graph with distinct vertices $v_1, v_2, \ldots, v_m$ and $w_1, w_2, \ldots, w_n$ that satisfies the following properties: For all $i, k = 1, 2, \ldots, m$ and for all $j, l = 1, 2, \ldots, n$,

1. There is an edge from each vertex v_i to each vertex w_j.
2. There is no edge from any vertex v_i to any other vertex v_k.
3. There is no edge from any vertex w_j to any other vertex w_l.

Example 10.1.10 Complete Bipartite Graphs: $K_{3,2}$ and $K_{3,3}$

The complete bipartite graphs $K_{3,2}$ and $K_{3,3}$ are illustrated below.



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

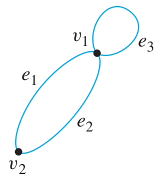
• Definition

A graph H is said to be a **subgraph** of a graph G if, and only if, every vertex in H is also a vertex in G, every edge in H is also an edge in G, and every edge in H has the same endpoints as it has in G.

Example 10.1.11 Subgraphs

List all subgraphs of the graph G with vertex set $\{v_1, v_2\}$ and edge set $\{e_1, e_2, e_3\}$, where the endpoints of e_1 are v_1 and v_2, the endpoints of e_2 are v_1 and v_2, and e_3 is a loop at v_1.

Solution G can be drawn as shown below.



There are 11 subgraphs of G, which can be grouped according to those that do not have any edges, those that have one edge, those that have two edges, and those that have three edges. The 11 subgraphs are shown in Figure 10.1.4.

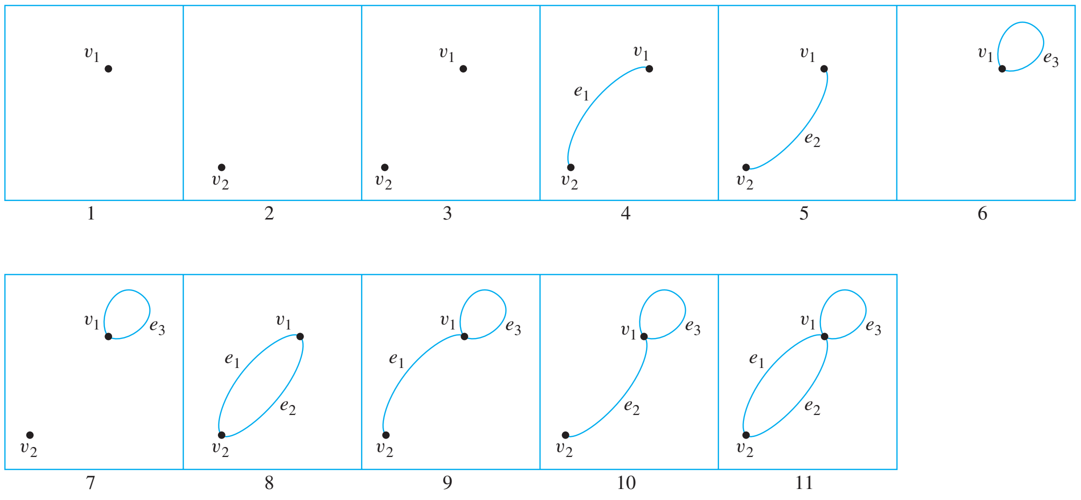


Figure 10.1.4

■

The Concept of Degree

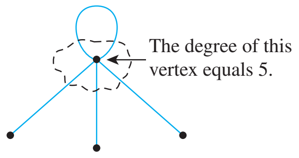
The *degree of a vertex* is the number of end segments of edges that "stick out of" the vertex. We will show that the sum of the degrees of all the vertices in a graph is twice the number of edges of the graph.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

• Definition

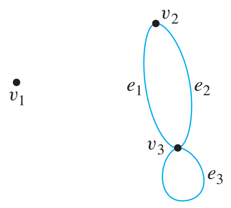
Let G be a graph and v a vertex of G. The **degree of *v*,** denoted **deg(*v*),** equals the number of edges that are incident on v, with an edge that is a loop counted twice. The **total degree of *G*** is the sum of the degrees of all the vertices of G.

Since an edge that is a loop is counted twice, the degree of a vertex can be obtained from the drawing of a graph by counting how many end segments of edges are incident on the vertex. This is illustrated below.



Example 10.1.12 Degree of a Vertex and Total Degree of a Graph

Find the degree of each vertex of the graph G shown below. Then find the total degree of G.



Solution

$\deg(v_1) = 0$ since no edge is incident on v_1 (v_1 is isolated).

$\deg(v_2) = 2$ since both e_1 and e_2 are incident on v_2.

$\deg(v_3) = 4$ since e_1 and e_2 are incident on v_3 and the loop e_3 is also incident on v_3 (and contributes 2 to the degree of v_3).

total degree of $G = \deg(v_1) + \deg(v_2) + \deg(v_3) = 0 + 2 + 4 = 6$. ■

Note that the total degree of the graph G of Example 10.1.12, which is 6, equals twice the number of edges of G, which is 3. Roughly speaking, this is true because each edge has two end segments, and each end segment is counted once toward the degree of some vertex. This result generalizes to any graph.

In fact, for any graph without loops, the general result can be explained as follows: Imagine a group of people at a party. Depending on how social they are, each person shakes hands with various other people. So each person participates in a certain number of handshakes—perhaps many, perhaps none—but because each handshake is experienced by two different people, if the numbers experienced by each person are added together, the sum will equal twice the total number of handshakes. This is such an attractive way of understanding the situation that the following theorem is often called the *handshake lemma* or the *handshake theorem*. As the proof demonstrates, the conclusion is true even if the graph contains loops.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

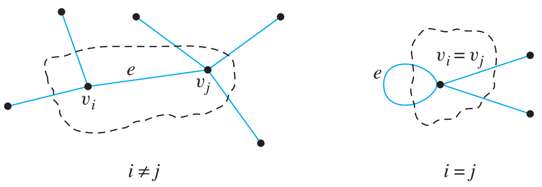
Theorem 10.1.1 The Handshake Theorem

If G is any graph, then the sum of the degrees of all the vertices of G equals twice the number of edges of G. Specifically, if the vertices of G are $v_1, v_2, \ldots, v_n$, where n is a nonnegative integer, then

$$\begin{aligned}\text{the total degree of } G &= \deg(v_1) + \deg(v_2) + \cdots + \deg(v_n)\\ &= 2 \cdot (\text{the number of edges of } G).\end{aligned}$$

Proof:

Let G be a particular but arbitrarily chosen graph, and suppose that G has n vertices $v_1, v_2, \ldots, v_n$ and m edges, where n is a positive integer and m is a nonnegative integer. We claim that each edge of G contributes 2 to the total degree of G. For suppose e is an arbitrarily chosen edge with endpoints v_i and v_j. This edge contributes 1 to the degree of v_i and 1 to the degree v_j. As shown below, this is true even if $i = j$, because an edge that is a loop is counted twice in computing the degree of the vertex on which it is incident.



Therefore, e contributes 2 to the total degree of G. Since e was arbitrarily chosen, this shows that *each* edge of G contributes 2 to the total degree of G. Thus

$$\text{the total degree of } G = 2 \cdot (\text{the number of edges of } G).$$

The following corollary is an immediate consequence of Theorem 10.1.1.

Corollary 10.1.2

The total degree of a graph is even.

Proof:

By Theorem 10.1.1 the total degree of G equals 2 times the number of edges, which is an integer, and so the total degree of G is even.

Example 10.1.13 Determining Whether Certain Graphs Exist

Draw a graph with the specified properties or show that no such graph exists.

a. A graph with four vertices of degrees 1, 1, 2, and 3

b. A graph with four vertices of degrees 1, 1, 3, and 3

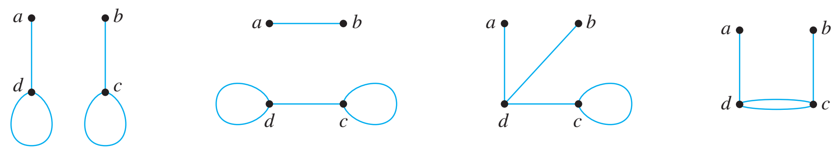
c. A simple graph with four vertices of degrees 1, 1, 3, and 3

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Solution

a. No such graph is possible. By Corollary 10.1.2, the total degree of a graph is even. But a graph with four vertices of degrees 1, 1, 2, and 3 would have a total degree of $1 + 1 + 2 + 3 = 7$, which is odd.

b. Let G be any of the graphs shown below.

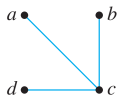


In each case, no matter how the edges are labeled, $\deg(a) = 1$, $\deg(b) = 1$, $\deg(c) = 3$, and $\deg(d) = 3$.

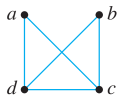
c. There is no simple graph with four vertices of degrees 1, 1, 3, and 3.

Proof (by contradiction):

Suppose there were a simple graph G with four vertices of degrees 1, 1, 3, and 3. Call a and b the vertices of degree 1, and call c and d the vertices of degree 3. Since $\deg(c) = 3$ and G does not have any loops or parallel edges (because it is simple), there must be edges that connect c to a, b, and d.



By the same reasoning, there must be edges connecting d to a, b, and c.



But then $\deg(a) \geq 2$ and $\deg(b) \geq 2$, which contradicts the supposition that these vertices have degree 1. Hence the supposition is false, and consequently there is no simple graph with four vertices of degrees 1, 1, 3, and 3. ■

Example 10.1.14 Application to an Acquaintance Graph

Is it possible in a group of nine people for each to be friends with exactly five others?

Solution The answer is no. Imagine constructing an "acquaintance graph" in which each of the nine people represented by a vertex and two vertices are joined by an edge if, and only if, the people they represent are friends. Suppose each of the people were friends with exactly five others. Then the degree of each of the nine vertices of the graph would be five, and so the total degree of the graph would be 45. But this contradicts Corollary 10.1.2, which says that the total degree of a graph is even. This contradiction shows that the supposition is false, and hence it is impossible for each person in a group of nine people to be friends with exactly five others. ■

The following proposition is easily deduced from Corollary 10.1.2 using properties of even and odd integers.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Proposition 10.1.3

In any graph there are an even number of vertices of odd degree.

Proof:

Suppose G is any graph, and suppose G has n vertices of odd degree and m vertices of even degree, where n is a positive integer and m is a nonnegative integer. *[We must show that n is even.]* Let E be the sum of the degrees of all the vertices of even degree, O the sum of the degrees of all the vertices of odd degree, and T the total degree of G. If $u_1, u_2, \ldots, u_m$ are the vertices of even degree and $v_1, v_2, \ldots, v_n$ are the vertices of odd degree, then

$$\begin{aligned} E &= \deg(u_1) + \deg(u_2) + \cdots + \deg(u_m), \\ O &= \deg(v_1) + \deg(v_2) + \cdots + \deg(v_n), \quad \text{and} \\ T &= \deg(u_1) + \cdots + \deg(u_m) + \deg(v_1) + \cdots + \deg(v_n) = E + O. \end{aligned}$$

Now T, the total degree of G, is an even integer by Corollary 10.1.2. Also E is even since either E is zero, which is even, or E is a sum of the numbers $\deg(u_i)$, each of which is even. But

$$T = E + O,$$

and therefore

$$O = T - E.$$

Hence O is a difference of two even integers, and so O is even.

By assumption, $\deg(v_i)$ is odd for all $i = 1, 2, \ldots, n$. Thus O, an even integer, is a sum of the n odd integers $\deg(v_1), \deg(v_2), \ldots, \deg(v_n)$. But if a sum of n odd integers is even, then n is even. (See exercise 32 at the end of this section.) Therefore, n is even *[as was to be shown]*.

Example 10.1.15 Applying the Fact That the Number of Vertices with Odd Degree Is Even

Is there a graph with ten vertices of degrees 1, 1, 2, 2, 2, 3, 4, 4, 4, and 6?

Solution No. Such a graph would have three vertices of odd degree, which is impossible by Proposition 10.1.3.

Note that this same result could have been deduced directly from Corollary 10.1.2 by computing the total degree $(1 + 1 + 2 + 2 + 2 + 3 + 4 + 4 + 4 + 6 = 29)$ and noting that it is odd. However, use of Proposition 10.1.3 gives the result without the need to perform this addition. ■

Test Yourself

Answers to Test Yourself questions are located at the end of each section.

1. A graph consists of two finite sets: _____ and _____, where each edge is associated with a set consisting of _____.
2. A loop in a graph is _____.
3. Two distinct edges in a graph are parallel if, and only if, _____.
4. Two vertices are called adjacent if, and only if, _____.
5. An edge is incident on _____.
6. Two edges incident on the same endpoint are _____.
7. A vertex on which no edges are incident is _____.
8. In a directed graph, each edge is associated with _____.
9. A simple graph is _____.
10. A complete graph on n vertices is a _____.
11. A complete bipartite graph on (m, n) vertices is a simple graph whose vertices can be partitioned into two disjoint sets

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

V_1 and V_2 in such a way that (1) each of the m vertices in V_1 is _____ to each of the n vertices in V_2, no vertex in V_1 is connected to _____, and no vertex in V_2 is connected to _____.

12. A graph H is a subgraph of a graph G if, and only if, (1) _____, (2) _____, and (3) _____.

13. The degree of a vertex in a graph is _____.

14. The total degree of a graph is defined as _____.

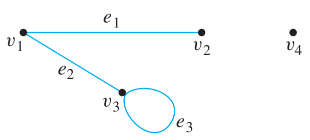
15. The handshake theorem says that the total degree of a graph is _____.

16. In any graph the number of vertices of odd degree is _____.

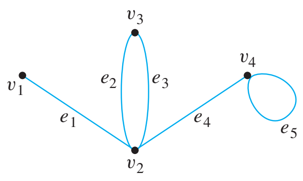
Exercise Set 10.1*

In 1 and 2, graphs are represented by drawings. Define each graph formally by specifying its vertex set, its edge set, and a table giving the edge-endpoint function.

1.



2.



In 3 and 4, draw pictures of the specified graphs.

3. Graph G has vertex set $\{v_1, v_2, v_3, v_4, v_5\}$ and edge set $\{e_1, e_2, e_3, e_4\}$, with edge-endpoint function as follows:

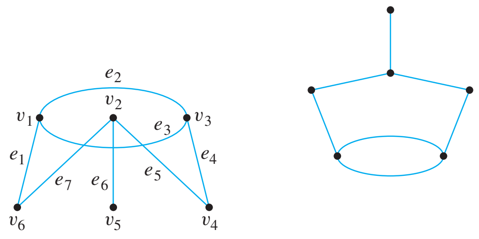
Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_1, v_2\}$
e_3	$\{v_2, v_3\}$
e_4	$\{v_2\}$

4. Graph H has vertex set $\{v_1, v_2, v_3, v_4, v_5\}$ and edge set $\{e_1, e_2, e_3, e_4\}$ with edge-endpoint function as follows:

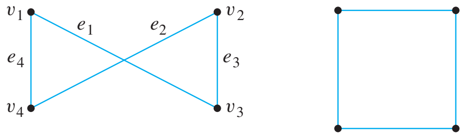
Edge	Endpoints
e_1	$\{v_1\}$
e_2	$\{v_2, v_3\}$
e_3	$\{v_2, v_3\}$
e_4	$\{v_1, v_5\}$

In 5–7, show that the two drawings represent the same graph by labeling the vertices and edges of the right-hand drawing to correspond to those of the left-hand drawing.

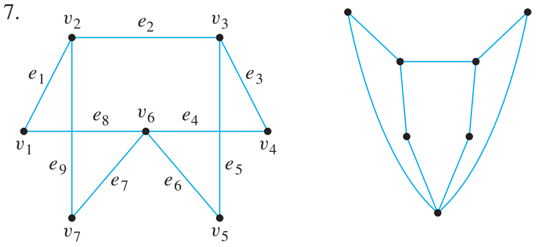
5.



6.



7.



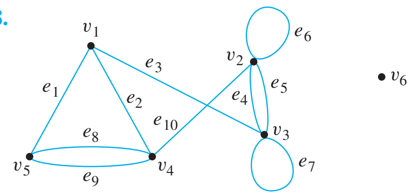
*For exercises with blue numbers or letters, solutions are given in Appendix B. The symbol *H* indicates that only a hint or a partial solution is given. The symbol ✶ signals that an exercise is more challenging than usual.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

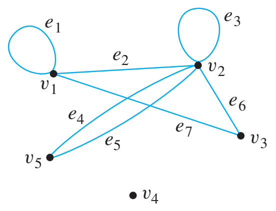
For each of the graphs in 8 and 9:

(i) Find all edges that are incident on v_1.
(ii) Find all vertices that are adjacent to v_3.
(iii) Find all edges that are adjacent to e_1.
(iv) Find all loops.
(v) Find all parallel edges.
(vi) Find all isolated vertices.
(vii) Find the degree of v_3.
(viii) Find the total degree of the graph.

8.



9.



10. Use the graph of Example 10.1.6 to determine
    a. whether *Sports Illustrated* contains printed writing;
    b. whether *Poetry Magazine* contains long words.

11. Find three other winning sequences of moves for the vegetarians and the cannibals in Example 10.1.7.

12. Another famous puzzle used as an example in the study of artificial intelligence seems first to have appeared in a collection of problems, *Problems for the Quickening of the Mind*, which was compiled about A.D. 775. It involves a wolf, a goat, a bag of cabbage, and a ferryman. From an initial position on the left bank of a river, the ferryman is to transport the wolf, the goat, and the cabbage to the right bank. The difficulty is that the ferryman's boat is only big enough for him to transport one object at a time, other than himself. Yet, for obvious reasons, the wolf cannot be left alone with the goat, and the goat cannot be left alone with the cabbage. How should the ferryman proceed?

13. Solve the vegetarians-and-cannibals puzzle for the case where there are three vegetarians and three cannibals to be transported from one side of a river to the other.

H **14.** Two jugs A and B have capacities of 3 quarts and 5 quarts, respectively. Can you use the jugs to measure out exactly 1 quart of water, while obeying the following restrictions? You may fill either jug to capacity from a water tap; you may empty the contents of either jug into a drain; and you may pour water from either jug into the other.

15. A graph has vertices of degrees 0, 2, 2, 3, and 9. How many edges does the graph have?

16. A graph has vertices of degrees 1, 1, 4, 4, and 6. How many edges does the graph have?

In each of 17–25, either draw a graph with the specified properties or explain why no such graph exists.

17. Graph with five vertices of degrees 1, 2, 3, 3, and 5.

18. Graph with four vertices of degrees 1, 2, 3, and 3.

19. Graph with four vertices of degrees 1, 1, 1, and 4.

20. Graph with four vertices of degrees 1, 2, 3, and 4.

21. Simple graph with four vertices of degrees 1, 2, 3, and 4.

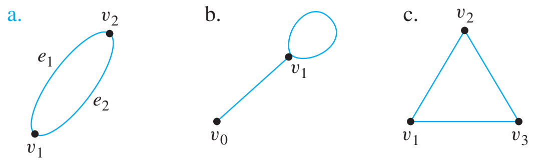
22. Simple graph with five vertices of degrees 2, 3, 3, 3, and 5.

23. Simple graph with five vertices of degrees 1, 1, 1, 2, and 3.

24. Simple graph with six edges and all vertices of degree 3.

25. Simple graph with nine edges and all vertices of degree 3.

26. Find all subgraphs of each of the following graphs.



27. **a.** In a group of 15 people, is it possible for each person to have exactly 3 friends? Explain. (Assume that friendship is a symmetric relationship: If x is a friend of y, then y is a friend of x.)
    b. In a group of 4 people, is it possible for each person to have exactly 3 friends? Why?

28. In a group of 25 people, is it possible for each to shake hands with exactly 3 other people? Explain.

29. Is there a simple graph, each of whose vertices has even degree? Explain.

30. Suppose that G is a graph with v vertices and e edges and that the degree of each vertex is at least $d_{\min}$ and at most $d_{\max}$. Show that

$$\frac{1}{2}d_{\min}\cdot v \leq e \leq \frac{1}{2}d_{\max}\cdot v.$$

31. Prove that any sum of an odd number of odd integers is odd.

H **32.** Deduce from exercise 31 that for any positive integer n, if there is a sum of n odd integers that is even, then n is even.

33. Recall that K_n denotes a complete graph on n vertices.
    a. Draw K_6.
    H **b.** Show that for all integers $n \geq 1$, the number of edges of K_n is $\dfrac{n(n-1)}{2}$.

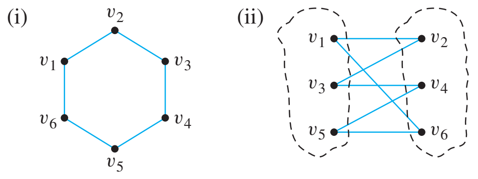
34. Use the result of exercise 33 to show that the number of edges of a simple graph with n vertices is less than or equal to $\dfrac{n(n-1)}{2}$.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

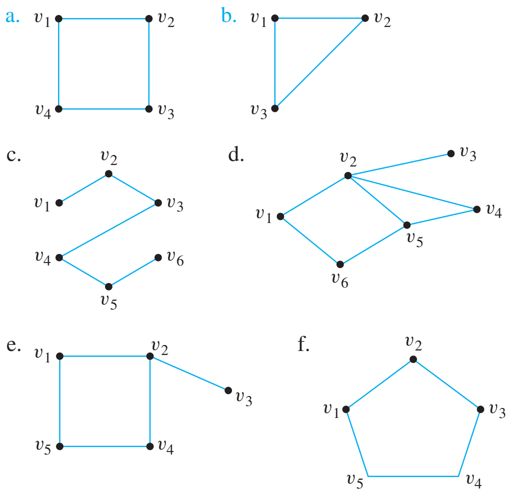
35. Is there a simple graph with twice as many edges as vertices? Explain. (You may find it helpful to use the result of exercise 34.)

36. Recall that $K_{m,n}$ denotes a complete bipartite graph on (m, n) vertices.
   a. Draw $K_{4,2}$
   b. Draw $K_{1,3}$
   c. Draw $K_{3,4}$
   d. How many vertices of $K_{m,n}$ have degree m? degree n?
   e. What is the total degree of $K_{m,n}$?
   f. Find a formula in terms of m and n for the number of edges of $K_{m,n}$. Explain.

37. A **bipartite graph** G is a simple graph whose vertex set can be partitioned into two disjoint nonempty subsets V_1 and V_2 such that vertices in V_1 may be connected to vertices in V_2, but no vertices in V_1 are connected to other vertices in V_1 and no vertices in V_2 are connected to other vertices in V_2. For example, the graph G illustrated in (i) can be redrawn as shown in (ii). From the drawing in (ii), you can see that G is bipartite with mutually disjoint vertex sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$.

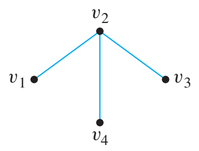

(i)
v_1 v_2 v_3 v_4 v_5 v_6
(ii)
v_1 v_2 v_3 v_4 v_5 v_6

Find which of the following graphs are bipartite. Redraw the bipartite graphs so that their bipartite nature is evident.

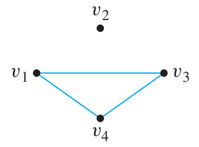

a. v_1 v_2 v_4 v_3
b. v_1 v_2 v_3
c. v_2 v_1 v_3 v_4 v_6 v_5
d. v_2 v_3 v_1 v_4 v_5 v_6
e. v_1 v_2 v_3 v_5 v_4
f. v_2 v_1 v_3 v_5 v_4

38. Suppose r and s are any positive integers. Does there exist a graph G with the property that G has vertices of degrees r and s and of no other degrees? Explain.

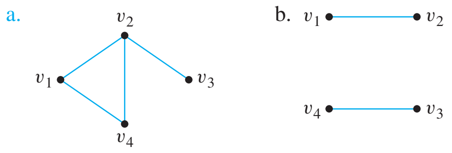
Definition: If G is a simple graph, the **complement of G,** denoted G', is obtained as follows: The vertex set of G' is identical to the vertex set of G. However, two distinct vertices v and w of G' are connected by an edge if, and only if, v and w are not connected by an edge in G. For example, if G is the graph



then G' is



39. Find the complement of each of the following graphs.



40. a. Find the complement of the graph K_4, the complete graph on four vertices. (See Example 10.1.9.)
   b. Find the complement of the graph $K_{3,2}$, the complete bipartite graph on (3, 2) vertices. (See Example 10.1.10.)

41. Suppose that in a group of five people A, B, C, D, and E the following pairs of people are acquainted with each other:
   A and C, A and D, B and C, C and D, C and E.
   a. Draw a graph to represent this situation.
   b. Draw a graph that illustrates who among these five people are *not* acquainted. That is, draw an edge between two people if, and only if, they are not acquainted.

H 42. Let G be a simple graph with n vertices. What is the relation between the number of edges of G and the number of edges of the complement G'?

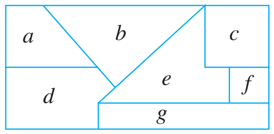
43. Show that at a party with at least two people, there are at least two mutual acquaintances or at least two mutual strangers.

44. a. In a simple graph, must every vertex have degree that is less than the number of vertices in the graph? Why?
   b. Can there be a simple graph that has four vertices each of different degrees?
   H ✶ c. Can there be a simple graph that has n vertices all of different degrees?

H ✶ 45. In a group of two or more people, must there always be at least two people who are acquainted with the same number of people within the group? Why?

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

46. Imagine that the diagram shown below is a map with countries labeled a–g. Is it possible to color the map with only three colors so that no two adjacent countries have the same color? To answer this question, draw and analyze a graph in which each country is represented by a vertex and two vertices are connected by an edge if, and only if, the countries share a common border.

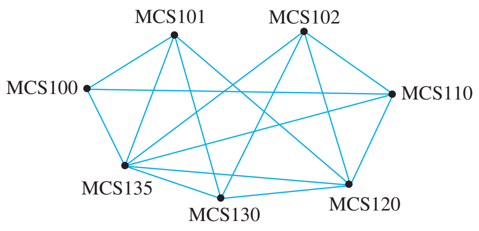


***H* 47.** In this exercise a graph is used to help solve a scheduling problem. Twelve faculty members in a mathematics department serve on the following committees:

Undergraduate Education: Tenner, Peterson, Kashina, Cohen
Graduate Education: Gatto, Yang, Cohen, Catoiu
Colloquium: Sahin, McMurry, Ash
Library: Cortzen, Tenner, Sahin
Hiring: Gatto, McMurry, Yang, Peterson
Personnel: Yang, Wang, Cortzen

The committees must all meet during the first week of classes, but there are only three time slots available. Find a schedule that will allow all faculty members to attend the meetings of all committees on which they serve. To do this, represent each committee as the vertex of a graph, and draw an edge between two vertices if the two committees have a common member. Find a way to color the vertices using only three colors so that no two committees have the same color, and explain how to use the result to schedule the meetings.

48. A department wants to schedule final exams so that no student has more than one exam on any given day. The vertices of the graph below show the courses that are being taken by more than one student, with an edge connecting two vertices if there is a student in both courses. Find a way to color the vertices of the graph with only four colors so that no two adjacent vertices have the same color and explain how to use the result to schedule the final exams.



Answers for Test Yourself

1. a finite, nonempty set of vertices; a finite set of edges; one or two vertices called its endpoints 2. an edge with a single endpoint 3. they have the same set of endpoints 4. they are connected by an edge 5. each of its endpoints 6. adjacent 7. isolated 8. an ordered pair of vertices called its endpoints 9. a graph with no loops or parallel edges 10. simple graph with n vertices whose set of edges contains exactly one edge for each pair of vertices 11. connected by an edge; any other vertex in V_1; any other vertex in V_2 12. every vertex in H is also a vertex in G; every edge in H is also an edge in G; every edge in H has the same endpoints as it has in G 13. the number of edges that are incident on the vertex, with an edge that is a loop counted twice 14. the sum of the degrees of all the vertices of the graph 15. equal to twice the number of edges of the graph 16. an even number

10.2 Trails, Paths, and Circuits

One can begin to reason only when a clear picture has been formed in the imagination.
— W. W. Sawyer, *Mathematician's Delight,* 1943

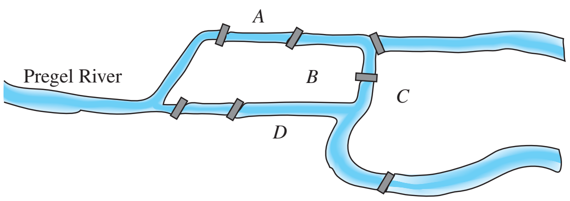
The subject of graph theory began in the year 1736 when the great mathematician Leonhard Euler published a paper giving the solution to the following puzzle:

The town of Königsberg in Prussia (now Kaliningrad in Russia) was built at a point where two branches of the Pregel River came together. It consisted of an island and some land along the river banks. These were connected by seven bridges as shown in Figure 10.2.1.

The question is this: Is it possible for a person to take a walk around town, starting and ending at the same location and crossing each of the seven bridges exactly once?*

*In his original paper, Euler did not require the walk to start and end at the same point. The analysis of the problem is simplified, however, by adding this condition. Later in the section, we discuss walks that start and end at different points.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



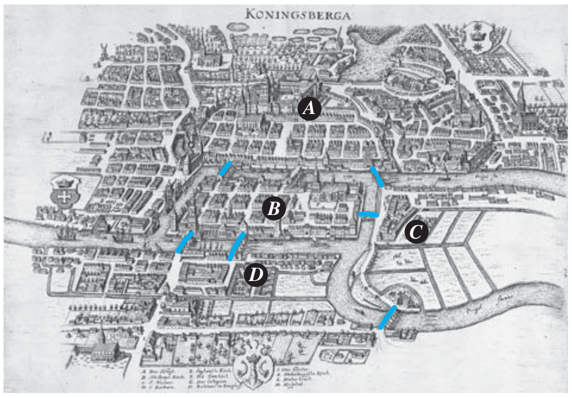


Figure 10.2.1 The Seven Bridges of Königsberg

Leonhard Euler (1707–1783)

To solve this puzzle, Euler translated it into a graph theory problem. He noticed that all points of a given land mass can be identified with each other since a person can travel from any one point to any other point of the same land mass without crossing a bridge. Thus for the purpose of solving the puzzle, the map of Königsberg can be identified with the graph shown in Figure 10.2.2, in which the vertices A, B, C, and D represent land masses and the seven edges represent the seven bridges.

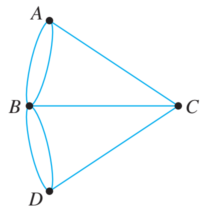


Figure 10.2.2 Graph Version of Königsberg Map

In terms of this graph, the question becomes the following:

> Is it possible to find a route through the graph that starts and ends at some vertex, one of A, B, C, or D, and traverses each edge exactly once?

Equivalently:

> Is it possible to trace this graph, starting and ending at the same point, without ever lifting your pencil from the paper?

Take a few minutes to think about the question yourself. Can you find a route that meets the requirements? Try it!

Looking for a route is frustrating because you continually find yourself at a vertex that does not have an unused edge on which to leave, while elsewhere there are unused edges that must still be traversed. If you start at vertex A, for example, each time you pass through vertex B, C, or D, you use up two edges because you arrive on one edge and depart on a different one. So, if it is possible to find a route that uses all the edges of the graph and starts and ends at A, then the total number of arrivals and departures from each vertex B, C, and D must be a multiple of 2. Or, in other words, the degrees of

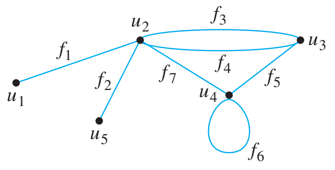
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

the vertices B, C, and D must be even. But they are not: $\deg(B) = 5$, $\deg(C) = 3$, and $\deg(D) = 3$. Hence there is no route that solves the puzzle by starting and ending at A. Similar reasoning can be used to show that there are no routes that solve the puzzle by starting and ending at B, C, or D. Therefore, it is impossible to travel all around the city crossing each bridge exactly once.

Definitions

Travel in a graph is accomplished by moving from one vertex to another along a sequence of adjacent edges. In the graph below, for instance, you can go from u_1 to u_4 by taking f_1 to u_2 and then f_7 to u_4. This is represented by writing

$$u_1 f_1 u_2 f_7 u_4.$$



Or you could take the roundabout route

$$u_1 f_1 u_2 f_3 u_3 f_4 u_2 f_3 u_3 f_5 u_4 f_6 u_4 f_7 u_2 f_3 u_3 f_5 u_4.$$

Certain types of sequences of adjacent vertices and edges are of special importance in graph theory: those that do not have a repeated edge, those that do not have a repeated vertex, and those that start and end at the same vertex.

• Definition

Let G be a graph, and let v and w be vertices in G.

A **walk from *v* to *w*** is a finite alternating sequence of adjacent vertices and edges of G. Thus a walk has the form

$$v_0 e_1 v_1 e_2 \cdots v_{n-1} e_n v_n,$$

where the v's represent vertices, the e's represent edges, $v_0 = v$, $v_n = w$, and for all $i = 1, 2, \ldots n$, v_{i-1} and v_i are the endpoints of e_i. The **trivial walk from *v* to *v*** consists of the single vertex v.

A **trail from *v* to *w*** is a walk from v to w that does not contain a repeated edge.

A **path from *v* to *w*** is a trail that does not contain a repeated vertex.

A **closed walk** is a walk that starts and ends at the same vertex.

A **circuit** is a closed walk that contains at least one edge and does not contain a repeated edge.

A **simple circuit** is a circuit that does not have any other repeated vertex except the first and last.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

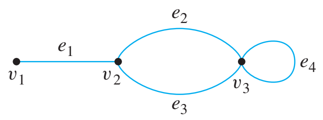
For ease of reference, these definitions are summarized in the following table:

	Repeated Edge?	Repeated Vertex?	Starts and Ends at Same Point?	Must Contain at Least One Edge?
Walk	allowed	allowed	allowed	no
Trail	no	allowed	allowed	no
Path	no	no	no	no
Closed walk	allowed	allowed	yes	no
Circuit	no	allowed	yes	yes
Simple circuit	no	first and last only	yes	yes

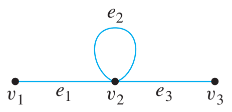
Often a walk can be specified unambiguously by giving either a sequence of edges or a sequence of vertices. The next two examples show how this is done.

Example 10.2.1 Notation for Walks

a. In the graph below, the notation $e_1e_2e_4e_3$ refers unambiguously to the following walk: $v_1e_1v_2e_2v_3e_4v_3e_3v_2$. On the other hand, the notation e_1 is ambiguous if used to refer to a walk. It could mean either $v_1e_1v_2$ or $v_2e_1v_1$.



b. In the graph of part (a), the notation v_2v_3 is ambiguous if used to refer to a walk. It could mean $v_2e_2v_3$ or $v_2e_3v_3$. On the other hand, in the graph below, the notation $v_1v_2v_2v_3$ refers unambiguously to the walk $v_1e_1v_2e_2v_2e_3v_3$.



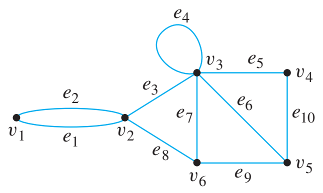
■

Note that if a graph G does not have any parallel edges, then any walk in G is uniquely determined by its sequence of vertices.

Example 10.2.2 Walks, Trails Paths, and Circuits

In the graph below, determine which of the following walks are trails, paths, circuits, or simple circuits.

a. $v_1e_1v_2e_3v_3e_4v_3e_5v_4$ b. $e_1e_3e_5e_5e_6$ c. $v_2v_3v_4v_5v_3v_6v_2$
d. $v_2v_3v_4v_5v_6v_2$ e. $v_1e_1v_2e_1v_1$ f. v_1



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Solution

a. This walk has a repeated vertex but does not have a repeated edge, so it is a trail from v_1 to v_4 but not a path.

b. This is just a walk from v_1 to v_5. It is not a trail because it has a repeated edge.

c. This walk starts and ends at v_2, contains at least one edge, and does not have a repeated edge, so it is a circuit. Since the vertex v_3 is repeated in the middle, it is not a simple circuit.

d. This walk starts and ends at v_2, contains at least one edge, does not have a repeated edge, and does not have a repeated vertex. Thus it is a simple circuit.

e. This is just a closed walk starting and ending at v_1. It is not a circuit because edge e_1 is repeated.

f. The first vertex of this walk is the same as its last vertex, but it does not contain an edge, and so it is not a circuit. It is a closed walk from v_1 to v_1. (It is also a trail from v_1 to v_1.) ■

Because most of the major developments in graph theory have happened relatively recently and in a variety of different contexts, the terms used in the subject have not been standardized. For example, what this book calls a *graph* is sometimes called a *multigraph*, what this book calls a *simple graph* is sometimes called a *graph*, what this book calls a *vertex* is sometimes called a *node*, and what this book calls an *edge* is sometimes called an *arc*. Similarly, instead of the word *trail*, the word *path* is sometimes used; instead of the word *path*, the words *simple path* are sometimes used; and instead of the words *simple circuit*, the word *cycle* is sometimes used. The terminology in this book is among the most common, but if you consult other sources, be sure to check their definitions.

Connectedness

It is easy to understand the concept of connectedness on an intuitive level. Roughly speaking, a graph is connected if it is possible to travel from any vertex to any other vertex along a sequence of adjacent edges of the graph. The formal definition of connectedness is stated in terms of walks.

• Definition

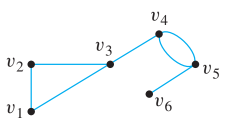
Let G be a graph. Two **vertices *v* and *w* of *G* are connected** if, and only if, there is a walk from v to w. The **graph *G* is connected** if, and only if, given *any* two vertices v and w in G, there is a walk from v to w. Symbolically,

$$G \text{ is connected} \quad \Leftrightarrow \quad \forall \text{ vertices } v, w \in V(G), \exists \text{ a walk from } v \text{ to } w.$$

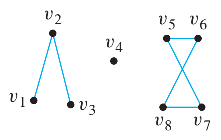
If you take the negation of this definition, you will see that a graph G is *not connected* if, and only if, there are two vertices of G that are not connected by any walk.

Example 10.2.3 Connected and Disconnected Graphs

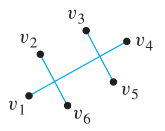
Which of the following graphs are connected?



(a)



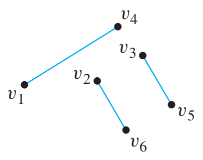
(b)



(c)

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Solution The graph represented in (a) is connected, whereas those of (b) and (c) are not. To understand why (c) is not connected, recall that in a drawing of a graph, two edges may cross at a point that is not a vertex. Thus the graph in (c) can be redrawn as follows:



■

Some useful facts relating circuits and connectedness are collected in the following lemma. Proofs of (a) and (b) are left for the exercises. The proof of (c) is in Section 10.5.

Lemma 10.2.1

Let G be a graph.

a. If G is connected, then any two distinct vertices of G can be connected by a path.

b. If vertices v and w are part of a circuit in G and one edge is removed from the circuit, then there still exists a trail from v to w in G.

c. If G is connected and G contains a circuit, then an edge of the circuit can be removed without disconnecting G.

Look back at Example 10.2.3. The graphs in (b) and (c) are both made up of three pieces, each of which is itself a connected graph. A *connected component* of a graph is a connected subgraph of largest possible size.

• Definition

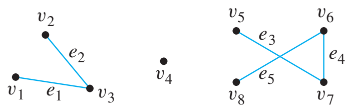
A graph H is a **connected component** of a graph G if, and only if,

1. H is subgraph of G;
2. H is connected; and
3. no connected subgraph of G has H as a subgraph and contains vertices or edges that are not in H.

The fact is that any graph is a kind of union of its connected components.

Example 10.2.4 Connected Components

Find all connected components of the following graph G.



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Solution G has three connected components: H_1, H_2, and H_3 with vertex sets V_1, V_2, and V_3 and edge sets E_1, E_2, and E_3, where

$$\begin{aligned} V_1 &= \{v_1, v_2, v_3\}, & E_1 &= \{e_1, e_2\}, \\ V_2 &= \{v_4\}, & E_2 &= \emptyset, \\ V_3 &= \{v_5, v_6, v_7, v_8\}, & E_3 &= \{e_3, e_4, e_5\}. \end{aligned}$$

■

Euler Circuits

Now we return to consider general problems similar to the puzzle of the Königsberg bridges. The following definition is made in honor of Euler.

• Definition

Let G be a graph. An **Euler circuit** for G is a circuit that contains every vertex and every edge of G. That is, an Euler circuit for G is a sequence of adjacent vertices and edges in G that has at least one edge, starts and ends at the same vertex, uses every vertex of G at least once, and uses every edge of G exactly once.

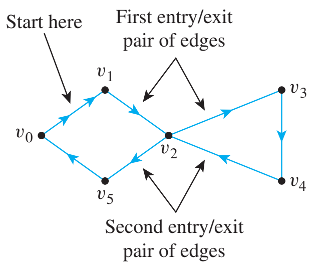
The analysis used earlier to solve the puzzle of the Königsberg bridges generalizes to prove the following theorem:

Theorem 10.2.2

If a graph has an Euler circuit, then every vertex of the graph has positive even degree.

Proof:

Suppose G is a graph that has an Euler circuit. *[We must show that given any vertex v of G, the degree of v is even.]* Let v be any particular but arbitrarily chosen vertex of G. Since the Euler circuit contains every edge of G, it contains all edges incident on v. Now imagine taking a journey that begins in the middle of one of the edges adjacent to the start of the Euler circuit and continues around the Euler circuit to end in the middle of the starting edge. (See Figure 10.2.3. There is such a starting edge because the Euler circuit has at least one edge.) Each time v is entered by traveling along one edge, it is immediately exited by traveling along another edge (since the journey ends in the *middle* of an edge).


Start here
First entry/exit pair of edges
v_1
v_3
v_0
v_2
v_5
v_4
Second entry/exit pair of edges

In this example, the Euler circuit is $v_0v_1v_2v_3v_4v_5v_0$, and v is v_2. Each time v_2 is entered by one edge, it is exited by another edge.

Figure 10.2.3 **Example for the Proof of Theorem 10.2.2**

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Because the Euler circuit uses every edge of G exactly once, every edge incident on v is traversed exactly once in this process. Hence the edges incident on v occur in entry/exit pairs, and consequently the degree of v must be a positive multiple of 2. But that means that v has positive even degree *[as was to be shown].*

Recall that the contrapositive of a statement is logically equivalent to the statement. The contrapositive of Theorem 10.2.2 is as follows:

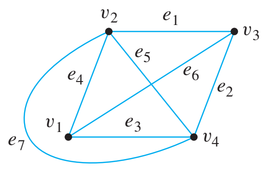
Contrapositive Version of Theorem 10.2.2

If some vertex of a graph has odd degree, then the graph does not have an Euler circuit.

This version of Theorem 10.2.2 is useful for showing that a given graph does *not* have an Euler circuit.

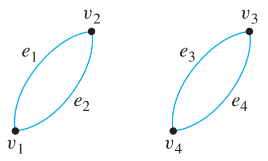
Example 10.2.5 Showing That a Graph Does Not Have an Euler Circuit

Show that the graph below does not have an Euler circuit.



Solution Vertices v_1 and v_3 both have degree 3, which is odd. Hence by (the contrapositive form of) Theorem 10.2.2, this graph does not have an Euler circuit. ■

Now consider the converse of Theorem 10.2.2: If every vertex of a graph has even degree, then the graph has an Euler circuit. Is this true? The answer is no. There is a graph G such that every vertex of G has even degree but G does not have an Euler circuit. In fact, there are many such graphs. The illustration below shows one example.



Every vertex has even degree, but the graph does not have an Euler circuit.

Note that the graph in the preceding drawing is not connected. It turns out that although the converse of Theorem 10.2.2 is false, a modified converse is true: If every vertex of a graph has positive even degree *and* if the graph is connected, then the graph has an Euler circuit. The proof of this fact is constructive: It contains an algorithm to find an Euler circuit for any connected graph in which every vertex has even degree.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Theorem 10.2.3

If a graph G is connected and the degree of every vertex of G is a positive even integer, then G has an Euler circuit.

Proof:

Suppose that G is any connected graph and suppose that every vertex of G is a positive even integer. *[We must find an Euler circuit for G.]* Construct a circuit C by the following algorithm:

Step 1: Pick any vertex v of G at which to start.
[This step can be accomplished because the vertex set of G is nonempty by assumption.]

Step 2: Pick any sequence of adjacent vertices and edges, starting and ending at v and never repeating an edge. Call the resulting circuit C.
[This step can be performed for the following reasons: Since the degree of each vertex of G is a positive even integer, as each vertex of G is entered by traveling on one edge, either the vertex is v itself and there is no other unused edge adjacent to v, or the vertex can be exited by traveling on another previously unused edge. Since the number of edges of the graph is finite (by definition of graph), the sequence of distinct edges cannot go on forever. The sequence can eventually return to v because the degree of v is a positive even integer, and so if an edge connects v to another vertex, there must be a different edge that connects back to v.]

Step 3: Check whether C contains every edge and vertex of G. If so, C is an Euler circuit, and we are finished. If not, perform the following steps.

Step 3a: Remove all edges of C from G and also any vertices that become isolated when the edges of C are removed. Call the resulting subgraph G'.
[Note that G' may not be connected (as illustrated in Figure 10.2.4), but every vertex of G' has positive, even degree (since removing the edges of C removes an even number of edges from each vertex, the difference of two even integers is even, and isolated vertices with degree 0 were removed.)]

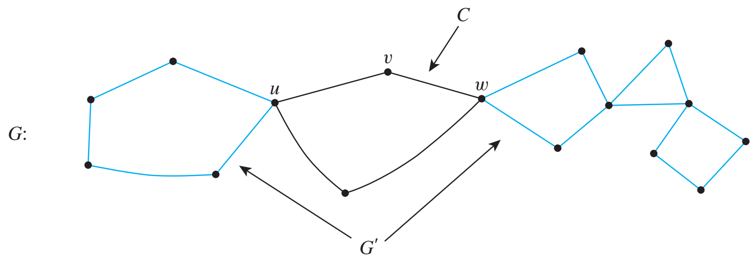


Figure 10.2.4

Step 3b: Pick any vertex w common to both C and G'.
[There must be at least one such vertex since G is connected. (See exercise 44.) (In Figure 10.2.4 there are two such vertices: u and w.)]

Step 3c: Pick any sequence of adjacent vertices and edges of G', starting and ending at w and never repeating an edge. Call the resulting circuit C'.
[This can be done since each vertex of G' has positive, even degree and G' is finite. See the justification for step 2.]

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Step 3d: Patch C and C' together to create a new circuit C'' as follows: Start at v and follow C all the way to w. Then follow C' all the way back to w. After that, continue along the untraveled portion of C to return to v. *[The effect of executing steps 3c and 3d for the graph of Figure 10.2.4 is shown in Figure 10.2.5.]*

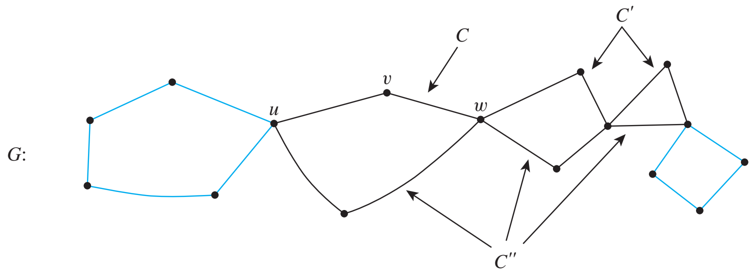


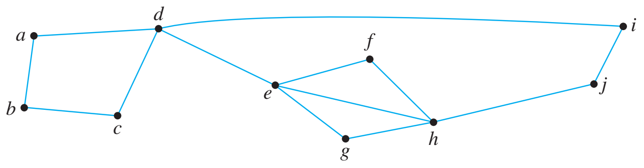
Figure 10.2.5

Step 3e: Let $C = C''$ and go back to step 3.

Since the graph G is finite, execution of the steps outlined in this algorithm must eventually terminate. At that point an Euler circuit for G will have been constructed. (Note that because of the element of choice in steps 1, 2, 3b, and 3c, a variety of different Euler circuits can be produced by using this algorithm.)

Example 10.2.6 Finding an Euler Circuit

Use Theorem 10.2.3 to check that the graph below has an Euler circuit. Then use the algorithm from the proof of the theorem to find an Euler circuit for the graph.



Solution Observe that

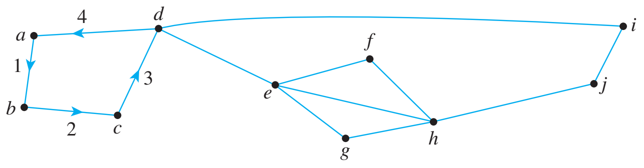
$$\deg(a) = \deg(b) = \deg(c) = \deg(f) = \deg(g) = \deg(i) = \deg(j) = 2$$

and that $\deg(d) = \deg(e) = \deg(h) = 4$. Hence all vertices have even degree. Also, the graph is connected. Thus, by Theorem 10.2.3, the graph has an Euler circuit.

To construct an Euler circuit using the algorithm of Theorem 10.2.3, let $v = a$ and let C be

$$C: abcda.$$

C is represented by the labeled edges shown below.



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

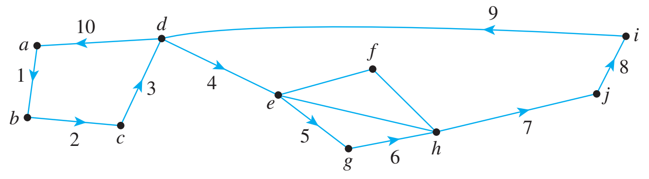
Observe that C is not an Euler circuit for the graph but that C intersects the rest of the graph at d. Let C' be

$$C'\colon deghjid.$$

Patch C' into C to obtain

$$C''\colon abcdeghjida.$$

Set $C = C''$. Then C is represented by the labeled edges shown below.



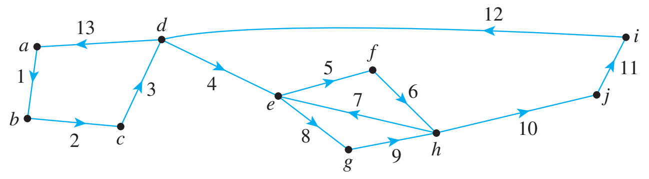
Observe that C is not an Euler circuit for the graph but that it intersects the rest of the graph at e. Let C' be

$$C'\colon efhe.$$

Patch C' into C to obtain

$$C''\colon abcdefheghjida.$$

Set $C = C''$. Then C is represented by the labeled edges shown below.



Since C includes every edge of the graph exactly once, C is an Euler circuit for the graph. ■

In exercise 45 at the end of this section you are asked to show that any graph with an Euler circuit is connected. This result can be combined with Theorems 10.2.2 and 10.2.3 to give a complete characterization of graphs that have Euler circuits, as stated in Theorem 10.2.4.

Theorem 10.2.4

A graph G has an Euler circuit if, and only if, G is connected and every vertex of G has positive even degree.

A corollary to Theorem 10.2.4 gives a criterion for determining when it is possible to find a walk from one vertex of a graph to another, passing through every vertex of the graph at least once and every edge of the graph exactly once.

• Definition

Let G be a graph, and let v and w be two distinct vertices of G. An **Euler trail from *v* to *w*** is a sequence of adjacent edges and vertices that starts at v, ends at w, passes through every vertex of G at least once, and traverses every edge of G exactly once.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

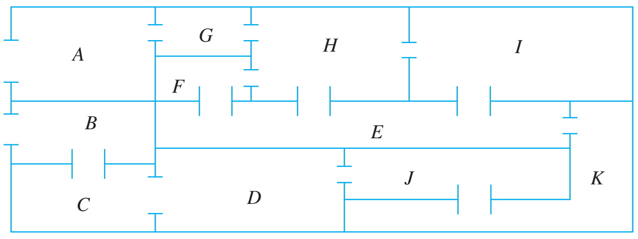
Corollary 10.2.5

Let G be a graph, and let v and w be two distinct vertices of G. There is an Euler path from v to w if, and only if, G is connected, v and w have odd degree, and all other vertices of G have positive even degree.

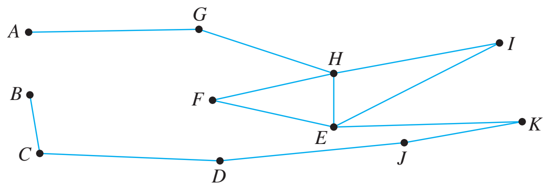
The proof of this corollary is left as an exercise.

Example 10.2.7 Finding an Euler Trail

The floor plan shown below is for a house that is open for public viewing. Is it possible to find a trail that starts in room A, ends in room B, and passes through every interior doorway of the house exactly once? If so, find such a trail.



Solution Let the floor plan of the house be represented by the graph below.



Each vertex of this graph has even degree except for A and B, each of which has degree 1. Hence by Corollary 10.2.5, there is an Euler path from A to B. One such trail is

$$AGHFEIHEKJDCB.$$

■

Hamiltonian Circuits

Sir Wm. Hamilton (1805–1865)

Theorem 10.2.4 completely answers the following question: Given a graph G, is it possible to find a circuit for G in which all the *edges* of G appear exactly once? A related question is this: Given a graph G, is it possible to find a circuit for G in which all the *vertices* of G (except the first and the last) appear exactly once?

In 1859 the Irish mathematician Sir William Rowan Hamilton introduced a puzzle in the shape of a dodecahedron (DOH-dek-a-HEE-dron). (Figure 10.2.6 contains a drawing of a dodecahedron, which is a solid figure with 12 identical pentagonal faces.)

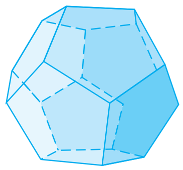
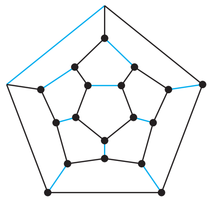


Figure 10.2.6 Dodecahedron

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Each vertex was labeled with the name of a city—London, Paris, Hong Kong, New York, and so on. The problem Hamilton posed was to start at one city and tour the world by visiting each other city exactly once and returning to the starting city. One way to solve the puzzle is to imagine the surface of the dodecahedron stretched out and laid flat in the plane, as follows:



The circuit denoted with black lines is one solution. Note that although every city is visited, many edges are omitted from the circuit. (More difficult versions of the puzzle required that certain cities be visited in a certain order.)

The following definition is made in honor of Hamilton.

• Definition

Given a graph G, a **Hamiltonian circuit** for G is a simple circuit that includes every vertex of G. That is, a Hamiltonian circuit for G is a sequence of adjacent vertices and distinct edges in which every vertex of G appears exactly once, except for the first and the last, which are the same.

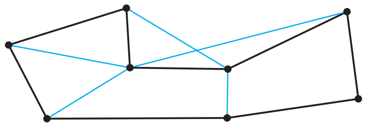
Note that although an Euler circuit for a graph G must include every vertex of G, it may visit some vertices more than once and hence may not be a Hamiltonian circuit. On the other hand, a Hamiltonian circuit for G does not need to include all the edges of G and hence may not be an Euler circuit.

Despite the analogous-sounding definitions of Euler and Hamiltonian circuits, the mathematics of the two are very different. Theorem 10.2.4 gives a simple criterion for determining whether a given graph has an Euler circuit. Unfortunately, there is no analogous criterion for determining whether a given graph has a Hamiltonian circuit, nor is there even an efficient algorithm for finding such a circuit. There is, however, a simple technique that can be used in many cases to show that a graph does *not* have a Hamiltonian circuit. This follows from the following considerations:

Suppose a graph G with at least two vertices has a Hamiltonian circuit C given concretely as

$$C\colon v_0 e_1 v_1 e_2 \cdots v_{n-1} e_n v_n.$$

Since C is a simple circuit, all the e_i are distinct and all the v_j are distinct except that $v_0 = v_n$. Let H be the subgraph of G that is formed using the vertices and edges of C. An example of such an H is shown below.



H is indicated by the black lines.

Note that H has the same number of edges as it has vertices since all its n edges are distinct and so are its n vertices $v_1, v_2, \ldots, v_n$. Also, by definition of Hamiltonian circuit,

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

every vertex of G is a vertex of H, and H is connected since any two of its vertices lie on a circuit. In addition, every vertex of H has degree 2. The reason for this is that there are exactly two edges incident on any vertex. These are e_i and e_{i+1} for any vertex v_i except $v_0 = v_n$, and they are e_1 and e_n for v_0 $(= v_n)$. These observations have established the truth of the following proposition in all cases where G has at least two vertices.

Proposition 10.2.6

If a graph G has a Hamiltonian circuit, then G has a subgraph H with the following properties:

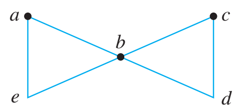
1. H contains every vertex of G.
2. H is connected.
3. H has the same number of edges as vertices.
4. Every vertex of H has degree 2.

Note that if G contains only one vertex and G has a Hamiltonian circuit, then the circuit has the form $v\ e\ v$, where v is the vertex of G and e is an edge incident on v. In this case, the subgraph H consisting of v and e satisfies conditions (1)–(4) of Proposition 10.2.6.

Recall that the contrapositive of a statement is logically equivalent to the statement. The contrapositive of Proposition 10.2.6 says that if a graph G does *not* have a subgraph H with properties (1)–(4), then G does *not* have a Hamiltonian circuit.

Example 10.2.8 Showing That a Graph Does Not Have a Hamiltonian Circuit

Prove that the graph G shown below does not have a Hamiltonian circuit.



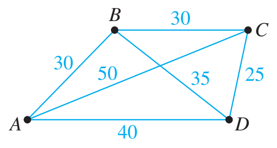
Solution If G has a Hamiltonian circuit, then by Proposition 10.2.6, G has a subgraph H that (1) contains every vertex of G, (2) is connected, (3) has the same number of edges as vertices, and (4) is such that every vertex has degree 2. Suppose such a subgraph H exists. In other words, suppose there is a connected subgraph H of G such that H has five vertices (a, b, c, d, e) and five edges and such that every vertex of H has degree 2. Since the degree of b in G is 4 and every vertex of H has degree 2, two edges incident on b must be removed from G to create H. Edge $\{a, b\}$ cannot be removed because if it were, vertex a would have degree less than 2 in H. Similar reasoning shows that edges $\{e, b\}$, $\{b, a\}$, and $\{b, d\}$ cannot be removed either. It follows that the degree of b in H must be 4, which contradicts the condition that every vertex in H has degree 2 in H. Hence no such subgraph H exists, and so G does not have a Hamiltonian circuit. ■

The next example illustrates a type of problem known as a **traveling salesman problem.** It is a variation of the problem of finding a Hamiltonian circuit for a graph.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.2.9 A Traveling Salesman Problem

Imagine that the drawing below is a map showing four cities and the distances in kilometers between them. Suppose that a salesman must travel to each city exactly once, starting and ending in city A. Which route from city to city will minimize the total distance that must be traveled?



Solution This problem can be solved by writing all possible Hamiltonian circuits starting and ending at A and calculating the total distance traveled for each.

Route	Total Distance (In Kilometers)
$ABCDA$	$30 + 30 + 25 + 40 = 125$
$ABDCA$	$30 + 35 + 25 + 50 = 140$
$ACBDA$	$50 + 30 + 35 + 40 = 155$
$ACDBA$	140 [$ABDCA$ backwards]
$ADBCA$	155 [$ACBDA$ backwards]
$ADCBA$	125 [$ABCDA$ backwards]

Thus either route $ABCDA$ or $ADCBA$ gives a minimum total distance of 125 kilometers. ■

The general traveling salesman problem involves finding a Hamiltonian circuit to minimize the total distance traveled for an arbitrary graph with n vertices in which each edge is marked with a distance. One way to solve the general problem is to use the method of Example 10.2.9: Write down all Hamiltonian circuits starting and ending at a particular vertex, compute the total distance for each, and pick one for which this total is minimal. However, even for medium-sized values of n this method is impractical. For a complete graph with 30 vertices, there would be $(29!)/2 \cong 4.42 \times 10^{30}$ Hamiltonian circuits starting and ending at a particular vertex to check. Even if each circuit could be found and its total distance computed in just one nanosecond, it would require approximately 1.4×10^{14} years to finish the computation. At present, there is no known algorithm for solving the general traveling salesman problem that is more efficient. However, there are efficient algorithms that find "pretty good" solutions—that is, circuits that, while not necessarily having the least possible total distances, have smaller total distances than most other Hamiltonian circuits.

Test Yourself

1. Let G be a graph and let v and w be vertices in G.
   (a) A walk from v to w is _____.
   (b) A trail from v to w is _____.
   (c) A path from v to w is _____.
   (d) A closed walk is _____.
   (e) A circuit is _____.
   (f) A simple circuit is _____.
   (g) A trivial walk is _____.
   (h) Vertices v and w are connected if, and only if, _____.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

2. A graph is connected if, and only if, _____.

3. Removing an edge from a circuit in a graph does not _____.

4. An Euler circuit in a graph is _____.

5. A graph has an Euler circuit if, and only if, _____.

6. Given vertices v and w in a graph, there is an Euler path from v to w if, and only if, _____.

7. A Hamiltonian circuit in a graph is _____.

8. If a graph G has a Hamiltonian circuit, then G has a subgraph H with the following properties: _____, _____, _____, and _____.

9. A traveling salesman problem involves finding a _____ that minimizes the total distance traveled for a graph in which each edge is marked with a distance.

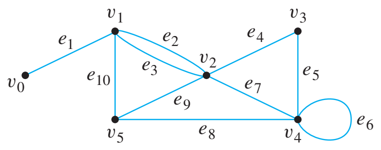
Exercise Set 10.2

1. In the graph below, determine whether the following walks are trails, paths, closed walks, circuits, simple circuits, or just walks.

a. $v_0e_1v_1e_{10}v_5e_9v_2e_2v_1$ b. $v_4e_7v_2e_9v_5e_{10}v_1e_3v_2e_9v_5$

c. v_2 d. $v_5v_2v_3v_4v_4v_5$

e. $v_2v_3v_4v_5v_2v_4v_3v_2$ f. $e_5e_8e_{10}e_3$

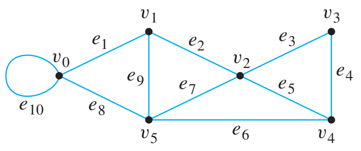


2. In the graph below, determine whether the following walks are trails, paths, closed walks, circuits, simple circuits, or just walks.

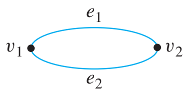
a. $v_1e_2v_2e_3v_3e_4v_4e_5v_2e_2v_1e_1v_0$ b. $v_2v_3v_4v_5v_2$

c. $v_4v_2v_3v_4v_5v_2v_4$ d. $v_2v_1v_5v_2v_3v_4v_2$

e. $v_0v_5v_2v_3v_4v_2v_1$ f. $v_5v_4v_2v_1$



3. Let G be the graph

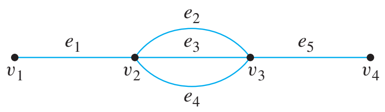


and consider the walk $v_1e_1v_2e_2v_1$.

a. Can this walk be written unambiguously as $v_1v_2v_1$? Why?

b. Can this walk be written unambiguously as e_1e_2? Why?

4. Consider the following graph.

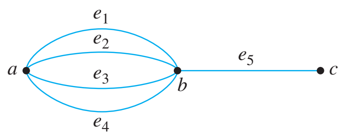


a. How many paths are there from v_1 to v_4?

b. How many trails are there from v_1 to v_4?

c. How many walks are there from v_1 to v_4?

5. Consider the following graph.



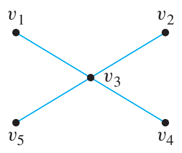
a. How many paths are there from a to c?

b. How many trails are there from a to c?

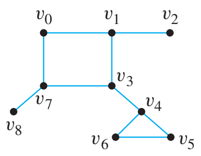
c. How many walks are there from a to c?

6. An edge whose removal disconnects the graph of which it is a part is called a **bridge.** Find all bridges for each of the following graphs.

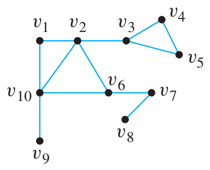
a.



b.



c.

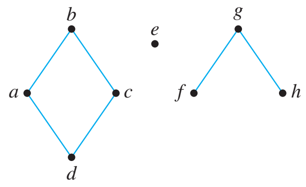


7. Given any positive integer n, (a) find a connected graph with n edges such that removal of just one edge disconnects the graph; (b) find a connected graph with n edges that cannot be disconnected by the removal of any single edge.

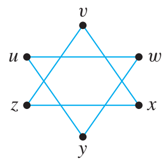
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

8. Find the number of connected components for each of the following graphs.

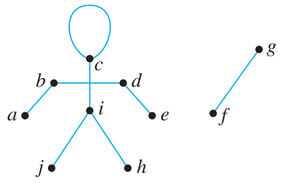
a.



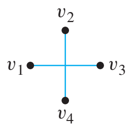
b.



c.



d.



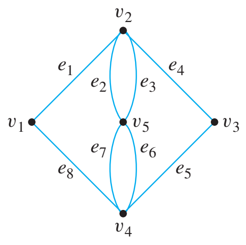
9. Each of (a)–(c) describes a graph. In each case answer *yes, no,* or *not necessarily* to this question: Does the graph have an Euler circuit? Justify your answers.
   a. G is a connected graph with five vertices of degrees 2, 2, 3, 3, and 4.
   b. G is a connected graph with five vertices of degrees 2, 2, 4, 4, and 6.
   c. G is a graph with five vertices of degrees 2, 2, 4, 4, and 6.

10. The solution for Example 10.2.5 shows a graph for which every vertex has even degree but which does not have an Euler circuit. Give another example of a graph satisfying these properties.

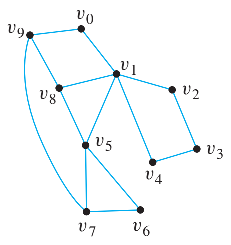
11. Is it possible for a citizen of Königsberg to make a tour of the city and cross each bridge exactly twice? (See Figure 10.2.1.) Why?

Determine which of the graphs in 12–17 have Euler circuits. If the graph does not have an Euler circuit, explain why not. If it does have an Euler circuit, describe one.

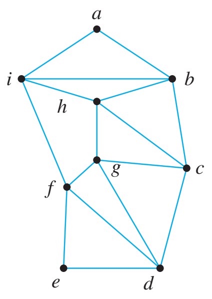
12.



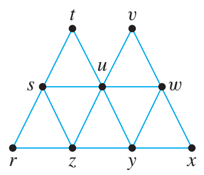
13.



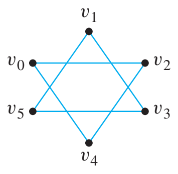
14.



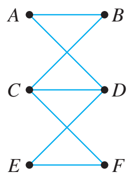
15.



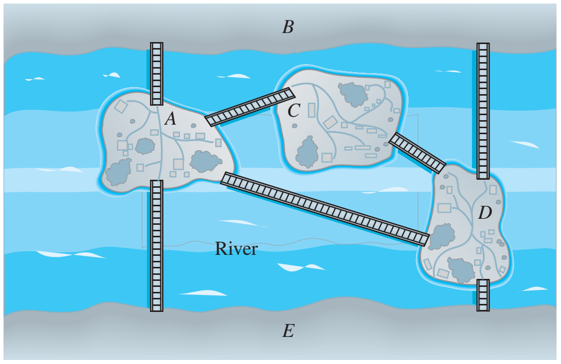
16.



17.

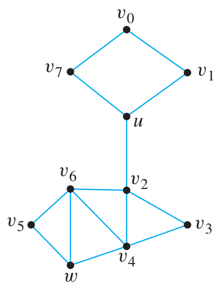


18. Is it possible to take a walk around the city whose map is shown below, starting and ending at the same point and crossing each bridge exactly once? If so, how can this be done?

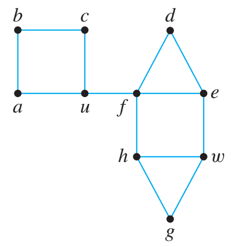


For each of the graphs in 19–21, determine whether there is an Euler path from u to w. If there is, find such a path.

19.

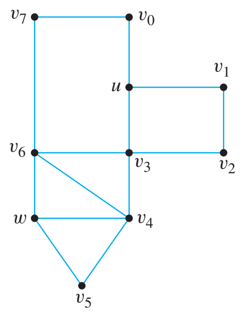


20.

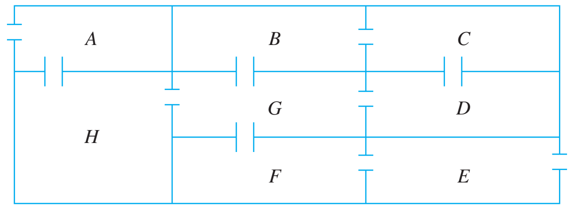


Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

21.

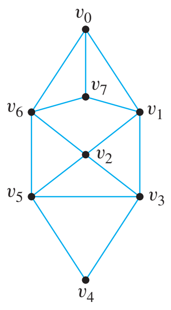


22. The following is a floor plan of a house. Is it possible to enter the house in room A, travel through every interior doorway of the house exactly once, and exit out of room E? If so, how can this be done?

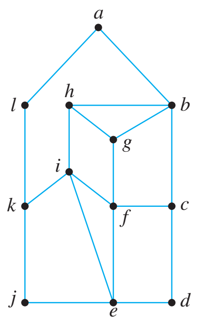


Find Hamiltonian circuits for each of the graphs in 23 and 24.

23.

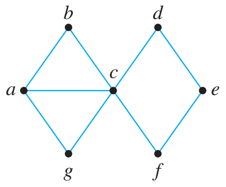


24.

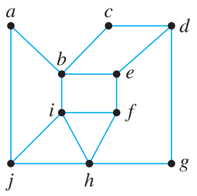


Show that none of the graphs in 25–27 has a Hamiltonian circuit.

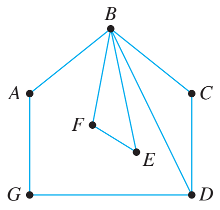
H 25.



26.

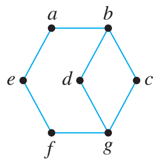


27.

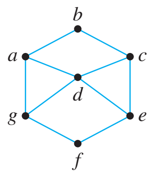


In 28–31 find Hamiltonian circuits for those graphs that have them. Explain why the other graphs do not.

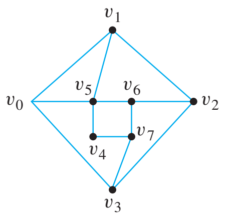
H 28.



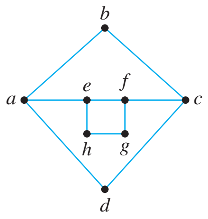
29.



30.



31.



H 32. Give two examples of graphs that have Euler circuits but not Hamiltonian circuits.

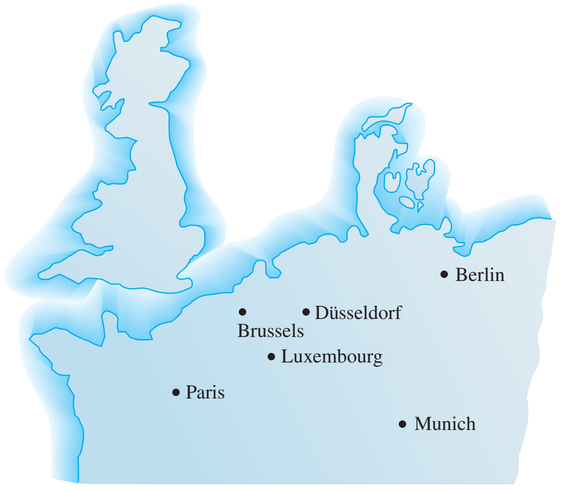
H 33. Give two examples of graphs that have Hamiltonian circuits but not Euler circuits.

H 34. Give two examples of graphs that have circuits that are both Euler circuits and Hamiltonian circuits.

H 35. Give two examples of graphs that have Euler circuits and Hamiltonian circuits that are not the same.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

36. A traveler in Europe wants to visit each of the cities shown on the map exactly once, starting and ending in Brussels. The distance (in kilometers) between each pair of cities is given in the table. Find a Hamiltonian circuit that minimizes the total distance traveled. (Use the map to narrow the possible circuits down to just a few. Then use the table to find the total distance for each of those.)



	Berlin	Brussels	Düsseldorf	Luxembourg	Munich
Brussels	783				
Düsseldorf	564	223			
Luxembourg	764	219	224		
Munich	585	771	613	517	
Paris	1,057	308	497	375	832

37. **a.** Prove that if a walk in a graph contains a repeated edge, then the walk contains a repeated vertex.
b. Explain how it follows from part (a) that any walk with no repeated vertex has no repeated edge.

38. Prove Lemma 10.2.1(a): If G is a connected graph, then any two distinct vertices of G can be connected by a path.

39. Prove Lemma 10.2.1(b): If vertices v and w are part of a circuit in a graph G and one edge is removed from the circuit, then there still exists a trail from v to w in G.

40. Draw a picture to illustrate Lemma 10.2.1(c): If a graph G is connected and G contains a circuit, then an edge of the circuit can be removed without disconnecting G.

41. Prove that if there is a trail in a graph G from a vertex v to a vertex w, then there is a trail from w to v.

H **42.** If a graph contains a circuit that starts and ends at a vertex v, does the graph contain a simple circuit that starts and ends at v? Why?

43. Prove that if there is a circuit in a graph that starts and ends at a vertex v and if w is another vertex in the circuit, then there is a circuit in the graph that starts and ends at w.

44. Let G be a connected graph, and let C be any circuit in G that does not contain every vertex of C. Let G' be the subgraph obtained by removing all the edges of C from G and also any vertices that become isolated when the edges of C are removed. Prove that there exists a vertex v such that v is in both C and G'.

45. Prove that any graph with an Euler circuit is connected.

46. Prove Corollary 10.2.5.

47. For what values of n does the complete graph K_n with n vertices have (a) an Euler circuit? (b) a Hamiltonian circuit? Justify your answers.

✶ 48. For what values of m and n does the complete bipartite graph on (m, n) vertices have (a) an Euler circuit? (b) a Hamiltonian circuit? Justify your answers.

✶ 49. What is the maximum number of edges a simple disconnected graph with n vertices can have? Prove your answer.

✶ 50. Show that a graph is bipartite if, and only if, it does not have a circuit with an odd number of edges. (See exercise 37 of Section 10.1 for the definition of bipartite graph.)

Answers for Test Yourself

1. (a) a finite alternating sequence of adjacent vertices and edges of G (b) a walk that does not contain a repeated edge (c) a trail that does not contain a repeated vertex (d) a walk that starts and ends at the same vertex (e) a closed walk that contains at least one edge and does not contain a repeated edge (f) a circuit that does not have any repeated vertex other than the first and the last (g) a walk consisting of a single vertex and no edge (h) there is a walk from v to w 2. given any two vertices in the graph, there is a walk from one to the other 3. disconnect the graph 4. a circuit that contains every vertex and every edge of the graph 5. the graph is connected, and every vertex has positive, even degree 6. the graph is connected, v and w have odd degree, and all other vertices have positive even degree 7. a simple circuit that includes every vertex of the graph 8. H contains every vertex of G; H is connected; H has the same number of edges as vertices; every vertex of H has degree 2 9. Hamiltonian circuit

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

10.3 Matrix Representations of Graphs

Order and simplification are the first steps toward the mastery of a subject.
— Thomas Mann, *The Magic Mountain*, 1924

How can graphs be represented inside a computer? It happens that all the information needed to specify a graph can be conveyed by a structure called a *matrix,* and matrices (*matrices* is the plural of *matrix*) are easy to represent inside computers. This section contains some basic definitions about matrices and matrix operations, a description of the relation between graphs and matrices, and some applications.

Matrices

Matrices are two-dimensional analogues of sequences. They are also called two-dimensional arrays.

• Definition

An $\boldsymbol{m \times n}$ (read "m by n") **matrix A over a set $\boldsymbol{S}$** is a rectangular array of elements of S arranged into m rows and n columns:

$$\mathbf{A} = \begin{array}{c} \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix} \begin{array}{l} \\ \\ \\ \leftarrow i\text{th row of } \mathbf{A} \\ \\ \\ \end{array} \\ \uparrow \\ j\text{th column of } \mathbf{A} \end{array}$$

We write $\mathbf{A} = (a_{ij})$.

The $\boldsymbol{i}$**th row of A** is

$$[a_{i1} \quad a_{i2} \quad \cdots \quad a_{in}]$$

and the $\boldsymbol{j}$**th column of A** is

$$\begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{bmatrix}.$$

The entry a_{ij} in the ith row and jth column of **A** is called the $\boldsymbol{ij}$**th entry of A.** An $m \times n$ matrix is said to have **size $\boldsymbol{m \times n}$.** If **A** and **B** are matrices, then $\mathbf{A} = \mathbf{B}$ if, and only if, **A** and **B** have the same size and the corresponding entries of **A** and **B** are all equal; that is,

$$a_{ij} = b_{ij} \quad \text{for all } i = 1, 2, \dots, m \text{ and } j = 1, 2 \dots, n.$$

A matrix for which the numbers of rows and columns are equal is called a **square matrix.** If **A** is a square matrix of size $n \times n$, then the **main diagonal of A** consists of all the entries $a_{11}, a_{22}, \dots, a_{nn}$:

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1i} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2i} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ii} & \cdots & a_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{ni} & \cdots & a_{nn} \end{bmatrix} \longleftarrow \text{main diagonal of } \mathbf{A}$$

Example 10.3.1 Matrix Terminology

The following is a 3 × 3 matrix over the set of integers.

$$\begin{bmatrix} 1 & 0 & -3 \\ 4 & -1 & 5 \\ -2 & 2 & 0 \end{bmatrix}$$

a. What is the entry in row 2, column 3?

b. What is the second column of **A**?

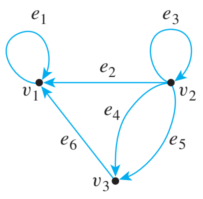
c. What are the entries in the main diagonal of **A**?

Solution

a. 5 b. $\begin{bmatrix} 0 \\ -1 \\ 2 \end{bmatrix}$ c. 1, −1, and 0 ■

Matrices and Directed Graphs

Consider the directed graph shown in Figure 10.3.1. This graph can be represented by the matrix $\mathbf{A} = (a_{ij})$ for which a_{ij} = the number of arrows from v_i to v_j, for all $i = 1, 2, 3$ and $j = 1, 2, 3$. Thus $a_{11} = 1$ because there is one arrow from v_1 to v_1, $a_{12} = 0$ because there is no arrow from v_1 to v_2, $a_{23} = 2$ because there are two arrows from v_2 to v_3, and so forth. **A** is called the *adjacency matrix* of the directed graph. For convenient reference, the rows and columns of **A** are often labeled with the vertices of the graph G.



Directed Graph G

(a)

$$\mathbf{A} = \begin{array}{c|ccc} & v_1 & v_2 & v_3 \\ \hline v_1 & 1 & 0 & 0 \\ v_2 & 1 & 1 & 2 \\ v_3 & 1 & 0 & 0 \end{array}$$

Adjacency Matrix

(b)

Figure 10.3.1 A Directed Graph and Its Adjacency Matrix

• Definition

Let G be a directed graph with ordered vertices $v_1, v_2, \ldots, v_n$. The **adjacency matrix of G** is the $n \times n$ matrix $\mathbf{A} = (a_{ij})$ over the set of nonnegative integers such that

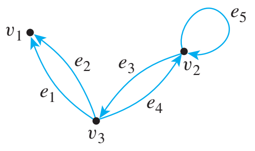
$$a_{ij} = \text{the number of arrows from } v_i \text{ to } v_j \quad \text{for all } i, j = 1, 2, \ldots, n.$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

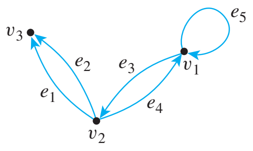
Note that nonzero entries along the main diagonal of an adjacency matrix indicate the presence of loops, and entries larger than 1 correspond to parallel edges. Moreover, if the vertices of a directed graph are reordered, then the entries in the rows and columns of the corresponding adjacency matrix are moved around.

Example 10.3.2 The Adjacency Matrix of a Graph

The two directed graphs shown below differ only in the ordering of their vertices. Find their adjacency matrices.



(a)



(b)

Solution Since both graphs have three vertices, both adjacency matrices are 3×3 matrices. For (a), all entries in the first row are 0 since there are no arrows from v_1 to any other vertex. For (b), the first two entries in the first row are 1 and the third entry is 0 since from v_1 there are single arrows to v_1 and to v_2 and no arrows to v_3. Continuing the analysis in this way, you obtain the following two adjacency matrices:

$$\begin{array}{c c} & \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 2 & 1 & 0 \end{bmatrix} \end{array} \qquad \begin{array}{c c} & \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} & \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

(a) (b) ■

If you are given a square matrix with nonnegative integer entries, you can construct a directed graph with that matrix as its adjacency matrix. However, the matrix does not tell you how to label the edges, so the directed graph is not uniquely determined.

Example 10.3.3 Obtaining a Directed Graph from a Matrix

Let

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 \end{bmatrix}.$$

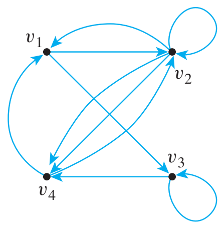
Draw a directed graph that has $\mathbf{A}$ as its adjacency matrix.

Solution Let G be the graph corresponding to $\mathbf{A}$, and let v_1, v_2, v_3, v_4 be the vertices of G. Label $\mathbf{A}$ across the top and down the left side with these vertex names, as shown below.

$$\mathbf{A} = \begin{array}{c c} & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} & \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Then, for instance, the 2 in the fourth row and the first column means that there are two arrows from v_4 to v_1. The 0 in the first row and the fourth column means that there is no arrow from v_1 to v_4. A corresponding directed graph is shown on the next page (without edge labels because the matrix does not determine those).

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



■

Matrices and Undirected Graphs

Once you know how to associate a matrix with a directed graph, the definition of the matrix corresponding to an undirected graph should seem natural to you. As before, you must order the vertices of the graph, but in this case you simply set the ijth entry of the adjacency matrix equal to the number of edges connecting the ith and jth vertices of the graph.

• Definition

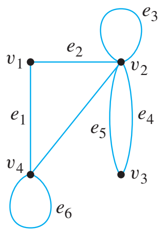
Let G be an undirected graph with ordered vertices $v_1, v_2, \ldots, v_n$. The **adjacency matrix of G** is the $n \times n$ matrix $\mathbf{A} = (a_{ij})$ over the set of nonnegative integers such that

$$a_{ij} = \text{ the number of edges connecting } v_i \text{ and } v_j$$

for all $i, j = 1, 2, \ldots, n$.

Example 10.3.4 Finding the Adjacency Matrix of a Graph

Find the adjacency matrix for the graph G shown below.



Solution

$$\mathbf{A} = \begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 1 & 0 & 1 \\ v_2 & 1 & 1 & 2 & 1 \\ v_3 & 0 & 2 & 0 & 0 \\ v_4 & 1 & 1 & 0 & 1 \end{array}$$

■

Note that if the matrix $\mathbf{A} = (a_{ij})$ in Example 10.3.4 is flipped across its main diagonal, it looks the same: $a_{ij} = a_{ji}$, for $i, j = 1, 2, \ldots, n$. Such a matrix is said to be *symmetric*.

• Definition

An $n \times n$ square matrix $\mathbf{A} = (a_{ij})$ is called **symmetric** if, and only if, for all $i, j = 1, 2, \ldots, n$,

$$a_{ij} = a_{ji}.$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.3.5 Symmetric Matrices

Which of the following matrices are symmetric?

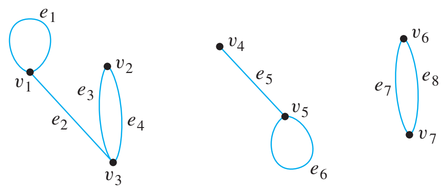
a. $\begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix}$ b. $\begin{bmatrix} 0 & 1 & 2 \\ 1 & 1 & 0 \\ 2 & 0 & 3 \end{bmatrix}$ c. $\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$

Solution Only (b) is symmetric. In (a) the entry in the first row and the second column differs from the entry in the second row and the first column; the matrix in (c) is not even square.

It is easy to see that the matrix of *any* undirected graph is symmetric since it is always the case that the number of edges joining v_i and v_j equals the number of edges joining v_j and v_i for all $i, j = 1, 2, \ldots, n$.

Matrices and Connected Components

Consider a graph G, as shown below, that consists of several connected components.



The adjacency matrix of G is

$$\mathbf{A} = \left[\begin{array}{ccc:cc:cc} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \end{array}\right].$$

As you can see, **A** consists of square matrix blocks (of different sizes) down its diagonal and blocks of 0's everywhere else. The reason is that vertices in each connected component share no edges with vertices in other connected components. For instance, since v_1, v_2, and v_3 share no edges with v_4, v_5, v_6, or v_7, all entries in the top three rows to the right of the third column are 0 and all entries in the left three columns below the third row are also 0. Sometimes matrices whose entries are all 0's are themselves denoted 0. If this convention is followed here, **A** is written as

$$\mathbf{A} = \left[\begin{array}{c|c|c} \begin{matrix} 1 & 0 & 1 \\ 0 & 0 & 2 \\ 1 & 2 & 0 \end{matrix} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \begin{matrix} 0 & 1 \\ 1 & 1 \end{matrix} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \begin{matrix} 0 & 2 \\ 2 & 0 \end{matrix} \end{array}\right]$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

The previous reasoning can be generalized to prove the following theorem:

Theorem 10.3.1

Let G be a graph with connected components $G_1, G_2, \ldots, G_k$. If there are n_i vertices in each connected component G_i and these vertices are numbered consecutively, then the adjacency matrix of G has the form

$$\begin{bmatrix} A_1 & O & O & \cdots & O & O \\ O & A_2 & O & \cdots & O & O \\ O & O & A_3 & \cdots & O & O \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ O & O & O & \cdots & O & A_k \end{bmatrix}$$

where each A_i is the $n_i \times n_i$ adjacency matrix of G_i, for all $i = 1, 2, \ldots, k$, and the O's represent matrices whose entries are all 0.

Matrix Multiplication

Matrix multiplication is an enormously useful operation that arises in many contexts, including the investigation of walks in graphs. Although matrix multiplication can be defined in quite abstract settings, the definition for matrices whose entries are real numbers will be sufficient for our applications. The product of two matrices is built up of *scalar* or *dot* products of their individual rows and columns.

• Definition

Suppose that all entries in matrices **A** and **B** are real numbers. If the number of elements, n, in the ith row of **A** equals the number of elements in the jth column of **B**, then the **scalar product** or **dot product** of the ith row of **A** and the jth column of **B** is the real number obtained as follows:

$$[a_{i1} \quad a_{i2} \quad \cdots \quad a_{in}] \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{bmatrix} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj}.$$

Example 10.3.6 Multiplying a Row and a Column

$$\begin{aligned} [3 \quad 0 \quad -1 \quad 2] \begin{bmatrix} -1 \\ 2 \\ 3 \\ 0 \end{bmatrix} &= 3\cdot(-1) + 0\cdot 2 + (-1)\cdot 3 + 2\cdot 0 \\ &= -3 + 0 - 3 + 0 = -6 \end{aligned}$$ ■

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

More generally, if **A** and **B** are matrices whose entries are real numbers and if **A** and **B** have *compatible sizes* in the sense that the number of columns of **A** equals the number of rows of **B**, then the product **AB** is defined. It is the matrix whose ijth entry is the scalar product of the ith row of **A** times the jth column of **B**, for all possible values of i and j.

• Definition

Let $\mathbf{A} = (a_{ij})$ be an $m \times k$ matrix and $\mathbf{B} = (b_{ij})$ a $k \times n$ matrix with real entries. The (matrix) product of **A** times **B**, denoted **AB**, is that matrix (c_{ij}) defined as follows:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & & \vdots \\ \boxed{a_{i1} \quad a_{i2} \quad \cdots \quad a_{ik}} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2j} & \cdots & b_{2n} \\ & \cdot & & \cdot & & \cdot \\ & \cdot & & \cdot & & \cdot \\ & \cdot & & \cdot & & \cdot \\ b_{k1} & b_{k2} & \cdots & b_{kj} & \cdots & b_{kn} \end{bmatrix}$$

$$= \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1j} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2j} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{i1} & c_{i2} & \cdots & \boxed{c_{ij}} & \cdots & c_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mj} & \cdots & c_{mn} \end{bmatrix}$$

where

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ik}b_{kj} = \sum_{r=1}^{k} a_{ir}b_{rj},$$

for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

Example 10.3.7 Computing a Matrix Product

Let $\mathbf{A} = \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix}$. Compute **AB**.

Solution **A** has size 2×3 and **B** has size 3×2, so the number of columns of **A** equals the number of rows of **B** and the matrix product of **A** and **B** can be computed. Then

$$\begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix},$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

where

$$c_{11} = 2\cdot 4 + 0\cdot 2 + 3\cdot(-2) = 2 \qquad \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix}$$

$$c_{12} = 2\cdot 3 + 0\cdot 2 + 3\cdot(-1) = 3 \qquad \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix}$$

$$c_{21} = (-1)\cdot 4 + 1\cdot 2 + 0\cdot(-2) = 2 \qquad \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix}$$

$$c_{22} = (-1)\cdot 3 + 0\cdot 2 + 3\cdot(-1) = -1 \qquad \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix}.$$

Hence

$$\mathbf{AB} = \begin{bmatrix} 2 & 3 \\ -2 & -1 \end{bmatrix}. \qquad ■$$

Matrix multiplication is both similar to and different from multiplication of real numbers. One difference is that although the product of any two numbers can be formed, only matrices with compatible sizes can be multiplied. Also, multiplication of real numbers is commutative (for all real numbers a and b, $ab = ba$), whereas matrix multiplication is not. For instance,

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 0 & 1 \end{bmatrix}, \quad \text{but} \quad \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

On the other hand, both real number and matrix multiplications are associative ($(ab)c = a(bc)$, for all elements a, b, and c for which the products are defined). This is proved in Example 10.3.8 for products of 2×2 matrices. Additional exploration of matrix multiplication is offered in the exercises.

Example 10.3.8 Associativity of Matrix Multiplication for 2 × 2 Matrices

Prove that if $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are 2×2 matrices over the set of real numbers, then $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$.

Solution Suppose $\mathbf{A} = (a_{ij})$, $\mathbf{B} = (b_{ij})$, and $\mathbf{C} = (c_{ij})$ are particular but arbitrarily chosen 2×2 matrices with real entries. Since the numbers of rows and columns are all the same, $\mathbf{AB}$, $\mathbf{BC}$, $(\mathbf{AB})\mathbf{C}$, and $\mathbf{A}(\mathbf{BC})$ are defined. Let $\mathbf{AB} = (d_{ij})$ and $\mathbf{BC} = (e_{ij})$. Then for all integers $i = 1, 2$ and $j = 1, 2$,

$$\begin{aligned}
\text{the } ij\text{th entry of } (\mathbf{AB})\mathbf{C} &= \sum_{r=1}^{2} d_{ir}c_{rj} && \text{by definition of the product of } \mathbf{AB} \text{ and } \mathbf{C} \\
&= d_{i1}c_{1j} + d_{i2}c_{2j} && \text{by definition of } \Sigma \\
&= \left(\sum_{r=1}^{2} a_{ir}b_{r1}\right)c_{1j} + \left(\sum_{r=1}^{2} a_{ir}b_{r2}\right)c_{2j} && \text{by definition of the product of } \mathbf{A} \text{ and } \mathbf{B} \\
&= (a_{i1}b_{11} + a_{i2}b_{21})c_{1_j} && \text{by definition of } \Sigma \\
&\qquad + (a_{i1}b_{12} + a_{i2}b_{22})c_{2j} \\
&= a_{i1}b_{11}c_{1j} + a_{i2}b_{21}c_{1j} + a_{i1}b_{12}c_{2j} + a_{i2}b_{22}c_{2j}.
\end{aligned}$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Similarly, the ijth entry of $\mathbf{A}(\mathbf{BC})$ is

$$\begin{aligned}(\mathbf{A}(\mathbf{BC}))_{ij} &= \sum_{r=1}^{2} a_{ir}e_{rj} \\ &= a_{i1}e_{1j} + a_{i2}e_{2j} \\ &= a_{i1}\left(\sum_{r=1}^{2} b_{1r}c_{rj}\right) + a_{i2}\left(\sum_{r=1}^{2} b_{2r}c_{rj}\right) \\ &= a_{i1}(b_{11}c_{1j} + b_{12}c_{2j}) + a_{i2}(b_{21}c_{1j} + b_{22}c_{2j}) \\ &= a_{i1}b_{11}c_{1j} + a_{i1}b_{12}c_{2j} + a_{i2}b_{21}c_{1j} + a_{i2}b_{22}c_{2j} \\ &= a_{i1}b_{11}c_{1j} + a_{i2}b_{21}c_{1j} + a_{i1}b_{12}c_{2j} + a_{i2}b_{22}c_{2j}.\end{aligned}$$

Comparing the results of the two computations shows that for all i and j,

$$\text{the } ij\text{th entry of } (\mathbf{AB})\mathbf{C} = \text{the } ij\text{th entry of } \mathbf{A}(\mathbf{BC}).$$

Since all corresponding entries are equal, $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$, as was to be shown. ■

As far as multiplicative identities are concerned, there are both similarities and differences between real numbers and matrices. You know that the number 1 acts as a multiplicative identity for products of real numbers. It turns out that there are certain matrices, called *identity matrices,* that act as multiplicative identities for certain matrix products. For instance, mentally perform the following matrix multiplications to check that for any real numbers a, b, c, d, e, f, g, h and i,

$$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\begin{bmatrix}a & b & c\\d & e & f\end{bmatrix} = \begin{bmatrix}a & b & c\\d & e & f\end{bmatrix}$$

and

$$\begin{bmatrix}a & b & c\\d & e & f\\g & h & i\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}a & b & c\\d & e & f\\g & h & i\end{bmatrix}.$$

These computations show that $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ acts as an identity on the left side for multiplication with 2×3 matrices and that $\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$ acts as an identity on the right side for multiplication with 3×3 matrices. Note that $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ cannot act as an identity on the right side for multiplication with 2×3 matrices because the sizes are not compatible.

David Eugene Smith Collection, Rare Book and Manuscript Library, Columbia University

Leopold Kronecker (1823–1891)

• Definition

For each positive integer n, the $\boldsymbol{n \times n}$ **identity matrix,** denoted $\mathbf{I}_n = (\delta_{ij})$ or just $\mathbf{I}$ (if the size of the matrix is obvious from context), is the $n \times n$ matrix in which all the entries in the main diagonal are 1's and all other entries are 0's. In other words,

$$\delta_{ij} = \begin{cases}1 & \text{if} \quad i = j\\0 & \text{if} \quad i \neq j\end{cases}, \qquad \text{for all } i, j = 1, 2, \ldots, n.$$

The German mathematician Leopold Kronecker introduced the symbol δ_{ij} to make matrix computations more convenient. In his honor, this symbol is called the *Kronecker delta.*

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.3.9 An Identity Matrix Acts as an Identity

Prove that if **A** is any $m \times n$ matrix and **I** is the $n \times n$ identity matrix, then $\mathbf{AI} = \mathbf{A}$. (In exercise 14 at the end of this section you are asked to show that if **I** is the $m \times m$ identity matrix, then $\mathbf{IA} = \mathbf{A}$.)

Proof:

Let **A** be any $n \times n$ matrix and let a_{ij} be the ijth entry of **A** for all integers $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Consider the product **AI**, where **I** is the $n \times n$ identity matrix. Observe that

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & 2_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

because

$$\begin{aligned} \text{the } ij\text{th entry of } \mathbf{AI} &= \sum_{r=1}^{n} a_{ir}\delta_{rj} && \text{by definition of 1} \\ &= a_{i1}\delta_{1j} + a_{i2}\delta_{2j} + \cdots && \text{by definition of } \Sigma \\ &\quad + a_{ij}\delta_{jj} + \cdots + a_{in}\delta_{nj} \\ &= a_{ij}\delta_{jj} && \text{since } \delta_{kj} = 0 \text{ whenever } k \neq j \text{ and } \delta_{jj} = 1 \\ &= a_{ij} \\ &= \text{the } ij\text{th entry of } \mathbf{A}. \end{aligned}$$

Thus $\mathbf{AI} = \mathbf{A}$, as was to be shown. ■

There are also similarities and differences between real numbers and matrices with respect to the computation of powers. Any number can be raised to a nonnegative integer power, but a matrix can be multiplied by itself only if it has the same number of rows as columns. As for real numbers, however, the definition of matrix powers is recursive. Just as any number to the zero power is defined to be 1, so any $n \times n$ matrix to the zero power is defined to be the $n \times n$ identity matrix. The nth power of an $n \times n$ matrix **A** is defined to be the product of **A** with its $(n-1)$st power.

• Definition

For any $n \times n$ matrix **A**, the **powers of A** are defined as follows:

$$\mathbf{A}^0 = \mathbf{I} \quad \text{where } \mathbf{I} \text{ is the } n \times n \text{ identity matrix}$$
$$\mathbf{A}^n = \mathbf{A}\mathbf{A}^{n-1} \quad \text{for all integers } n \geq 1$$

Example 10.3.10 Powers of a Matrix

Let $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 0 \end{bmatrix}$. Compute $\mathbf{A}^0$, $\mathbf{A}^1$, $\mathbf{A}^2$, and $\mathbf{A}^3$.

Solution

$$\mathbf{A}^0 = \text{the } 2 \times 2 \text{ identity matrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$\mathbf{A}^1 = \mathbf{A}\mathbf{A}^0 = \mathbf{AI} = \mathbf{A}$$

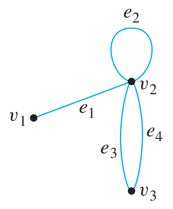
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

$$\mathbf{A}^2 = \mathbf{A}\mathbf{A}^1 = \mathbf{A}\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 2 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 2 \\ 2 & 4 \end{bmatrix}$$

$$\mathbf{A}^3 = \mathbf{A}\mathbf{A}^2 = \begin{bmatrix} 1 & 2 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} 5 & 2 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 9 & 10 \\ 10 & 4 \end{bmatrix}$$ ■

Counting Walks of Length N

A walk in a graph consists of an alternating sequence of vertices and edges. If repeated edges are counted each time they occur, then the number of edges in the sequence is called the **length** of the walk. For instance, the walk $v_2e_3v_3e_4v_2e_2v_2e_3v_3$ has length 4 (counting e_3 twice). Consider the following graph G:



How many distinct walks of length 2 connect v_2 and v_2? Your can list the possibilities systematically as follows: From v_1, the first edge of the walk must go to *some* vertex of G: v_1, v_2, or v_3. There is one walk of length 2 from v_2 to v_2 that starts by going from v_2 to v_1:

$$v_2e_1v_1e_1v_2.$$

There is one walk of length 2 from v_2 to v_2 that starts by going from v_2 to v_2:

$$v_2e_2v_2e_2v_2.$$

And there are four walks of length 2 from v_2 to v_2 that start by going from v_2 to v_3:

$$v_2e_3v_3e_4v_2,$$
$$v_2e_4v_3e_3v_2,$$
$$v_2e_3v_3e_3v_2,$$
$$v_2e_4v_3e_4v_2.$$

Thus the answer is six.

The general question of finding the number of walks that have a given length and connect two particular vertices of a graph can easily be answered using matrix multiplication. Consider the adjacency matrix $\mathbf{A}$ of the graph G on the previous page:

$$\mathbf{A} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array}\begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix} \end{array}.$$

Compute $\mathbf{A}^2$ as follows:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 6 & 2 \\ 2 & 2 & 4 \end{bmatrix}.$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Note that the entry in the second row and the second column is 6, which equals the number of walks of length 2 from v_2 to v_2. This is no accident! To compute a_{22}, you multiply the second row of $\mathbf{A}$ times the second column of $\mathbf{A}$ to obtain a sum of three terms:

$$\begin{bmatrix} 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = 1 \cdot 1 + 1 \cdot 1 + 2 \cdot 2.$$

Observe that

$$\begin{bmatrix} \text{the first term} \\ \text{of this sum} \end{bmatrix} = \begin{bmatrix} \text{number of} \\ \text{edges from} \\ v_2 \text{ to } v_1 \end{bmatrix} \cdot \begin{bmatrix} \text{number of} \\ \text{edges from} \\ v_1 \text{ to } v_2 \end{bmatrix} = \begin{bmatrix} \text{number of pairs} \\ \text{of edges from} \\ v_2 \text{ to } v_1 \text{ and } v_1 \text{ to } v_2 \end{bmatrix}.$$

Now consider the ith term of this sum, for each $i = 1, 2,$ and 3. It equals the number of edges from v_2 to v_i times the number of edges from v_i to v_2. By the multiplication rule this equals the number of pairs of edges from v_2 to v_i and from v_i back to v_2. But this equals the number of walks of length 2 that start and end at v_2 and pass through v_i. Since this analysis holds for each term of the sum for $i = 1, 2,$ and 3, the sum as a whole equals the total number of walks of length 2 that start and end at v_2:

$$1 \cdot 1 + 1 \cdot 1 + 2 \cdot 2 = 1 + 1 + 4 = 6.$$

More generally, if $\mathbf{A}$ is the adjacency matrix of a graph G, the ijth entry of $\mathbf{A}^2$ equals the number of walks of length 2 connecting the ith vertex to the jth vertex of G. Even more generally, if n is any positive integer, the ijth entry of $\mathbf{A}^n$ equals the number of walks of length n connecting the ith and the jth vertices of G.

Theorem 10.3.2

If G is a graph with vertices $v_1, v_2, \ldots, v_m$ and $\mathbf{A}$ is the adjacency matrix of G, then for each positive integer n and for all integers $i, j = 1, 2, \ldots, m$,

the ijth entry of $\mathbf{A}^n$ = the number of walks of length n from v_i to v_j.

Proof:

Suppose G is a graph with vertices $v_1, v_2, \ldots, v_m$ and $\mathbf{A}$ is the adjacency matrix of G. Let $P(n)$ be the sentence

For all integers $i, j = 1, 2, \ldots, m$, $\leftarrow P(n)$
the ijth entry of $\mathbf{A}^n$ = the number of walks of length n from v_i to v_j.

We will use mathematical induction to show that $P(n)$ is true for all integers $n \geq 1$.

Show that $P(1)$ is true:

The ijth entry of $\mathbf{A}^1$ = the ijth entry of $\mathbf{A}$ — because $\mathbf{A}^1 = \mathbf{A}$
= the number of edges connecting v_i to v_j — by definition of adjacency matrix
= the number of walks of length 1 from v_i to v_j — because a walk of length 1 contains a single edge.

Show that for all integers k with $k \geq 1$, if $P(k)$ is true then $P(k+1)$ is true:

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Let k be any integer with $k \geq 1$, and suppose that

For all integers $i, j = 1, 2, \ldots, m$,
the ijth entry of $\mathbf{A}^k =$ the number of walks of length k from v_i to v_j. $\leftarrow P(k)$ inductive hypothesis

We must show that

For all integers $i, j = 1, 2, \ldots, m$,
the ijth entry of $\mathbf{A}^{k+1} =$ the number of walks of length $k+1$ from v_i to v_j. $\leftarrow P(k+1)$

Let $\mathbf{A} = (a_{ij})$ and $\mathbf{A}^k = (b_{ij})$. Since $\mathbf{A}^{k+1} = \mathbf{A}\mathbf{A}^k$, the ijth entry of $\mathbf{A}^{k+1}$ is obtained by multiplying the ith row of $\mathbf{A}$ by the jth column of $\mathbf{A}^k$:

$$\text{the } ij\text{th entry of } \mathbf{A}^{k+1} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{im}b_{mj} \qquad 10.3.1$$

for all $i, j = 1, 2, \ldots, m$. Now consider the individual terms of this sum: a_{i1} is the number of edges from v_i to v_1; and, by inductive hypothesis, b_{1j} is the number of walks of length k from v_1 to v_j. But any edge from v_i to v_1 can be joined with any walk of length k from v_1 to v_j to create a walk of length $k+1$ from v_i to v_j with v_1 as its second vertex. Thus, by the multiplication rule,

$$a_{i1}b_{1j} = \begin{bmatrix}\text{the number of walks of length } k+1 \text{ from} \\ v_i \text{ to } v_j \text{ that have } v_1 \text{ as their second vertex}\end{bmatrix}.$$

More generally, for each integer $r = 1, 2, \ldots, m$,

$$a_{ir}b_{rj} = \begin{bmatrix}\text{the number of walks of length } k+1 \text{ from} \\ v_i \text{ to } v_j \text{ that have } v_r \text{ as their second vertex}\end{bmatrix}.$$

Since any walk of length $k+1$ from v_i to v_j must have *one* of the vertices $v_1, v_2, \ldots, v_m$ as its second vertex, the total number of walks of length $k+1$ from v_i to v_j equals the sum in (10.3.1), which equals the ijth entry of $\mathbf{A}^{k+1}$. Hence

$$\text{the } ij\text{th entry of } \mathbf{A}^{k+1} = \text{the number of walks of length } k+1 \text{ from } v_i \text{ to } v_j$$

[as was to be shown].
[Since both the basis step and the inductive step have been proved, the sentence $P(n)$ is true for all integers $n \geq 1$.]

Test Yourself

1. In the adjacency matrix for a directed graph, the entry in the ith row and jth column is _____.

2. In the adjacency matrix for an undirected graph, the entry in the ith row and jth column is _____.

3. An $n \times n$ square matrix is called symmetric if, and only if, for all integers i and j from 1 to n, the entry in row _____ and column _____ equals the entry in row _____ and column _____.

4. The ijth entry in the product of two matrices **A** and **B** is obtained by multiplying row _____ of **A** by row _____ of **B**.

5. In an $n \times n$ identity matrix the entries on the main diagonal are all _____ and the off-diagonal entries are all _____.

6. If G is a graph with vertices $v_1, v_2, \ldots, v_m$ and **A** is the adjacency matrix of G, then for each positive integer n and for all integers i and j with $i, j = 1, 2, \ldots, m$, the ijth entry of $\mathbf{A}^n =$ _____.

Exercise Set 10.3

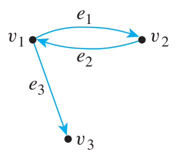
1. Find real numbers a, b, and c such that the following are true.

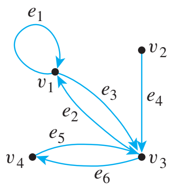
   a. $\begin{bmatrix} a+b & a-c \\ c & b-a \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 3 \end{bmatrix}$

   b. $\begin{bmatrix} 2a & b+c \\ c-a & 2b-a \end{bmatrix} = \begin{bmatrix} 4 & 3 \\ 1 & -2 \end{bmatrix}$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

2. Find the adjacency matrices for the following directed graphs.

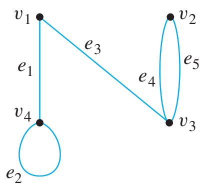
a. 

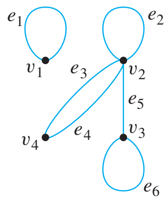
b. 

3. Find directed graphs that have the following adjacency matrices:

a. $\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 2 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$ b. $\begin{bmatrix} 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

4. Find adjacency matrices for the following (undirected) graphs.

a. 

b. 

c. K_4, the complete graph on four vertices

d. $K_{2,3}$, the complete bipartite graph on (2, 3) vertices

5. Find graphs that have the following adjacency matrices.

a. $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \end{bmatrix}$ b. $\begin{bmatrix} 0 & 2 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

6. The following are adjacency matrices for graphs. In each case determine whether the graph is connected by analyzing the matrix without drawing the graph.

a. $\begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$ b. $\begin{bmatrix} 0 & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$

7. Suppose that for all positive integers i, all the entries in the ith row and ith column of the adjacency matrix of a graph are 0. What can you conclude about the graph?

8. Find each of the following products.

a. $\begin{bmatrix} 2 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix}$ b. $\begin{bmatrix} 4 & -1 & 7 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$

9. Find each of the following products.

a. $\begin{bmatrix} 3 & 0 \\ 1 & -2 \end{bmatrix} \begin{bmatrix} 1 & -1 & 4 \\ 0 & 2 & 1 \end{bmatrix}$

b. $\begin{bmatrix} 2 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 5 & -4 \\ -2 & 2 \end{bmatrix}$

c. $\begin{bmatrix} -1 \\ 2 \end{bmatrix} \begin{bmatrix} 2 & 3 \end{bmatrix}$

10. Let $\mathbf{A} = \begin{bmatrix} 1 & 1 & -1 \\ 0 & -2 & 1 \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} -2 & 0 \\ 1 & 3 \end{bmatrix}$, and

$\mathbf{C} = \begin{bmatrix} 0 & -2 \\ 3 & 1 \\ 1 & 0 \end{bmatrix}$.

For each of the following, determine whether the indicated product exists, and compute it if it does.

a. **AB** b. **BA** c. $\mathbf{A}^2$ d. **BC** e. **CB**
f. $\mathbf{B}^2$ g. $\mathbf{B}^3$ h. $\mathbf{C}^2$ i. **AC** j. **CA**

11. Give an example different from that in the text to show that matrix multiplication is not commutative. That is, find 2×2 matrices **A** and **B** such that **AB** and **BA** both exist but $\mathbf{AB} \neq \mathbf{BA}$.

12. Let O denote the matrix $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. Find 2×2 matrices **A** and **B** such that $\mathbf{A} \neq \text{O}$ and $\mathbf{B} \neq \text{O}$, but $\mathbf{AB} = \text{O}$.

13. Let O denote the matrix $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. Find 2×2 matrices **A** and **B** such that $\mathbf{A} \neq \mathbf{B}$, $\mathbf{B} \neq \text{O}$, and $\mathbf{AB} \neq \text{O}$, but $\mathbf{BA} = \text{O}$.

In 14–18 assume the entries of all matrices are real numbers.

H 14. Prove that if **I** is the $m \times m$ identity matrix and **A** is any $m \times n$ matrix, then $\mathbf{IA} = \mathbf{A}$.

15. Prove that if **A** is an $m \times m$ symmetric matrix, then $\mathbf{A}^2$ is symmetric.

16. Prove that matrix multiplication is associative: If **A**, **B**, and **C** are any $m \times k$, $k \times r$, and $r \times n$ matrices, respectively, then $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$.

17. Use mathematical induction and the result of exercise 16 to prove that if **A** is any $m \times m$ matrix, then $\mathbf{A}^n\mathbf{A} = \mathbf{A}\mathbf{A}^n$ for all integers $n \geq 1$.

18. Use mathematical induction to prove that if **A** is an $m \times m$ symmetric matrix, then for any integer $n \geq 1$, $\mathbf{A}^n$ is also symmetric.

19. a. Let $\mathbf{A} = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$. Find $\mathbf{A}^2$ and $\mathbf{A}^3$.

b. Let G be the graph with vertices v_1, v_2, and v_3 and with **A** as its adjacency matrix. Use the answers to part (a) to find the number of walks of length 2 from v_1 to v_3 and the number of walks of length 3 from v_1 to v_3. Do not draw G to solve this problem.

c. Examine the calculations you performed in answering part (a) to find five walks of length 2 from v_3 to v_3. Then draw G and find the walks by visual inspection.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

20. The following is an adjacency matrix for a graph:

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Answer the following questions by examining the matrix and its powers only, not by drawing the graph:

a. How many walks of length 2 are there from v_2 to v_3?
b. How many walks of length 2 are there from v_3 to v_4?
c. How many walks of length 3 are there from v_1 to v_4?
d. How many walks of length 3 are there from v_2 to v_3?

21. Let **A** be the adjacent matrix for K_3, the complete graph on three vertices. Use mathematical induction to prove that for each positive integer n, all the entries along the main diagonal of $\mathbf{A}^n$ are equal to each other and all the entries that do not lie along the main diagonal are equal to each other.

22. a. Draw a graph that has

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 0 \\ 2 & 1 & 1 & 0 & 0 \end{bmatrix}$$

as its adjacency matrix. Is this graph bipartite? (For a definition of bipartite, see exercise 37 in Section 10.1.)

Definition: Given an $m \times n$ matrix **A** whose ijth entry is denoted a_{ij}, the **transpose of A** is the matrix $\mathbf{A}^t$ whose ijth entry is a_{ji}, for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

Note that the first row of **A** becomes the first column of $\mathbf{A}^t$, the second row of **A** becomes the second column of $\mathbf{A}^t$, and so forth. For instance,

$$\text{if } \mathbf{A} = \begin{bmatrix} 0 & 2 & 1 \\ 1 & 2 & 3 \end{bmatrix}, \text{ then } \mathbf{A}^t = \begin{bmatrix} 0 & 1 \\ 2 & 2 \\ 1 & 3 \end{bmatrix}.$$

H b. Show that a graph with n vertices is bipartite if, and only if, for some labeling of its vertices, its adjacency matrix has the form

$$\begin{bmatrix} \text{O} & \mathbf{A} \\ \mathbf{A}^t & \text{O} \end{bmatrix}$$

where **A** is a $k \times (n-k)$ matrix for some integer k such that $0 < k < n$, the top left O represents a $k \times k$ matrix all of whose entries are 0, $\mathbf{A}^t$ is the transpose of **A**, and the bottom right O represents an $(n-k) \times (n-k)$ matrix all of whose entries are 0.

23. a. Let G be a graph with n vertices, and let v and w be distinct vertices of G. Prove that if there is a walk from v to w, then there is a walk from v to w that has length less than or equal to $n - 1$.

H b. If $\mathbf{A} = (a_{ij})$ and $\mathbf{B} = (b_{ij})$ are any $m \times n$ matrices, the matrix $\mathbf{A} + \mathbf{B}$ is the $m \times n$ matrix whose ijth entry is $a_{ij} + b_{ij}$ for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Let G be a graph with n vertices where $n > 1$, and let **A** be the adjacency matrix of G. Prove that G is connected if, and only if, every entry of $\mathbf{A} + \mathbf{A}^2 + \cdots + \mathbf{A}^{n-1}$ is positive.

Answers for Test Yourself

1. the number of arrows from v_i (the ith vertex) to v_j (the jth vertex) 2. the number of edges connecting v_i (the ith vertex) and v_j (the jth vertex) 3. i; j; j; i 4. i; j 5. 1; 0 6. the number of walks of length n from v_i to v_j

10.4 Isomorphisms of Graphs

Thinking is a momentary dismissal of irrelevancies. — R. Buckminster Fuller, 1969

Recall from Example 10.1.3 that the two drawings shown in Figure 10.4.1 both represent the same graph: Their vertex and edge sets are identical, and their edge-endpoint functions are the same. Call this graph G.

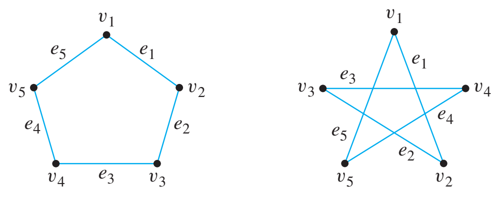


Figure 10.4.1

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Now consider the graph G' represented in Figure 10.4.2.

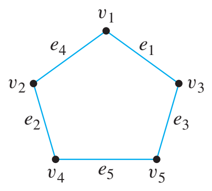
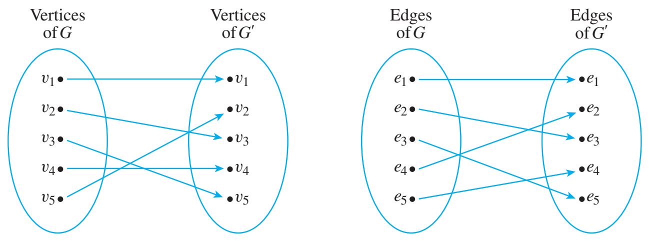


Figure 10.4.2

Observe that G' is a different graph from G (for instance, in G the endpoints of e_1 are v_1 and v_2, whereas in G' the endpoints of e_1 are v_1 and v_3). Yet G' is certainly very similar to G. In fact, if the vertices and edges of G' are relabeled by the functions shown in Figure 10.4.3, then G' becomes the same as G.



Vertices of G
Vertices of G'
Edges of G
Edges of G'
v_1 v_2 v_3 v_4 v_5
v_1 v_2 v_3 v_4 v_5
e_1 e_2 e_3 e_4 e_5
e_1 e_2 e_3 e_4 e_5

Figure 10.4.3

Note that these relabeling functions are one-to-one and onto.

Two graphs that are the same except for the labeling of their vertices and edges are called *isomorphic*. The word *isomorphism* comes from the Greek, meaning "same form." Isomorphic graphs are those that have essentially the same form.

• Definition

Let G and G' be graphs with vertex sets $V(G)$ and $V(G')$ and edge sets $E(G)$ and $E(G')$, respectively. **G is isomorphic to G'** if, and only if, there exist one-to-one correspondences $g\colon V(G) \to V(G')$ and $h\colon E(G) \to E(G')$ that preserve the edge-endpoint functions of G and G' in the sense that for all $v \in V(G)$ and $e \in E(G)$,

$$v \text{ is an endpoint of } e \quad \Leftrightarrow \quad g(v) \text{ is an endpoint of } h(e). \qquad 10.4.1$$

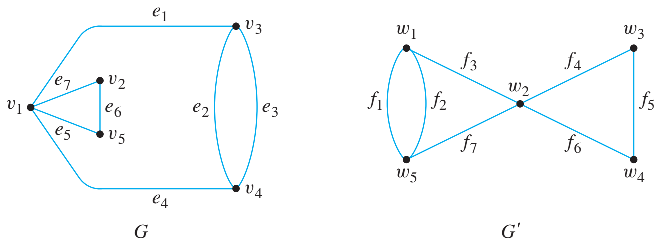
In words, G is isomorphic to G' if, and only if, the vertices and edges of G and G' can be matched up by one-to-one, onto functions such that the edges between corresponding vertices correspond to each other.

It is common in mathematics to identify objects that are isomorphic. For instance, if we are given a graph G with five vertices such that each pair of vertices is connected by an edge, then we may identify G with K_5, saying that G is K_5 rather than that G is isomorphic to K_5.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.4.1 Showing That Two Graphs Are Isomorphic

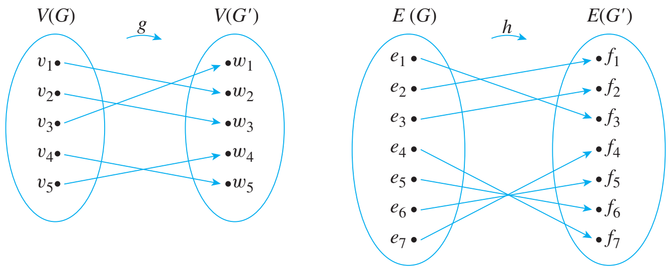
Show that the following two graphs are isomorphic.



Solution To solve this problem, you must find functions $g: V(G) \to V(G')$ and $h: E(G) \to E(G')$ such that for all $v \in V(G)$ and $e \in E(G)$, v is an endpoint of e if, and only if, $g(v)$ is an endpoint of $h(e)$. Setting up such functions is partly a matter of trial and error and partly a matter of deduction. For instance, since e_2 and e_3 are parallel (have the same endpoints), $h(e_2)$ and $h(e_3)$ must be parallel also. So $h(e_2) = f_1$ and $h(e_3) = f_2$ or $h(e_2) = f_2$ and $h(e_3) = f_1$. Also, the endpoints of e_2 and e_3 must correspond to the endpoints of f_1 and f_2, and so $g(v_3) = w_1$ and $g(v_4) = w_5$ or $g(v_3) = w_5$ and $g(v_4) = w_1$.

Similarly, since v_1 is the endpoint of four distinct edges (e_1, e_7, e_5, and e_4), $g(v_1)$ must also be the endpoint of four distinct edges (because every edge incident on $g(v_1)$ is the image under h of an edge incident on v_1 and h is one-to-one and onto). But the only vertex in G' that has four edges coming out of it is w_2, and so $g(v_1) = w_2$. Now if $g(v_3) = w_1$, then since v_1 and v_3 are endpoints of e_1 in G, $g(v_1) = w_2$ and $g(v_3) = w_1$ must be endpoints of $h(e_1)$ in G'. This implies that $h(e_1) = f_3$.

By continuing in this way, possibly making some arbitrary choices as you go, you eventually can find functions g and h to define the isomorphism between G and G'. One pair of functions (there are several) is the following:



■

It is not hard to show that graph isomorphism is an equivalence relation on a set of graphs; in other words, it is reflexive, symmetric, and transitive.

Theorem 10.4.1 Graph Isomorphism is an Equivalence Relation

Let S be a set of graphs and let R be the the relation of graph isomorphism on S. Then R is an equivalence relation on S.

Proof:

R is reflexive: Given any graph G in S, define a graph isomorphism from G to G by using the identity functions on the set of vertices and on the set of edges of G.

R is symmetric: Given any graphs G and G' in S such that G is isomorphic to G', we must show that G' is isomorphic to G.

continued on page 678

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

But this is true because if g and h are vertex and edge correspondences from G to G' that preserve the edge-endpoint functions, then g^{-1} and h^{-1} are vertex and edge correspondences from G' to G that preserve the edge-endpoint functions.

R is transitive: Given any graphs G, G', and G'' in S such that G is isomorphic to G' and G' is isomorphic to G'', we must show that G is isomorphic to G''.

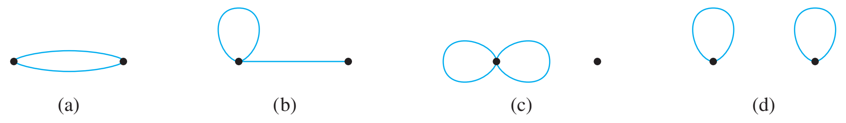
Note As a consequence of the symmetry property, you can simply say "G and G' are isomorphic" instead of "G is isomorphic to G'" or "G' is isomorphic to G."

But this follows from the fact that if g_1 and h_1 are vertex and edge correspondences from G to G' that preserve the edge-endpoint functions of G and G' and g_2 and h_2 are vertex and edge correspondences from G' to G'' that preserve the edge-endpoint functions of G' and G'', then $g_2 \circ g_1$ and $h_2 \circ h_1$ are vertex and edge correspondences from G to G'' that preserve the edge-endpoint functions of G and G''.

Example 10.4.2 Finding Representatives of Isomorphism Classes

Find all nonisomorphic graphs that have two vertices and two edges. In other words, find a collection of representative graphs with two vertices and two edges such that every such graph is isomorphic to one in the collection.

Solution There are four nonisomorphic graphs that have two vertices and two edges. These can be drawn without vertex and edge labels because any two labelings give isomorphic graphs.



To see that these four drawings show all the nonisomorphic graphs that have two vertices and two edges, first note whether one of the edges joins the two vertices or not. If it does, there are two possibilities: The other edge can also join the two vertices (as in (a)) or it can be a loop incident on one of them (as in (b)—it makes no difference *which* vertex is chosen to have the loop because interchanging the two vertex labels gives isomorphic graphs). If neither edge joins the two vertices, then both edges are loops. In this case, there are only two possibilities: Either both loops are incident on the same vertex (as in (c)) or the two loops are incident on separate vertices (as in (d)). There are no other possibilities for placing the edges, so the listing is complete. ■

Now consider the question, "Is there a general method to figure out whether graphs G and G' are isomorphic?" In other words, is there some algorithm that will accept graphs G and G' as input and produce a statement as to whether they are isomorphic? In fact, there is such an algorithm. It consists of generating all one-to-one, onto functions from the set of vertices of G to the set of vertices of G' and from the set of edges of G to the set of edges of G' and checking each pair to determine whether it preserves the edge-endpoint functions of G and G'. The problem with this algorithm is that it takes an unreasonably long time to perform, even on a high-speed computer. If G and G' each have n vertices and m edges, the number of one-to-one correspondences from vertices to vertices is $n!$ and the number of one-to-one correspondences from edges to edges is $m!$, so the total number of pairs of functions to check is $n! \cdot m!$. For instance, if $m = n = 20$, there would be $20! \cdot 20! \cong 5.9 \times 10^{36}$ pairs to check. Assuming that each check takes just 1 nanosecond, the total time would be approximately 1.9×10^{20} years!

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Unfortunately, there is no more efficient general method known for checking whether two graphs are isomorphic. However, there are some simple tests that can be used to show that certain pairs of graphs are *not* isomorphic. For instance, if two graphs are isomorphic, then they have the same number of vertices (because there is a one-to-one correspondence from the vertex set of one graph to the vertex set of the other). It follows that if you are given two graphs, one with 16 vertices and the other with 17, you can immediately conclude that the two are not isomorphic. More generally, a property that is preserved by graph isomorphism is called an *isomorphic invariant*. For instance, "having 16 vertices" is an isomorphic invariant: If one graph has 16 vertices, then so does any graph that is isomorphic to it.

• Definition

A property P is called an **invariant for graph isomorphism** if, and only if, given any graphs G and G', if G has property P and G' is isomorphic to G, then G' has property P.

Theorem 10.4.2

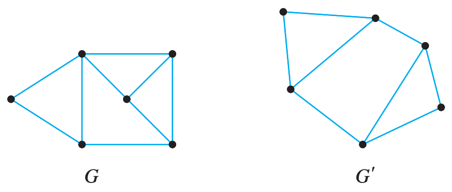
Each of the following properties is an invariant for graph isomorphism, where n, m, and k are all nonnegative integers:

1. has n vertices;
2. has m edges;
3. has a vertex of degree k;
4. has m vertices of degree k;
5. has a circuit of length k;
6. has a simple circuit of length k;
7. has m simple circuits of length k;
8. is connected;
9. has an Euler circuit;
10. has a Hamiltonian circuit.

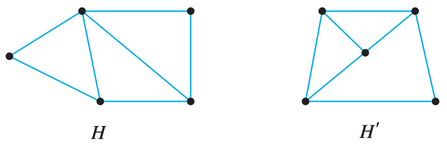
Example 10.4.3 Showing That Two Graph Are Not Isomorphic

Show that the following pairs of graphs are not isomorphic by finding an isomorphic invariant that they do not share.

a.



b.



Solution

a. G has nine edges; G' has only eight.

b. H has a vertex of degree 4; H' does not. ■

We prove part (3) of Theorem 10.4.2 on the next page and leave the proofs of the other parts as exercises.

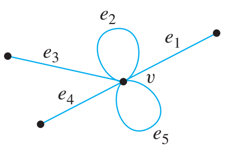
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.4.4 Proof of Theorem 10.4.2, Part (3)

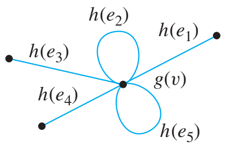
Prove that if G is a graph that has a vertex of degree k and G' is isomorphic to G, then G' has a vertex of degree k.

Proof:

Suppose G and G' are isomorphic graphs and G has a vertex v of degree k, where k is a nonnegative integer. *[We must show that G' has a vertex of degree k.]* Since G and G' are isomorphic, there are one-to-one, onto functions g and h from the vertices of G to the vertices of G' and from the edges of G to the edges of G' that preserve the edge-endpoint functions in the sense that for all edges e and all vertices u of G, u is an endpoint of e if, and only if, $g(u)$ is an endpoint of $h(e)$. An example for a particular vertex v is shown below.



Degree $v = 3 + 2 \cdot 2 = 7$



Degree $g(v) = 3 + 2 \cdot 2 = 7$

Let $e_1, e_2, \ldots, e_m$ be the m distinct edges that are incident on a vertex v in G, where m is a nonnegative integer. Then $h(e_1), h(e_2), \ldots, h(e_m)$ are m distinct edges that are incident on $g(v)$ in G'. *[The reason why $h(e_1), h(e_2), \ldots, h(e_m)$ are distinct is that h is one-to-one and $e_1, e_2, \ldots, e_m$ are distinct. And the reason why $h(e_1), h(e_2), \ldots, h(e_m)$ are incident on $g(v)$ is that g and h preserve the edge-endpoint functions of G and G' and $e_1, e_2, \ldots, e_m$ are incident on v.]*

Also, there are no edges incident on $g(v)$ other than the ones that are images under g of edges incident on v *[because g is onto and g and h preserve the edge-endpoint functions of G and G']*. Thus the number of edges incident on v equals the number of edges incident on $g(v)$.

Finally, an edge e is a loop at v if, and only if, $h(e)$ is a loop at $g(v)$, so the number of loops incident on v equals the number of loops incident on $g(v)$. *[For since g and h preserve the edge-endpoint functions of G and G', a vertex w is an endpoint of e in G if, and only if, $g(w)$ is an endpoint of $h(e)$ in G'. It follows that v is the only endpoint of e in G if, and only if, $g(v)$ is the only endpoint of $h(e)$ in G'.]*

Now the degree of v, which is k, equals the number of edges incident on v plus the number of edges incident on v that are loops (since each loop contributes 2 to the degree of v). But we have already shown that the number of edges incident on v equals the number of edges incident on $g(v)$ and that the number of loops incident on v equals the number of loops incident on $g(v)$. Hence $g(v)$ also has degree k. ■

Graph Isomorphism for Simple Graphs

When graphs G and G' are both simple, the definition of G being isomorphic to G' can be written without referring to the correspondence between the edges of G and the edges of G'.

• Definition

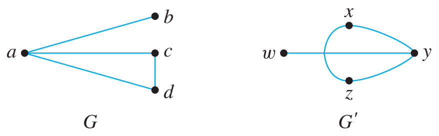
If G and G' are simple graphs, then **G is isomorphic to G'** if, and only if, there exists a one-to-one correspondence g from the vertex set $V(G)$ of G to the vertex set $V(G')$ of G' that preserves the edge-endpoint functions of G and G' in the sense that for all vertices u and v of G,

$$\{u, v\} \text{ is an edge in } G \quad \Leftrightarrow \quad \{g(u), g(v)\} \text{ is an edge in } G'. \qquad 10.4.2$$

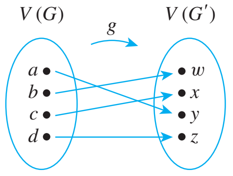
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.4.5 Isomorphism of Simple Graphs

Are the two graphs shown below isomorphic? If so, define an isomorphism.



Solution Yes. Define $f\colon V(G) \to V(G')$ by the arrow diagram shown below.



Then g is one-to-one and onto by inspection. The fact that g preserves the edge-endpoint functions of G and G' is shown by the following table:

Edges of G	**Edges of G'**
$\{a, b\}$	$\{y, w\} = \{g(a), g(b)\}$
$\{a, c\}$	$\{y, x\} = \{g(a), g(c)\}$
$\{a, d\}$	$\{y, z\} = \{g(a), g(d)\}$
$\{c, d\}$	$\{x, z\} = \{g(c), g(d)\}$

■

Test Yourself

1. If G and G' are graphs, then G is isomorphic to G' if, and only if, there exist a one-to-one correspondence g from the vertex set of G to the vertex set of G' and a one-to-one correspondence h from the edge set of G to the edge set of G' such that for all vertices v and edges e in G, v is an endpoint of e if, and only if, _____.

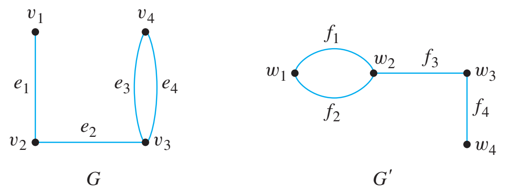
2. A property P is an invariant for graph isomorphism if, and only if, given any graphs G and G', if G has property P and G' is isomorphic to G then _____.

3. Some invariants for graph isomorphisms are _____, _____, _____, _____, _____, _____, _____, _____, _____, and _____.

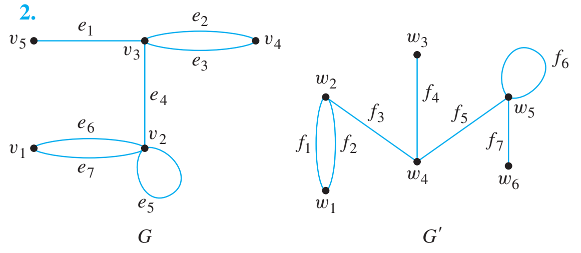
Exercise Set 10.4

For each pair of graphs G and G' in 1–5, determine whether G and G' are isomorphic. If they are, give functions $g\colon V(G) \to V(G')$ and $h\colon E(G) \to E(G')$ that define the isomorphism. If they are not, give an invariant for graph isomorphism that they do not share.

1.

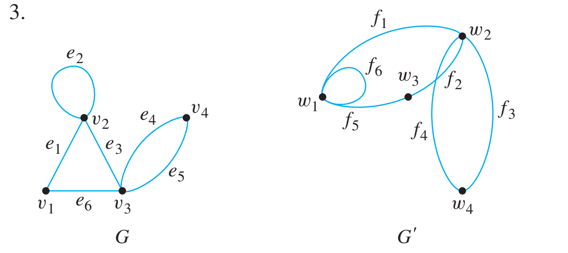


2.

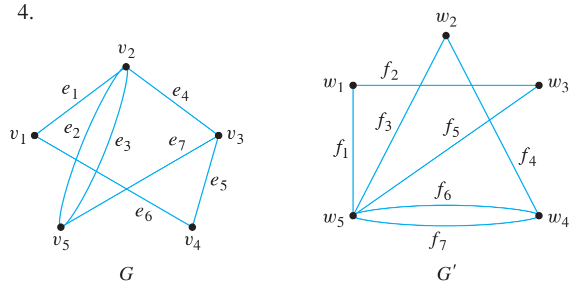


Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

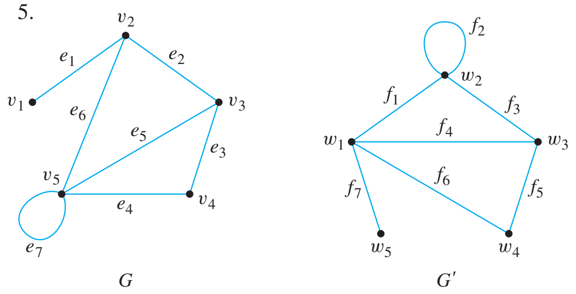
3.



4.

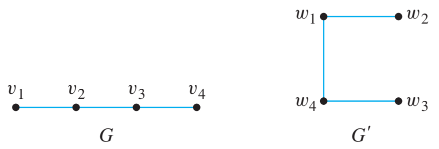


5.

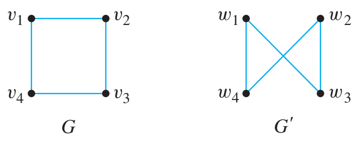


For each pair of simple graphs G and G' in 6–13, determine whether G and G' are isomorphic. If they are, give a function $g\colon V(G) \to V(G')$ that defines the isomorphism. If they are not, give an invariant for graph isomorphism that they do not share.

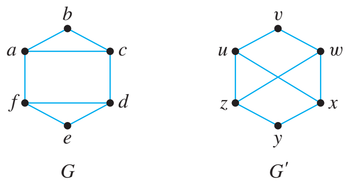
6.



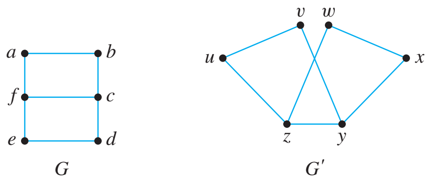
7.



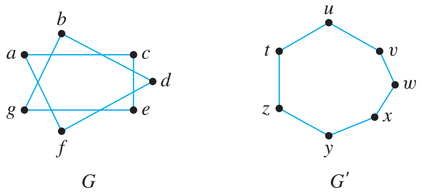
8.



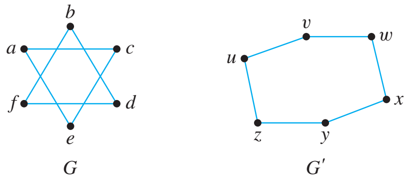
9.



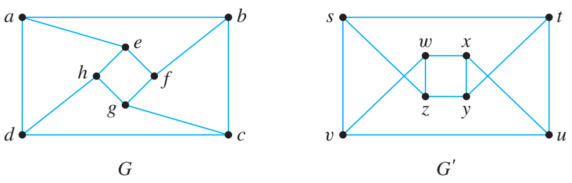
10.



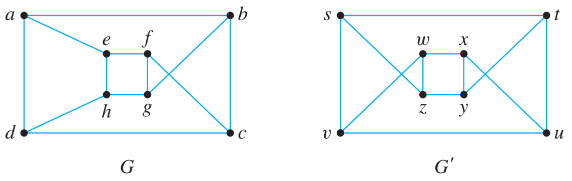
11.



12.



13.



14. Draw all nonisomorphic simple graphs with three vertices.

15. Draw all nonisomorphic simple graphs with four vertices.

16. Draw all nonisomorphic graphs with three vertices and no more than two edges.

17. Draw all nonisomorphic graphs with four vertices and no more than two edges.

H **18.** Draw all nonisomorphic graphs with four vertices and three edges.

19. Draw all nonisomorphic graphs with six vertices, all having degree 2.

20. Draw four nonisomorphic graphs with six vertices, two of degree 4 and four of degree 3.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Prove that each of the properties in 21–29 is an invariant for graph isomorphism. Assume that n, m, and k are all nonnegative integers.

21. Has n vertices

22. Has m edges

23. Has a circuit of length k

24. Has a simple circuit of length k

H **25.** Has m vertices of degree k

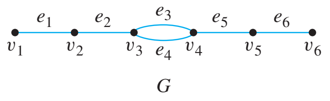
26. Has m simple circuits of length k

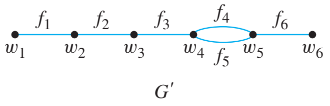
H **27.** Is connected

28. Has an Euler circuit

29. Has a Hamiltonian circuit

30. Show that the following two graphs are not isomorphic by supposing they are isomorphic and deriving a contradiction.





Answers for Test Yourself

1. $g(v)$ is an endpoint of $h(e)$ 2. G' has property P 3. has n vertices; has m edges; has a vertex of degree k; has m vertices of degree k; has a circuit of length k; has a simple circuit of length k; has m simple circuits of length k; is connected; has an Euler circuit; has a Hamiltonian circuit

10.5 Trees

We are not very pleased when we are forced to accept a mathematical truth by virtue of a complicated chain of formal conclusions and computations, which we traverse blindly, link by link, feeling our way by touch. We want first an overview of the aim and of the road; we want to understand the idea of the proof, the deeper context.
— Hermann Weyl, 1885–1955

If a friend asks what you are studying and you answer “trees,” your friend is likely to infer you are taking a course in botany. But trees are also a subject for mathematical investigation. In mathematics, a tree is a connected graph that does not contain any circuits. Mathematical trees are similar in certain ways to their botanical namesakes.

• Definition

A graph is said to be **circuit-free** if, and only if, it has no circuits. A graph is called a **tree** if, and only if, it is circuit-free and connected. A **trivial tree** is a graph that consists of a single vertex. A graph is called a **forest** if, and only if, it is circuit-free and not connected.

Example 10.5.1 Trees and Non-Trees

All the graphs shown in Figure 10.5.1 are trees, whereas those in Figure 10.5.2 are not.

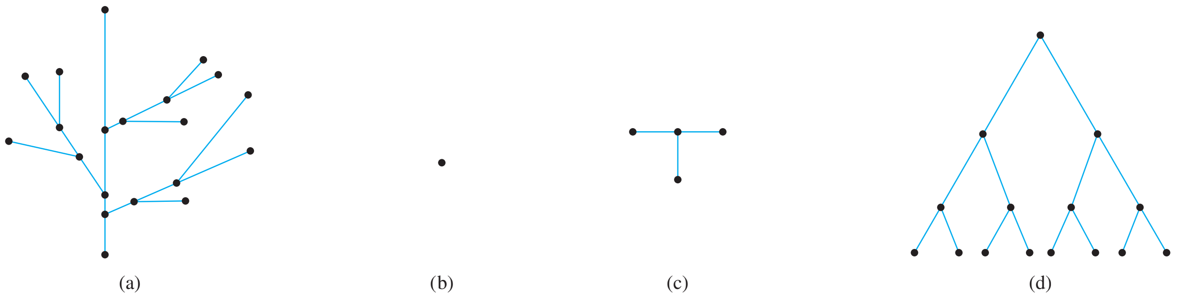


Figure 10.5.1 **Trees. All the graphs in (a)–(d) are connected and circuit-free.**

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

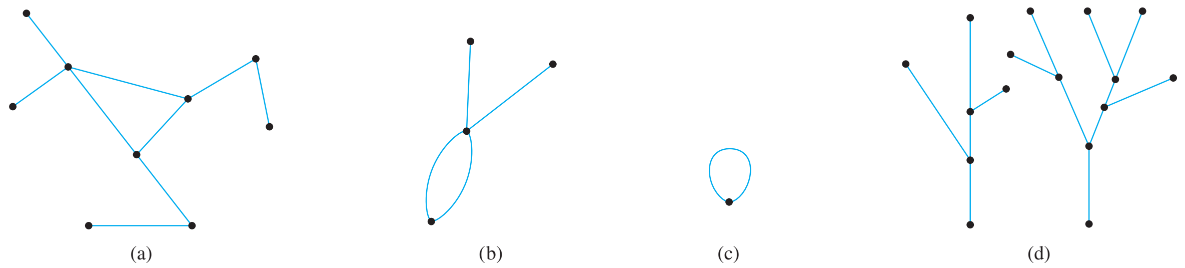


Figure 10.5.2 Non-Trees. The graphs in (a), (b), and (c) all have circuits, and the graph in (d) is not connected. ■

Examples of Trees

The following examples illustrate just a few of the many and varied situations in which mathematical trees arise.

Example 10.5.2 A Decision Tree

During orientation week, a college administers an exam to all entering students to determine placement in the mathematics curriculum. The exam consists of two parts, and placement recommendations are made as indicated by the tree shown in Figure 10.5.3. Read the tree from left to right to decide what course should be recommended for a student who scored 9 on part I and 7 on part II.

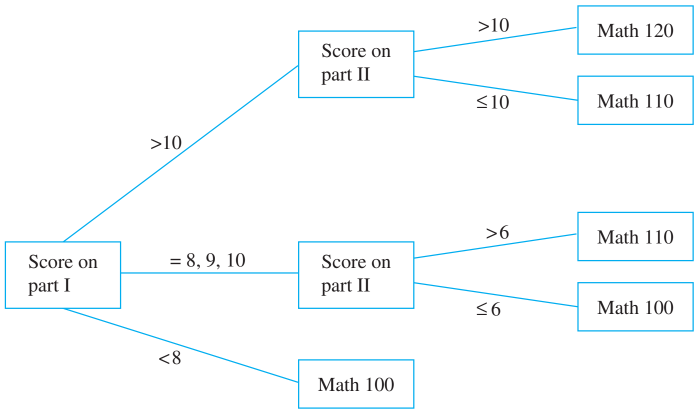


Figure 10.5.3

Solution Since the student scored 9 on part I, the score on part II is checked. Since it is greater than 6, the student should be advised to take Math 110. ■

Example 10.5.3 A Parse Tree

In the last 30 years, Noam Chomsky and others have developed new ways to describe the syntax (or grammatical structure) of natural languages such as English. As is discussed briefly in Chapter 12, this work has proved useful in constructing compilers for high-level computer languages. In the study of grammars, trees are often used to show the derivation of grammatically correct sentences from certain basic rules. Such trees are called **syntactic derivation trees** or **parse trees.**

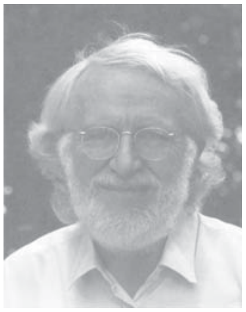
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

A very small subset of English grammar, for example, specifies that

1. a sentence can be produced by writing first a noun phrase and then a verb phrase;
2. a noun phrase can be produced by writing an article and then a noun;
3. a noun phrase can also be produced by writing an article, then an adjective, and then a noun;
4. a verb phrase can be produced by writing a verb and then a noun phrase;
5. one article is "the";
6. one adjective is "young";
7. one verb is "caught";
8. one noun is "man";
9. one (other) noun is "ball."

Courtesy of IBM Corporation

John Backus
(1924–1998)


Courtesy of Peter Naur

Peter Naur
(born 1928)

The rules of a grammar are called **productions.** It is customary to express them using the shorthand notation illustrated below. This notation, introduced by John Backus in 1959 and modified by Peter Naur in 1960, was used to describe the computer language Algol and is called the **Backus-Naur notation.** In the notation, the symbol | represents the word *or,* and angle brackets $\langle\ \rangle$ are used to enclose terms to be defined (such as a sentence or noun phrase).

1. $\langle\text{sentence}\rangle \to \langle\text{noun phrase}\rangle\langle\text{verb phrase}\rangle$

2., 3. $\langle\text{noun phrase}\rangle \to \langle\text{article}\rangle\langle\text{noun}\rangle \mid \langle\text{article}\rangle\langle\text{adjective}\rangle\langle\text{noun}\rangle$

4. $\langle\text{verb phrase}\rangle \to \langle\text{verb}\rangle\langle\text{noun phrase}\rangle$

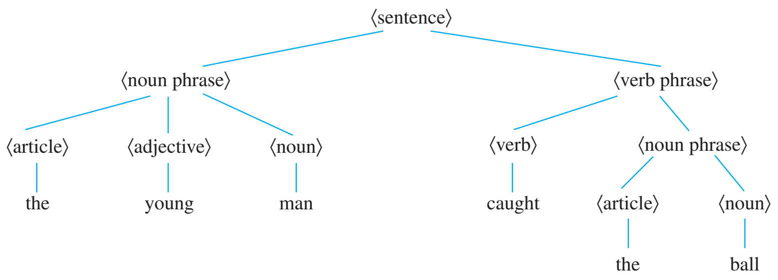
5. $\langle\text{article}\rangle \to \text{the}$

6. $\langle\text{adjective}\rangle \to \text{young}$

7, 8. $\langle\text{noun}\rangle \to \text{man} \mid \text{ball}$

9. $\langle\text{verb}\rangle \to \text{caught}$

The derivation of the sentence "The young man caught the ball" from the above rules is described by the tree shown below.

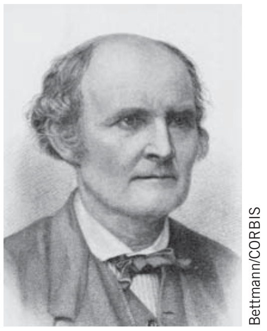


In the study of linguistics, **syntax** refers to the grammatical structure of sentences, and **semantics** refers to the meanings of words and their interrelations. A sentence can be syntactically correct but semantically incorrect, as in the nonsensical sentence "The young ball caught the man," which can be derived from the rules given above. Or a sentence can contain syntactic errors but not semantic ones, as, for instance, when a two-year-old child says, "Me hungry!" ■

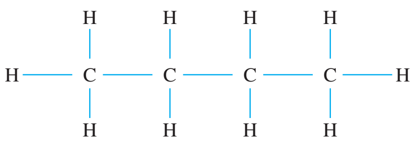
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

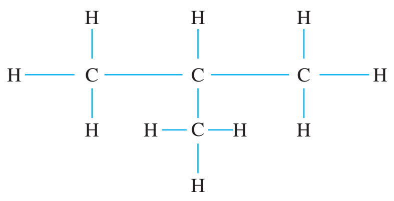
Example 10.5.4 Structure of Hydrocarbon Molecules

The German physicist Gustav Kirchhoff (1824–1887) was the first to analyze the behavior of mathematical trees in connection with the investigation of electrical circuits. Soon after (and independently), the English mathematician Arthur Cayley used the mathematics of trees to enumerate all isomers for certain hydrocarbons. Hydrocarbon molecules are composed of carbon and hydrogen; each carbon atom can form up to four chemical bonds with other atoms, and each hydrogen atom can form one bond with another atom. Thus the structure of hydrocarbon molecules can be represented by graphs such as those shown following, in which the vertices represent atoms of hydrogen and carbon, denoted H and C, and the edges represent the chemical bonds between them.


Bettmann/CORBIS

Arthur Cayley (1821–1895)

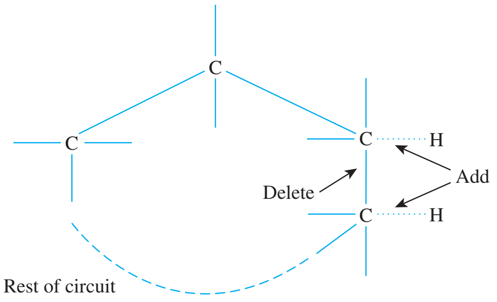

Butane


Isobutane

Note that each of these graphs has four carbon atoms and ten hydrogen atoms, but the two graphs show different configurations of atoms. When two molecules have the same chemical formulae (in this case C_4H_{10}) but different chemical bonds, they are called *isomers*.

Certain *saturated hydrocarbon* molecules contain the maximum number of hydrogen atoms for a given number of carbon atoms. Cayley showed that if such a saturated hydrocarbon molecule has k carbon atoms, then it has $2k + 2$ hydrogen atoms. The first step in doing so is to prove that the graph of such a saturated hydrocarbon molecule is a tree. Prove this using proof by contradiction. (You are asked to finish the derivation of Cayley's result in exercise 4 at the end of this section.)

Solution Suppose there is a hydrocarbon molecule that contains the maximum number of hydrogen atoms for the number of its carbon atoms and whose graph G is not a tree. *[We must derive a contradiction.]* Since G is not a tree, G is not connected or G has a circuit. But the graph of any molecule is connected (all the atoms in a molecule must be connected to each other), and so G must have a nontrivial circuit. Now the edges of the circuit can link only carbon atoms because every vertex of a circuit has degree at least 2 and a hydrogen atom vertex has degree 1. Delete one edge of the circuit and add two new edges to join each of the newly disconnected carbon atom vertices to a hydrogen atom vertex as shown below.



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

The resulting molecule has two more hydrogen atoms than the given molecule, but the number of carbon atoms is unchanged. This contradicts the supposition that the given molecule has the maximum number of hydrogen atoms for the given number of carbon atoms. Hence the supposition is false, and so G is a tree. ■

Characterizing Trees

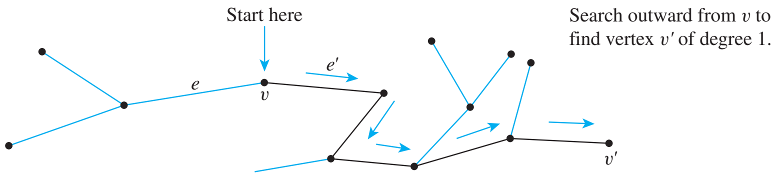
There is a somewhat surprising relation between the number of vertices and the number of edges of a tree. It turns out that if n is a positive integer, then any tree with n vertices (no matter what its shape) has $n - 1$ edges. Perhaps even more surprisingly, a partial converse to this fact is also true—namely, any *connected* graph with n vertices and $n - 1$ edges is a tree. It follows from these facts that if even one new edge (but no new vertex) is added to a tree, the resulting graph must contain a circuit. Also, from the fact that removing an edge from a circuit does not disconnect a graph, it can be shown that every connected graph has a subgraph that is a tree. It follows that if n is a positive integer, any graph with n vertices and *fewer* than $n - 1$ edges is not connected.

A small but very important fact necessary to derive the first main theorem about trees is that any nontrivial tree must have at least one vertex of degree 1.

Lemma 10.5.1

Any tree that has more than one vertex has at least one vertex of degree 1.

A constructive way to understand this lemma is to imagine being given a tree T with more than one vertex. You pick a vertex v at random and then search outward along a path from v looking for a vertex of degree 1. As you reach each new vertex, you check whether it has degree 1. If it does, you are finished. If it does not, you exit from the vertex along a different edge from the one you entered on. Because T is circuit-free, the vertices included in the path never repeat. And since the number of vertices of T is finite, the process of building a path must eventually terminate. When that happens, the final vertex v' of the path must have degree 1. This process is illustrated below.



This discussion is made precise in the following proof.

Proof:

Let T be a particular but arbitrarily chosen tree that has more than one vertex, and consider the following algorithm:

Step 1: Pick a vertex v of T and let e be an edge incident on v.
[If there were no edge incident on v, then v would be an isolated vertex. But this would contradict the assumption that T is connected (since it is a tree) and has at least two vertices.]

Step 2: While $\deg(v) > 1$, repeat steps 2a, 2b, and 2c:

continued on page 688

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Step 2a: Choose e' to be an edge incident on v such that $e' \neq e$. *[Such an edge exists because* $\deg(v) > 1$ *and so there are at least two edges incident on* v.*]*

Step 2b: Let v' be the vertex at the other end of e' from v. *[Since T is a tree, e' cannot be a loop and therefore e' has two distinct endpoints.]*

Step 2c: Let $e = e'$ and $v = v'$. *[This is just a renaming process in preparation for a repetition of step 2.]*

The algorithm just described must eventually terminate because the set of vertices of the tree T is finite and T is circuit-free. When it does, a vertex v of degree 1 will have been found.

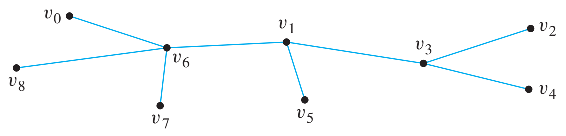
Using Lemma 10.5.1 it is not difficult to show that, in fact, any tree that has more than one vertex has at least *two* vertices of degree 1. This extension of Lemma 10.5.1 is left to the exercises at the end of this section.

• Definition

Let T be a tree. If T has only one or two vertices, then each is called a **terminal vertex.** If T has at least three vertices, then a vertex of degree 1 in T is called a **terminal vertex** (or a **leaf**), and a vertex of degree greater than 1 in T is called an **internal vertex** (or a **branch vertex**).

Example 10.5.5 Terminal and Internal Vertices

Find all terminal vertices and all internal vertices in the following tree:



Solution The terminal vertices are v_0, v_2, v_4, v_5, v_7, and v_8. The internal vertices are v_6, v_1, and v_3. ■

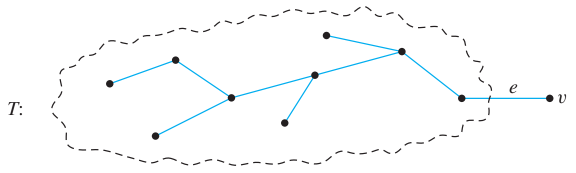
The following is the first of the two main theorems about trees:

Theorem 10.5.2

For any positive integer n, any tree with n vertices has $n - 1$ edges.

The proof is by mathematical induction. To do the inductive step, you assume the theorem is true for a positive integer k and then show it is true for $k + 1$. Thus you assume you have a tree T with $k + 1$ vertices, and you must show that T has $(k + 1) - 1 = k$ edges. As you do this, you are free to use the inductive hypothesis that *any* tree with k vertices has $k - 1$ edges. To make use of the inductive hypothesis, you need to reduce the tree T with $k + 1$ vertices to a tree with just k vertices. But by Lemma 10.5.1, T has a vertex v of degree 1, and since T is connected, v is attached to the rest of T by a single edge e as sketched on the next page.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



Rest of T

Now if e and v are removed from T, what remains is a tree T' with $(k+1)-1=k$ vertices. By inductive hypothesis, then, T' has $k-1$ edges. But the original tree T has one more vertex and one more edge than T'. Hence T must have $(k-1)+1=k$ edges, as was to be shown. A formal version of this argument is given below.

Proof (by mathematical induction):

Let the property $P(n)$ be the sentence

Any tree with n vertices has $n-1$ edges. $\leftarrow P(n)$

We use mathematical induction to show that this property is true for all integers $n \geq 1$.

***Show that P(1) is true*:** Let T be any tree with one vertex. Then T has zero edges (since it contains no loops). But $0 = 1 - 1$, so $P(1)$ is true.

***Show that for all integers k ≥ 1, if P(k) is true then P(k + 1) is true*:** Suppose k is any positive integer for which $P(k)$ is true. In other words, suppose that

Any tree with k vertices has k - 1 edges. $\leftarrow P(k)$ inductive hypothesis

We must show that $P(k+1)$ is true. In other words, we must show that

Any tree with $k+1$ vertices has $(k+1)-1=k$ edges. $\leftarrow P(k+1)$

Let T be a particular but arbitrarily chosen tree with $k+1$ vertices. *[We must show that T has k edges.]* Since k is a positive integer, $(k+1) \geq 2$, and so T has more than one vertex. Hence by Lemma 10.5.1, T has a vertex v of degree 1. Also, since T has more than one vertex, there is at least one other vertex in T besides v. Thus there is an edge e connecting v to the rest of T. Define a subgraph T' of T so that

$$V(T') = V(T) - \{v\}$$

Then

$$E(T') = E(T) - \{e\}.$$

1. The number of vertices of T' is $(k+1)-1=k$.
2. T' is circuit-free (since T is circuit-free, and removing an edge and a vertex cannot create a circuit).
3. T' is connected (see exercise 24 at the end of this section).

Hence, by the definition of tree, T' is a tree. Since T' has k vertices, by inductive hypothesis

$$\begin{aligned}\text{the number of edges of } T' &= (\text{the number of vertices of } T') - 1 \\ &= k - 1.\end{aligned}$$

continued on page 690

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

But then

$$\begin{aligned}\text{the number of edges of } T &= (\text{the number of edges of } T') + 1\\ &= (k-1)+1\\ &= k.\end{aligned}$$

[This is what was to be shown.]

Example 10.5.6 Determining Whether a Graph Is a Tree

A graph G has ten vertices and twelve edges. Is it a tree?

Solution No. By Theorem 10.5.2, any tree with ten vertices has nine edges, not twelve. ■

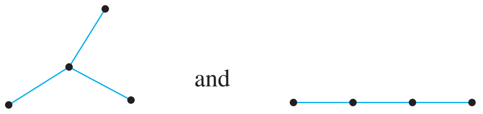
Example 10.5.7 Finding Trees Satisfying Given Conditions

Find all nonisomorphic trees with four vertices.

Solution By Theorem 10.5.2, any tree with four vertices has three edges. Thus the total degree of a tree with four vertices must be 6. Also, every tree with more than one vertex has at least two vertices of degree 1 (see the comment following Lemma 10.5.1 and exercises 5 and 29 at the end of this section). Thus the following combinations of degrees for the vertices are the only ones possible:

$$1, 1, 1, 3 \quad \text{and} \quad 1, 1, 2, 2.$$

There are two nonisomorphic trees corresponding to both of these possibilities, as shown below.



■

To prove the second major theorem about trees, we need another lemma.

Lemma 10.5.3

If G is any connected graph, C is any circuit in G, and any one of the edges of C is removed from G, then the graph that remains is connected.

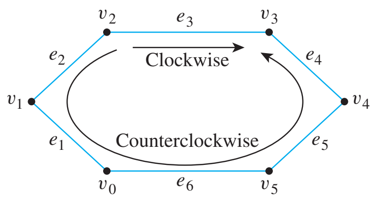
Essentially, the reason why Lemma 10.5.3 is true is that any two vertices in a circuit are connected by two distinct paths. It is possible to draw the graph so that one of these goes "clockwise" and the other goes "counterclockwise" around the circuit. For example, in the circuit shown on the next page, the clockwise path from v_2 to v_3 is

$$v_2 e_3 v_3$$

and the counterclockwise path from v_2 to v_3 is

$$v_2 e_2 v_1 e_1 v_0 e_6 v_5 e_5 v_4 e_4 v_3.$$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.



Proof:

Suppose G is a connected graph, C is a circuit in G, and e is an edge of C. Form a subgraph G' of G by removing e from G. Thus

$$V(G') = V(G)$$
$$E(G') = E(G) - \{e\}.$$

We must show that G' is connected. *[To show a graph is connected, we must show that if u and w are any vertices of the graph, then there exists a walk in G' from u to w.]* Suppose u and w are any two vertices of G'. *[We must find a walk from u to w.]* Since the vertex sets of G and G' are the same, u and w are both vertices of G, and since G is connected, there is a walk W in G from u to w.

Case 1 (e is not an edge of W): The only edge in G that is not in G' is e, so in this case W is also a walk in G'. Hence u is connected to w by a walk in G'.

Case 2 (e is an edge of W): In this case the walk W from u to w includes a section of the circuit C that contains e. Let C be denoted as follows:

$$C\colon v_0e_1v_1e_2v_2 \cdots e_nv_n\ (= v_0).$$

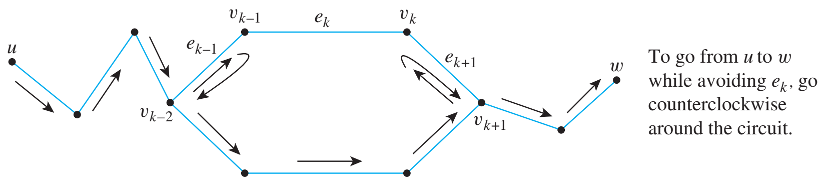
Now e is one of the edges of C, so, to be specific, let $e = e_k$. Then the walk W contains either the sequence

$$v_{k-1}e_kv_k \quad \text{or} \quad v_ke_kv_{k-1}.$$

If W contains $v_{k-1}e_kv_k$, connect v_{k-1} to v_k by taking the "counterclockwise" walk W' defined as follows:

$$W'\colon v_{k-1}e_{k-1}v_{k-2} \cdots v_0e_nv_{n-1} \cdots e_{k+1}v_k.$$

An example showing how to go from u to w while avoiding e_k is given in Figure 10.5.4.



u
v_{k-1}
e_k
v_k
e_{k-1}
e_{k+1}
w
v_{k-2}
v_{k+1}
To go from u to w while avoiding e_k, go counterclockwise around the circuit.

Figure 10.5.4 **An Example of a Walk from u to w That Does Not Include Edge e_k**

If W contains $v_ke_kv_{k-1}$, connect v_k to v_{k-1} by taking the "clockwise" walk W'' defined as follows:

$$W''\colon v_ke_{k+1}v_{k+1} \cdots v_ne_1v_1e_2 \cdots e_{k-1}v_{k-1}.$$

continued on page 692

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Now patch either W' or W'' into W to form a new walk from u to w. For instance, to patch W' into W, start with the section of W from u to v_{k-1}, then take W' from v_{k-1} to v_k, and finally take the section of W from v_k to w. If this new walk still contains an occurrence of e, just repeat the process described previously until all occurrences are eliminated. *[This must happen eventually since the number of occurrences of e in C is finite.]* The result is a walk from u to w that does not contain e and hence is a walk in G'.

The previous arguments show that both in case 1 and in case 2 there is a walk in G' from u to w. Since the choice of u and w was arbitrary, G' is connected.

The second major theorem about trees is a modified converse to Theorem 10.5.2.

Theorem 10.5.4

For any positive integer n, if G is a connected graph with n vertices and $n-1$ edges, then G is a tree.

Proof:

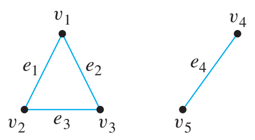
Let n be a positive integer and suppose G is a particular but arbitrarily chosen graph that is connected and has n vertices and $n-1$ edges. *[We must show that G is a tree. Now a tree is a connected, circuit-free graph. Since we already know G is connected, it suffices to show that G is circuit-free.]* Suppose G is not circuit-free. That is, suppose G has a circuit C. *[We must derive a contradiction.]* By Lemma 10.5.3, an edge of C can be removed from G to obtain a graph G' that is connected. If G' has a circuit, then repeat this process: Remove an edge of the circuit from G' to form a new connected graph. Continue repeating the process of removing edges from circuits until eventually a graph G'' is obtained that is connected and is circuit-free. By definition, G'' is a tree. Since no vertices were removed from G to form G'', G'' has n vertices just as G does. Thus, by Theorem 10.5.2, G'' has $n-1$ edges. But the supposition that G has a circuit implies that at least one edge of G is removed to form G''. Hence G'' has no more than $(n-1)-1 = n-2$ edges, which contradicts its having $n-1$ edges. So the supposition is false. Hence G is circuit-free, and therefore G is a tree *[as was to be shown].*

Theorem 10.5.4 is not a full converse of Theorem 10.5.2. Although it is true that every *connected* graph with n vertices and $n-1$ edges (where n is a positive integer) is a tree, it is not true that *every* graph with n vertices and $n-1$ edges is a tree.

Example 10.5.8 A Graph with n Vertices and $n-1$ Edges That Is Not a Tree

Give an example of a graph with five vertices and four edges that is not a tree.

Solution By Theorem 10.5.4, such a graph cannot be connected. One example of such an unconnected graph is shown below.



■

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Test Yourself

1. A circuit-free graph is a graph with _____.
2. A forest is a graph that is _____, and a tree is a graph that is _____.
3. A trivial tree is a graph that consists of _____.
4. Any tree with at least two vertices has at least one vertex of degree _____.
5. If a tree T has at least two vertices, then a terminal vertex (or leaf) in T is a vertex of degree _____ and an internal vertex (or branch vertex) in T is a vertex of degree _____.
6. For any positive integer n, any tree with n vertices has _____.
7. For any positive integer n, if G is a connected graph with n vertices and $n-1$ edges then _____.

Exercise Set 10.5

1. Read the tree in Example 10.5.2 from left to right to answer the following questions:
   a. What course should a student who scored 12 on part I and 4 on part II take?
   b. What course should a student who scored 8 on part I and 9 on part II take?
2. Draw trees to show the derivations of the following sentences from the rules given in Example 10.5.3.
   a. The young ball caught the man.
   b. The man caught the young ball.

H **3.** What is the total degree of a tree with n vertices? Why?

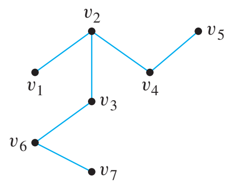
4. Let G be the graph of a hydrocarbon molecule with the maximum number of hydrogen atoms for the number of its carbon atoms.
   a. Draw the graph of G if G has three carbon atoms and eight hydrogen atoms.
   b. Draw the graphs of three isomers of C_5H_{12}.
   c. Use Example 10.5.4 and exercise 3 to prove that if the vertices of G consist of k carbon atoms and m hydrogen atoms, then G has a total degree of $2k + 2m - 2$.
   H **d.** Prove that if the vertices of G consist of k carbon atoms and m hydrogen atoms, then G has a total degree of $4k + m$.
   e. Equate the results of (c) and (d) to prove Cayley's result that a saturated hydrocarbon molecule with k carbon atoms and a maximum number of hydrogen atoms has $2k + 2$ hydrogen atoms.

H **5.** Extend the argument given in the proof of Lemma 10.5.1 to show that a tree with more than one vertex has at least two vertices of degree 1.

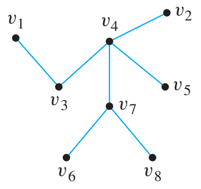
6. If graphs are allowed to have an infinite number of vertices and edges, then Lemma 10.5.1 is false. Give a counterexample that shows this. In other words, give an example of an "infinite tree" (a connected, circuit-free graph with an infinite number of vertices and edges) that has no vertex of degree 1.

7. Find all terminal vertices and all internal vertices for the following trees.

a.



b.



In each of 8–21, either draw a graph with the given specifications or explain why no such graph exists.

8. Tree, nine vertices, nine edges

9. Graph, connected, nine vertices, nine edges

10. Graph, circuit-free, nine vertices, six edges

11. Tree, six vertices, total degree 14

12. Tree, five vertices, total degree 8

13. Graph, connected, six vertices, five edges, has a nontrivial circuit

14. Graph, two vertices, one edge, not a tree

15. Graph, circuit-free, seven vertices, four edges

16. Tree, twelve vertices, fifteen edges

17. Graph, six vertices, five edges, not a tree

18. Tree, five vertices, total degree 10

19. Graph, connected, ten vertices, nine edges, has a nontrivial circuit

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

20. Simple graph, connected, six vertices, six edges

21. Tree, ten vertices, total degree 24

22. A connected graph has twelve vertices and eleven edges. Does it have a vertex of degree 1? Why?

23. A connected graph has nine vertices and twelve edges. Does it have a nontrivial circuit? Why?

24. Suppose that v is a vertex of degree 1 in a connected graph G and that e is the edge incident on v. Let G' be the subgraph of G obtained by removing v and e from G. Must G' be connected? Why?

25. A graph has eight vertices and six edges. Is it connected? Why?

H **26.** If a graph has n vertices and $n-2$ or fewer edges, can it be connected? Why?

27. A circuit-free graph has ten vertices and nine edges. Is it connected? Why?

H **28.** Is a circuit-free graph with n vertices and at least $n-1$ edges connected? Why?

29. Prove that every nontrivial tree has at least two vertices of degree 1 by filling in the details and completing the following argument: Let T be a nontrivial tree and let S be the set of all paths from one vertex to another of T. Among all the paths in S, choose a path P with the most edges. (Why is it possible to find such a P?) What can you say about the initial and final vertices of P? Why?

30. Find all nonisomorphic trees with five vertices.

31. a. Prove that the following is an invariant for graph isomorphism: A vertex of degree i is adjacent to a vertex of degree j.

H **b.** Find all nonisomorphic trees with six vertices.

Answers for Test Yourself

1. no circuits 2. circuit-free and not connected; connected and circuit-free 3. a single vertex (and no edges) 4. 1 5. 1; greater than 1 (*Or*: at least 2) 6. $n-1$ edges 7. G is a tree

10.6 *Rooted Trees*

Let us grant that the pursuit of mathematics is a divine madness of the human spirit, a refuge from the goading urgency of contingent happenings. — Alfred North Whitehead, 1861–1947

An outdoor tree is rooted and so is the kind of family tree that shows all the descendants of one particular person. The terminology and notation of rooted trees blends the language of botanical trees and that of family trees. In mathematics, a rooted tree is a tree in which one vertex has been distinguished from the others and is designated the *root*. Given any other vertex v in the tree, there is a unique path from the root to v. (After all, if there were two distinct paths, a circuit could be constructed.) The number of edges in such a path is called the *level* of v, and the *height* of the tree is the length of the longest such path. It is traditional in drawing rooted trees to place the root at the top (as is done in family trees) and show the branches descending from it.

• Definition

A **rooted tree** is a tree in which there is one vertex that is distinguished from the others and is called the **root.** The **level** of a vertex is the number of edges along the unique path between it and the root. The **height** of a rooted tree is the maximum level of any vertex of the tree. Given the root or any internal vertex v of a rooted tree, the **children** of v are all those vertices that are adjacent to v and are one level farther away from the root than v. If w is a child of v, then v is called the **parent** of w, and two distinct vertices that are both children of the same parent are called **siblings.** Given two distinct vertices v and w, if v lies on the unique path between w and the root, then v is an **ancestor** of w and w is a **descendant** of v.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

These terms are illustrated in Figure 10.6.1.

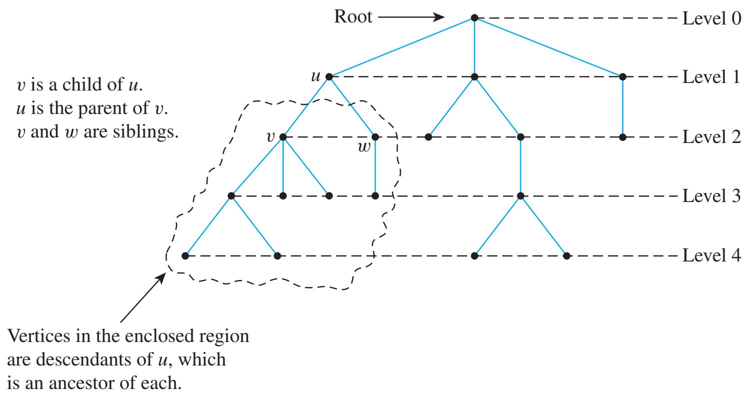

Root
Level 0
Level 1
Level 2
Level 3
Level 4
u
v
w
v is a child of u.
u is the parent of v.
v and w are siblings.
Vertices in the enclosed region are descendants of u, which is an ancestor of each.

Figure 10.6.1 A Rooted Tree

Example 10.6.1 Rooted Trees

Consider the tree with root v_0 shown below.

a. What is the level of v_5?

b. What is the level of v_0?

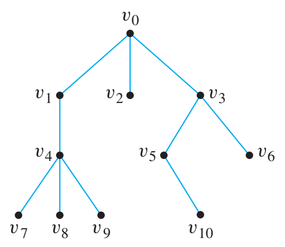
c. What is the height of this rooted tree?

d. What are the children of v_3?

e. What is the parent of v_2?

f. What are the siblings of v_8?

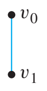
g. What are the descendants of v_3?



Solution

a. 2 b. 0 c. 3 d. v_5 and v_6 e. v_0 f. v_7 and v_9 g. v_5, v_6, v_{10} ■

Note that in the tree with root v_0 shown below, v_1 has level 1 and is the child of v_0, and both v_0 and v_1 are terminal vertices.



Binary Trees

When every vertex in a rooted tree has at most two children and each child is designated either the (unique) left child or the (unique) right child, the result is a *binary tree*.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

• Definition

A **binary tree** is a rooted tree in which every parent has at most two children. Each child in a binary tree is designated either a **left child** or a **right child** (but not both), and every parent has at most one left child and one right child. A **full binary tree** is a binary tree in which each parent has exactly two children.

Given any parent v in a binary tree T, if v has a left child, then the **left subtree** of v is the binary tree whose root is the left child of v, whose vertices consist of the left child of v and all its descendants, and whose edges consist of all those edges of T that connect the vertices of the left subtree. The **right subtree** of v is defined analogously.

These terms are illustrated in Figure 10.6.2.

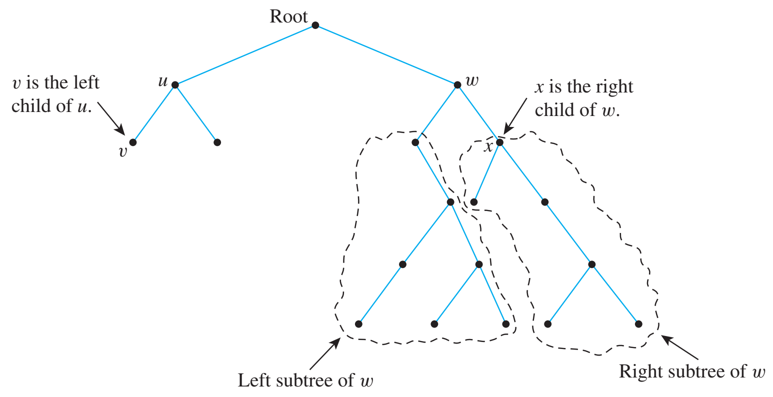
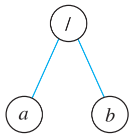


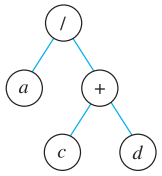
Figure 10.6.2 A Binary Tree

Example 10.6.2 Representation of Algebraic Expressions

Binary trees are used in many ways in computer science. One use is to represent algebraic expressions with arbitrary nesting of balanced parentheses. For instance, the following (labeled) binary tree represents the expression a/b: The operator is at the root and acts on the left and right children of the root in left-right order.



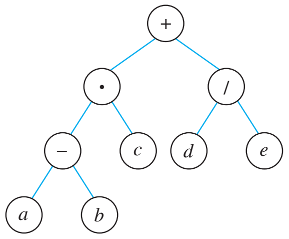
More generally, the binary tree shown below represents the expression $a/(c+d)$. In such a representation, the internal vertices are arithmetic operators, the terminal vertices are variables, and the operator at each vertex acts on its left and right subtrees in left-right order.



Draw a binary tree to represent the expression $((a-b)\cdot c)+(d/e)$.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Solution



■

An interesting theorem about binary trees says that if you know the number of internal vertices of a full binary tree, then you can calculate both the total number of vertices and the number of terminal vertices, and conversely. More specifically, a full binary tree with k internal vertices has a total of $2k + 1$ vertices of which $k + 1$ are terminal vertices.

Theorem 10.6.1

If k is a positive integer and T is a full binary tree with k internal vertices, then T has a total of $2k + 1$ vertices and has $k + 1$ terminal vertices.

Proof:

Suppose k is a positive integer and T is a full binary tree with k internal vertices. Observe that the set of all vertices of T can be partitioned into two disjoint subsets: the set of all vertices that have a parent and the set of all vertices that do not have a parent. Now there is just one vertex that does not have a parent, namely the root. Also, since every internal vertex of a full binary tree has exactly two children, the number of vertices that have a parent is twice the number of parents, or $2k$, since each parent is an internal vertex. Hence

$$\begin{aligned}\begin{bmatrix}\text{the total number}\\ \text{of vertices of } T\end{bmatrix} &= \begin{bmatrix}\text{the number of}\\ \text{vertices that}\\ \text{have a parent}\end{bmatrix} + \begin{bmatrix}\text{the number of}\\ \text{vertices that do}\\ \text{not have a parent}\end{bmatrix}\\ &= \qquad 2k \qquad + \qquad 1.\end{aligned}$$

But it is also true that the total number of vertices of T equals the number of internal vertices plus the number of terminal vertices. Thus

$$\begin{aligned}\begin{bmatrix}\text{the total number}\\ \text{of vertices of } T\end{bmatrix} &= \begin{bmatrix}\text{the number of}\\ \text{internal vertices}\end{bmatrix} + \begin{bmatrix}\text{the number of}\\ \text{terminal vertices}\end{bmatrix}\\ &= \qquad k \qquad + \begin{bmatrix}\text{the number of}\\ \text{terminal vertices}\end{bmatrix}\end{aligned}$$

Now equate the two expressions for the total number of vertices of T:

$$2k + 1 = k + \begin{bmatrix}\text{the number of}\\ \text{terminal vertices}\end{bmatrix}$$

Solving this equation gives

$$\begin{bmatrix}\text{the number of}\\ \text{terminal vertices}\end{bmatrix} = (2k + 1) - k = k + 1.$$

Thus the total number of vertices is $2k + 1$ and the number of terminal vertices is $k + 1$ *[as was to be shown].*

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.6.3 Determining Whether a Certain Full Binary Tree Exists

Is there a full binary tree that has 10 internal vertices and 13 terminal vertices?

Solution No. By Theorem 10.6.1, a full binary tree with 10 internal vertices has $10 + 1 = 11$ terminal vertices, not 13. ■

Another interesting theorem about binary trees specifies the maximum number of terminal vertices of a binary tree of a given height. Specifically, the maximum number of terminal vertices of a binary tree of height h is 2^h. Another way to say this is that a binary tree with t terminal vertices has height of at least $\log_2 t$.

Theorem 10.6.2

For all integers $h \geq 0$, if T is any binary tree with of height h and t terminal vertices, then

$$t \leq 2^h.$$

Equivalently, $$\log_2 t \leq h.$$

Proof (by strong mathematical induction):

Let $P(h)$ be the sentence

> If T is any binary tree of height h, then the number of terminal vertices of T is at most 2^h. $\leftarrow P(h)$

Show that $P(0)$ is true: We must show that if T is any binary tree of height 0, then the number of terminal vertices of T is at most 2^0. Suppose T is a tree of height 0. Then T consists of a single vertex, the root. By definition this is a terminal vertex and so the number of terminal vertices is $t = 1 = 2^\circ = 2^h$. Hence $t \leq 2^h$ *[as was to be shown].*

Show that for all integers $k \geq 0$, if $P(i)$ is true for all integers i from 0 through k, then it is true for $k + 1$:
Let k be any integer with $k \geq 0$, and suppose that

> For all integers i from 0 through k, if T is any binary tree of height i, then the number of terminal vertices of T is at most 2^i. $\leftarrow$ inductive hypothesis

We must show that

> If T is any binary tree of height $k + 1$, then the number of terminal vertices of T is 2^{k+1}. $\leftarrow P(k+1)$

Let T be a binary tree of height $k + 1$, root v, and t terminal vertices. Because $k \geq 0$, we have that $k + 1 \geq 1$ and so v has at least one child.

Case 1 (v has only one child): In this case we may assume without loss of generality that v's child is a left child and denote it by v_L. Let T_L be the left subtree of v. Then v_L is the root of T_L. (This situation is illustrated in Figure 10.6.3.) Because v has only one child, v is itself a terminal vertex, so the total number of terminal vertices in T equals the number of terminal vertices in T_L plus 1. Thus if t_L is the number of terminal vertices in T_L, then $t = t_L + 1$.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Now by inductive hypothesis, $t_L \leq 2^k$ because the height of T_L is k, one less than the height of T. Also, because v has a child, $k+1 \geq 1$, and so $2^k \geq 2^0 = 1$. Therefore,

$$t \;=\; t_L + 1 \;\leq\; 2^k + 1 \;\leq\; 2^k + 2^k \;= 2 \cdot 2^k = 2^{(k+1)}.$$

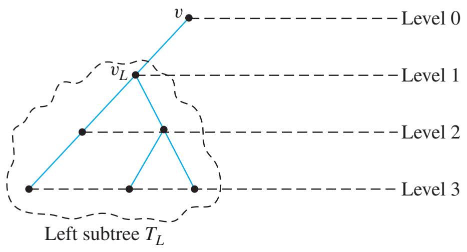


Figure 10.6.3 **A Binary Tree Whose Root Has One Child**

Case 2 (v has two children): In this case, v has both a left child, v_L, and a right child, v_R, and v_L and v_R are roots of a left subtree T_L and a right subtree T_R. Note that T_L and T_R are binary trees because T is a binary tree. (This situation is illustrated in Figure 10.6.4.)

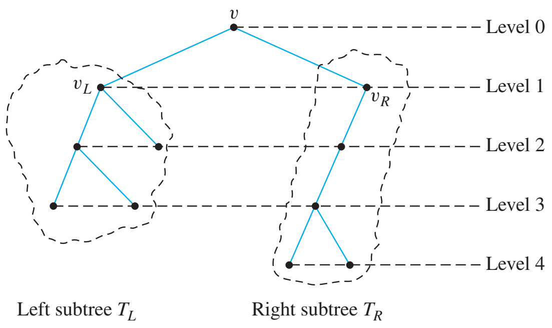


Figure 10.6.4 **A Binary Tree Whose Root Has Two Children**

Now v_L and v_R are the roots of the left and right subtrees of v, denoted T_L and T_R, respectively. Note that T_L and T_R are binary trees because T is a binary tree. Let h_L and h_R be the heights of T_L and T_R, respectively. Then $h_L \leq k$ and $h_R \leq k$ since T is obtained by joining T_L and T_R and adding a level. Let t_L and t_R be the numbers of terminal vertices of T_L and T_R, respectively. Then, since both T_L and T_R have heights less than $k+1$, by inductive hypothesis

$$t_L \leq 2^{h_L} \quad \text{and} \quad t_R \leq 2^{h_R}.$$

But the terminal vertices of T consist exactly of the terminal vertices of T_L together with the terminal vertices of T_R. Therefore,

$$t = t_L + t_R \leq 2^{h_L} + 2^{h_R}$$ by inductive hypothesis since $h_L \leq k$ and $h_R \leq k$

continued on page 700

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Hence,

$$t \le 2^k + 2^k = 2 \cdot 2^k = 2^{k+1} \qquad \text{by basic algebra.}$$

Thus the number of terminal vertices is at most $2k + 1$ *[as was to be shown].*

Since both the basis step and the inductive step have been proved, we conclude that for all integers $h \ge 0$, if T is any binary tree with height h and t terminal vertices, then $t \le 2^h$.

The equivalent inequality $\log_2 t \le h$ follows from the fact that the logarithmic function with base 2 is increasing. In other words, for all positive real numbers x and y,

$$\text{if } x < y \text{ then } \log_2 x < \log_2 y.$$

Thus if we apply the logarithmic function with base 2 to both sides of

$$t \le 2^h,$$

we obtain

$$\log_2 t \le \log_2(2^h).$$

Now by definition of logarithm, $\log_2(2^h) = h$ *[because* $\log_2(2^h)$ *is the exponent to which* 2 *must be raised to obtain* 2^h*].* Hence

$$\log_2 t \le h$$

[as was to be shown].

Example 10.6.4 Determining Whether a Certain Binary Tree Exists

Is there a binary tree that has height 5 and 38 terminal vertices?

Solution No. By Theorem 10.6.2, any binary tree T with height 5 has at most $2^5 = 32$ terminal vertices, so such a tree cannot have 38 terminal vertices. ■

Test Yourself

1. A rooted tree is a tree in which _____. The level of a vertex in a rooted tree is _____. The height of a rooted tree is _____.

2. A binary tree is a rooted tree in which _____.

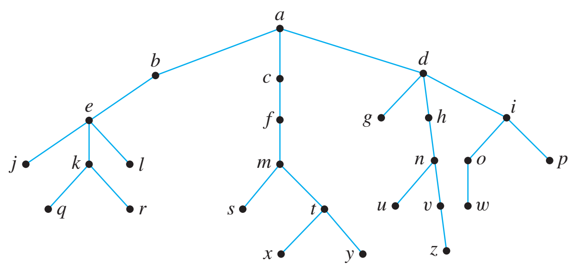
3. A full binary tree is a rooted tree in which _____.

4. If k is a positive integer and T is a full binary tree with k internal vertices, then T has a total of _____ vertices and has _____ terminal vertices.

5. If T is a binary tree that has t terminal vertices and height h, then t and h are related by the inequality _____.

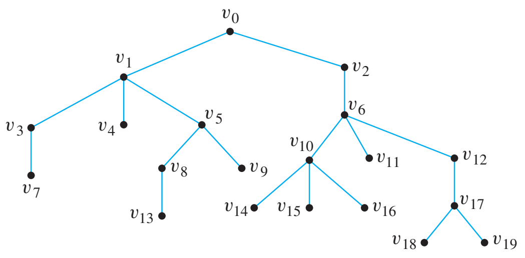
Exercise Set 10.6

1. Consider the tree shown at right with root a.
a. What is the level of n?
b. What is the level of a?
c. What is the height of this rooted tree?
d. What are the children of n?
e. What is the parent of g?
f. What are the siblings of j?
g. What are the descendants of f?



Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

2. Consider the tree shown below with root v_0.
   a. What is the level of v_8?
   b. What is the level of v_0?
   c. What is the height of this rooted tree?
   d. What are the children of v_{10}?
   e. What is the parent of v_5?
   f. What are the siblings of v_1?
   g. What are the descendants of v_{12}?



3. Draw binary trees to represent the following expressions:
   a. $a \cdot b - (c/(d+e))$ b. $a/(b - c \cdot d)$

In each of 4–20 either draw a graph with the given specifications or explain why no such graph exists.

4. Full binary tree, five internal vertices

5. Full binary tree, five internal vertices, seven terminal vertices

6. Full binary tree, seven vertices, of which four are internal vertices

7. Full binary tree, twelve vertices

8. Full binary tree, nine vertices

9. Binary tree, height 3, seven terminal vertices

10. Full binary tree, height 3, six terminal vertices

11. Binary tree, height 3, nine terminal vertices

12. Full binary tree, eight internal vertices, seven terminal vertices.

13. Binary tree, height 4, eight terminal vertices

14. Full binary tree, seven vertices

15. Full binary tree, nine vertices, five internal vertices

16. Full binary tree, four internal vertices

17. Binary tree, height 4, eighteen terminal vertices

18. Full binary tree, sixteen vertices

19. Full binary tree, height 3, seven terminal vertices

20. What can you deduce about the height of a binary tree if you know that it has the following properties?
   a. Twenty-five terminal vertices
   b. Forty terminal vertices
   c. Sixty terminal vertices

Answers for Test Yourself

1. one vertex is distinguished from the others and is called the root; the number of edges along the unique path between it and the root; the maximum level of any vertex of the tree 2. every parent has at most two children 3. every parent has exactly two children 4. $2k+1$; $k+1$ 5. $t \le 2^h$, or, equivalently, $\log_2 t \le h$

10.7 Spanning Trees and Shortest Paths

I contend that each science is a real science insofar as it is mathematics.
— Immanuel Kant, 1724–1804

An East Coast airline company wants to expand service to the Midwest and has received permission from the Federal Aviation Authority to fly any of the routes shown in Figure 10.7.1.

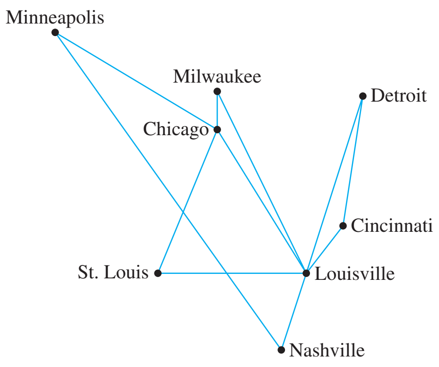


Figure 10.7.1

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s).
Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

The company wishes to legitimately advertise service to all the cities shown but, for reasons of economy, wants to use the least possible number of individual routes to connect them. One possible route system is given in Figure 10.7.2.

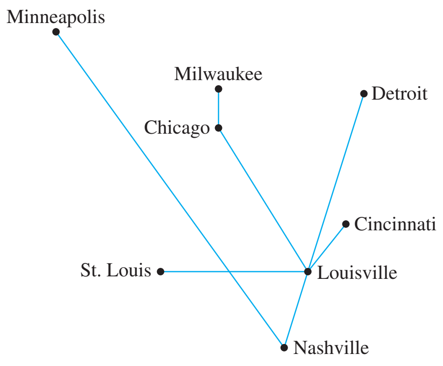


Figure 10.7.2

Clearly this system joins all the cities. Is the number of individual routes minimal? The answer is yes, and the reason may surprise you.

The fact is that the graph of any system of routes that satisfies the company's wishes is a tree, because if the graph were to contain a circuit, then one of the routes in the circuit could be removed without disconnecting the graph (by Lemma 10.5.3), and that would give a smaller total number of routes. But any tree with eight vertices has seven edges. Therefore, any system of routes that connects all eight vertices and yet minimizes the total number of routes consists of seven routes.

• Definition

A **spanning tree** for a graph G is a subgraph of G that contains every vertex of G and is a tree.

The preceding discussion contains the essence of the proof of the following proposition:

Proposition 10.7.1

1. Every connected graph has a spanning tree.
2. Any two spanning trees for a graph have the same number of edges.

Proof of (1):

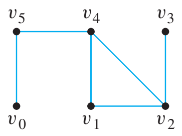
Suppose G is a connected graph. If G is circuit-free, then G is its own spanning tree and we are done. If not, then G has at least one circuit C_1. By Lemma 10.5.3, the subgraph of G obtained by removing an edge from C_1 is connected. If this subgraph is circuit-free, then it is a spanning tree and we are done. If not, then it has at least one circuit C_2, and, as above, an edge can be removed from C_2 to obtain a connected subgraph. Continuing in this way, we can remove successive edges from circuits, until eventually we obtain a connected, circuit-free subgraph T of G. *[This must happen at some point because the number of edges of G is finite, and at no stage does removal of an edge disconnect the subgraph.]* Also, T contains every vertex of G because no vertices of G were removed in constructing it. Thus T is a spanning tree for G.

The proof of part (2) is left as an exercise.

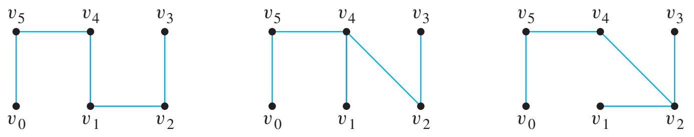
Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Example 10.7.1 Spanning Trees

Find all spanning trees for the graph G pictured below.



Solution The graph G has one circuit $v_2v_1v_4v_2$, and removal of any edge of the circuit gives a tree. Thus, as shown below, there are three spanning trees for G.



■

Minimum Spanning Trees

The graph of the routes allowed by the Federal Aviation Authority shown in Figure 10.7.1 can be annotated by adding the distances (in miles) between each pair of cities. This is done in Figure 10.7.3.

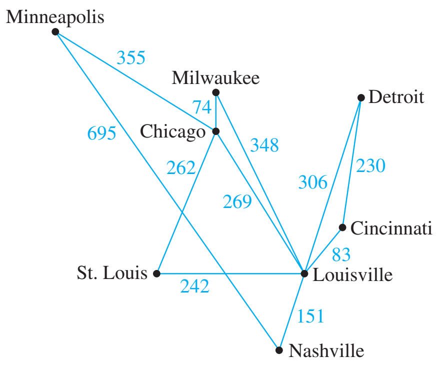


Figure 10.7.3

Now suppose the airline company wants to serve all the cities shown, but with a route system that minimizes the total mileage. Note that such a system is a tree, because if the system contained a circuit, removal of an edge from the circuit would not affect a person's ability to reach every city in the system from every other (again, by Lemma 10.5.3), but it would reduce the total mileage of the system.

More generally, a graph whose edges are labeled with numbers (known as *weights*) is called a *weighed graph*. A *minimum-weight spanning tree,* or simply a *minimum spanning tree,* is a spanning tree for which the sum of the weights of all the edges is as small as possible.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

• Definition

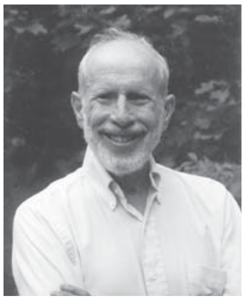
A **weighted graph** is a graph for which each edge has an associated positive real number **weight.** The sum of the weights of all the edges is the **total weight** of the graph. A **minimum spanning tree** for a connected weighted graph is a spanning tree that has the least possible total weight compared to all other spanning trees for the graph.

If G is a weighed graph and e is an edge of G, then $\boldsymbol{w(e)}$ denotes the weight of e and $\boldsymbol{w(G)}$ denotes the total weight of G.

The problem of finding a minimum spanning tree for a graph is certainly solvable. One solution is to list all spanning trees for the graph, compute the total weight of each, and choose one for which this total is a minimum. (Note that the well-ordering principle for the integers guarantees the existence of such a minimum total.) This solution, however, is inefficient in its use of computing time because the number of distinct spanning trees is so large. For instance, a complete graph with n vertices has n^{n-2} spanning trees. Even using the fastest computers available today, examining all such trees in a graph with approximately 100 vertices would require more time than is estimated to remain in the life of the universe.

In 1956 and 1957 Joseph B. Kruskal and Robert C. Prim each described much more efficient algorithms to construct minimum spanning trees. Even for large graphs, both algorithms can be implemented so as to take relatively short computing times.

Kruskal's Algorithm



Joseph Kruskal (born 1928)

In Kruskal's algorithm, the edges of a connected weighted graph are examined one by one in order of increasing weight. At each stage the edge being examined is added to what will become the minimum spanning tree, provided that this addition does not create a circuit. After $n-1$ edges have been added (where n is the number of vertices of the graph), these edges, together with the vertices of the graph, form a minimum spanning tree for the graph.

Algorithm 10.7.1 Kruskal

```
Input: G [a connected weighted graph with n vertices, where n is a positive integer]
Algorithm Body:
[Build a subgraph T of G to consist of all the vertices of G with edges added in order of
increasing weight. At each stage, let m be the number of edges of T.]

1. Initialize T to have all the vertices of G and no edges.

2. Let E be the set of all edges of G, and let $m := 0$.

3. while $(m < n - 1)$
     3a. Find an edge e in E of least weight.
     3b. Delete e from E.
     3c. if addition of e to the edge set of T does not produce a circuit
             then add e to the edge set of T and set $m := m + 1$
   end while
Output: T [T is a minimum spanning tree for G.]
```

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

The following example shows how Kruskal's algorithm works for the graph of the airline route system.

Example 10.7.2 Action of Kruskal's Algorithm

Describe the action of Kruskal's algorithm on the graph shown in Figure 10.7.4, where $n = 8$.

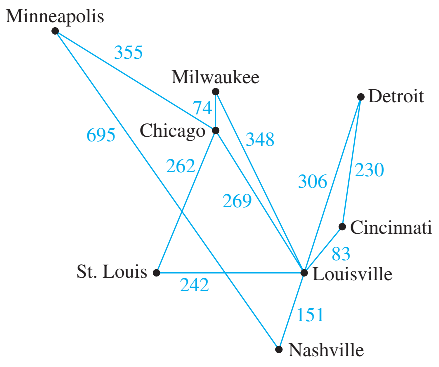


Figure 10.7.4

Solution

Iteration Number	Edge Considered	Weight	Action Taken
1	Chicago–Milwaukee	74	added
2	Louisville–Cincinnati	83	added
3	Louisville–Nashville	151	added
4	Cincinnati–Detroit	230	added
5	St. Louis–Louisville	242	added
6	St. Louis–Chicago	262	added
7	Chicago–Louisville	269	not added
8	Louisville–Detroit	306	not added
9	Louisville–Milwaukee	348	not added
10	Minneapolis–Chicago	355	added

The tree produced by Kruskal's algorithm is shown in Figure 10.7.5.

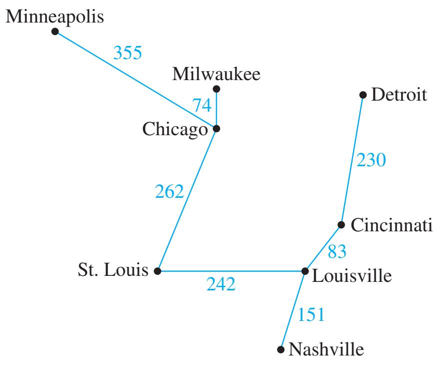


Figure 10.7.5 ■

When Kruskal's algorithm is used on a graph in which some edges have the same weight as others, more than one minimum spanning tree can occur as output. To make

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

the output unique, the edges of the graph can be placed in an array and edges having the same weight can be added in the order they appear in the array.

It is not obvious from the description of Kruskal's algorithm that it does what it is supposed to do. To be specific, what guarantees that it is possible at each stage to find an edge of least weight whose addition does not produce a circuit? And if such edges can be found, what guarantees that they will all eventually connect? And if they do connect, what guarantees that the resulting tree has minimum weight? Of course, the mere fact that Kruskal's algorithm is printed in this book may lead you to believe that everything works out. But the questions above are real, and they deserve serious answers.

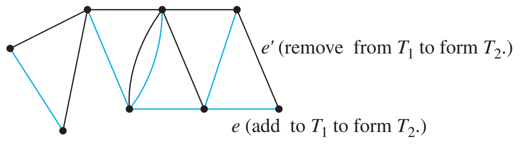
Theorem 10.7.2 Correctness of Kruskal's Algorithm

When a connected, weighted graph is input to Kruskal's algorithm, the output is a minimum spanning tree.

Proof:

Suppose that G is a connected, weighted graph with n vertices and that T is a subgraph of G produced when G is input to Kruskal's algorithm. Clearly T is circuit-free *[since no edge that completes a circuit is ever added to T].* Also T is connected. For as long as T has more than one connected component, the set of edges of G that can be added to T without creating a circuit is nonempty. *[The reason is that since G is connected, given any vertex v_1 in one connected component C_1 of T and any vertex v_2 in another connected component C_2, there is a path in G from v_1 to v_2. Since C_1 and C_2 are distinct, there is an edge e of this path that is not in T. Adding e to T does not create a circuit in T, because deletion of an edge from a circuit does not disconnect a graph and deletion of e would.]* The preceding arguments show that T is circuit-free and connected. Since by construction T contains every vertex of G, T is a spanning tree for G.

Next we show that T has minimum weight. Let T_1 be any minimum spanning tree for G such that the number of edges T_1 and T have in common is a maximum. Suppose that $T \neq T_1$. Then there is an edge e in T that is not an edge of T_1. *[Since trees T and T_1 both have the same vertex set, if they differ at all, they must have different, but same-size, edge sets.]* Now adding e to T_1 produces a graph with a unique circuit (see exercise 19 at the end of this section). Let e' be an edge of this circuit such that e' is not in T. *[Such an edge must exist because T is a tree and hence circuit-free.]* Let T_2 be the graph obtained from T_1 by removing e' and adding e. This situation is illustrated below.


e' (remove from T_1 to form T_2.)
e (add to T_1 to form T_2.)

The entire graph is G. T_1 has black edges. e is in T but not T_1. e' is in T_1 but not T.

Note that T_2 has $n - 1$ edges and n vertices and that T_2 is connected *[since by Lemma 10.5.3 the subgraph obtained by removing an edge from a circuit in a connected graph is connected].* Consequently, T_2 is a spanning tree for G. In addition,

$$w(T_2) = w(T_1) - w(e') + w(e).$$

Now $w(e) \leq w(e')$ because at the stage in Kruskal's algorithm when e was added to T, e' was available to be added *[since it was not already in T, and at that stage its*

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

addition could not produce a circuit since e was not in T], and e' *would* have been added had its weight been less than that of e. Thus

$$\begin{aligned} w(T_2) &= w(T_1) - \underbrace{[w(e') - w(e)]}_{\geq 0} \\ &\leq w(T_1). \end{aligned}$$

But T_1 is a minimum spanning tree. Since T_2 is a spanning tree with weight less than or equal to the weight of T_1, T_2 is also a minimum spanning tree for G.

Finally, note that by construction, T_2 has one more edge in common with T than T_1 does, which contradicts the choice of T_1 as a minimum spanning tree for G with a maximum number of edges in common with T. Thus the supposition that $T \neq T_1$ is false, and hence T itself is a minimum spanning tree for G.

Prim's Algorithm

Robert Prim (born 1921)

Prim's algorithm works differently from Kruskal's. It builds a minimum spanning tree T by expanding outward in connected links from some vertex. One edge and one vertex are added at each stage. The edge added is the one of least weight that connects the vertices already in T with those not in T, and the vertex is the endpoint of this edge that is not already in T.

Algorithm 10.7.2

Input: G *[a connected weighted graph with n vertices where n is a positive integer]*

Algorithm Body:

[Build a subgraph T of G by starting with any vertex v of G and attaching edges (with their endpoints) one by one to an as-yet-unconnected vertex of G, each time choosing an edge of least weight that is adjacent to a vertex of T.]

```
1. Pick a vertex $v$ of $G$ and let $T$ be the graph with one vertex, $v$, and no edges.
2. Let $V$ be the set of all vertices of $G$ except $v$.
3. for $i := 1$ to $n - 1$
     3a. Find an edge $e$ of $G$ such that (1) $e$ connects $T$ to one of the vertices in
         $V$, and (2) $e$ has the least weight of all edges connecting $T$ to a vertex
         in $V$. Let $w$ be the endpoint of $e$ that is in $V$.
     3b. Add $e$ and $w$ to the edge and vertex sets of $T$, and delete $w$ from $V$.
   next $i$
```

Output: T *[T is a minimum spanning tree for G.]*

The following example shows how Prim's algorithm works for the graph of the airline route system.

Example 10.7.3 Action of Prim's Algorithm

Describe the action of Prim's algorithm for the graph in Figure 10.7.6 using the Minneapolis vertex as a starting point.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

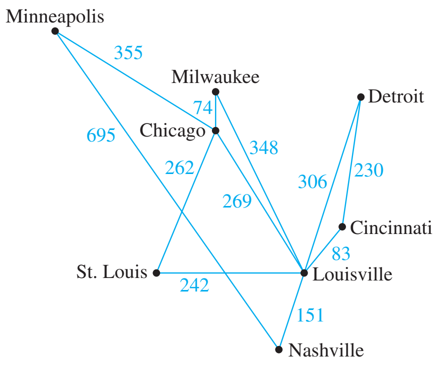


Figure 10.7.6

Solution

Iteration Number	Vertex Added	Edge Added	Weight
0	Minneapolis		
1	Chicago	Minneapolis–Chicago	355
2	Milwaukee	Chicago–Milwaukee	74
3	St. Louis	Chicago–St. Louis	262
4	Louisville	St. Louis–Louisville	242
5	Cincinnati	Louisville–Cincinnati	83
6	Nashville	Louisville–Nashville	151
7	Detroit	Cincinnati–Detroit	230

■

Note that the tree obtained is the same as that obtained by Kruskal's algorithm, but the edges are added in a different order.

As with Kruskal's algorithm, in order to ensure a unique output, the edges of the graph could be placed in an array and those with the same weight could be added in the order they appear in the array. It is not hard to see that when a connected graph is input to Prim's algorithm, the result is a spanning tree. What is not so clear is that this spanning tree is a minimum. The proof of the following theorem establishes that it is.

Theorem 10.7.3 Correctness of Prim's Algorithm

When a connected, weighted graph G is input to Prim's algorithm, the output is a minimum spanning tree for G.

Proof:

Let G be a connected, weighted graph, and suppose G is input to Prim's algorithm. At each stage of execution of the algorithm, an edge must be found that connects a vertex in a subgraph to a vertex outside the subgraph. As long as there are vertices outside the subgraph, the connectedness of G ensures that such an edge can always be found. *[For if one vertex in the subgraph and one vertex outside it are chosen, then by the connectedness of G there is a walk in G linking the two. As one travels along this walk, at some point one moves along an edge from a vertex inside the subgraph to a vertex outside the subgraph.]*

Now it is clear that the output T of Prim's algorithm is a tree because the edge and vertex added to T at each stage are connected to other edges and vertices of T

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

and because at no stage is a circuit created since each edge added connects vertices in two disconnected sets. *[Consequently, removal of a newly added edge produces a disconnected graph, whereas by Lemma 10.5.3, removal of an edge from a circuit produces a connected graph.]* Also, T includes every vertex of G because T, being a tree with $n-1$ edges, has n vertices *[and that is all G has]*. Thus T is a spanning tree for G.

Next we show that T has minimum weight. Let T_1 be a minimum spanning tree for G such that the number of edges T_1 and T have in common is a maximum. Suppose that $T \neq T_1$. Then there is an edge e in T that is not an edge of T_1. *[Since trees T and T_1 both have the same vertex set if they differ at all, they must have different, same-size edge sets.]* Of all such edges, let e be the last that was added when T was constructed using Prim's algorithm. Let S be the set of vertices of T just before the addition of e. Then one endpoint, say v of e, is in S and the other, say w, is not. Since T_1 is a spanning tree, there is a path in T_1 joining v to w. And since $v \in S$ and $w \notin S$, as one travels along this path, one must encounter an edge e' that joins a vertex in S to one that is not in S and that therefore is not in T because e was the last edge added to T. Now at the stage when e was added to T, e' could also have been added and it *would* have been added instead of e had its weight been less than that of e. Since e' was not added at that stage, we conclude that

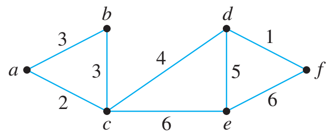
$$w(e') \geq w(e).$$

Let T_2 be the graph obtained from T_1 by removing e' and adding e. *[Thus T_2 has one more edge in common with T than T_1 does.]* Note that T_2 is a tree. The reason is that since e' is part of a path in T_1 from v to w, and e connects v and w, adding e to T_1 creates a circuit. When e' is removed from this circuit, the resulting subgraph remains connected. In fact, T_2 is a spanning tree for G since no vertices were removed in forming T_2 from T_1. The argument showing that $w(T_2) \leq w(T_1)$ is left as an exercise. *[It is virtually identical to part of the proof of Theorem 10.7.2.]* It follows that T_2 is a minimum spanning tree for G.

By construction, T_2 has one more edge in common with T than T_1, does which contradicts the choice of T_1 as a minimum spanning tree for G with a maximum number of edges in common with T. It follows that $T = T_1$, and hence T itself is a minimum spanning tree for G.

Example 10.7.4 Finding Minimum Spanning Trees

Find all minimum spanning trees for the following graph. Use Kruskal's algorithm and Prim's algorithm starting at vertex a. Indicate the order in which edges are added to form each tree.



Solution When Kruskal's algorithm is applied, edges are added in one of the following two orders:

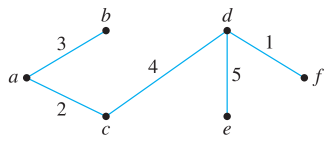
1. $\{d, f\}, \{a, c\}, \{a, b\}, \{c, d\}, \{d, e\}$
2. $\{d, f\}, \{a, c\}, \{b, c\}, \{c, d\}, \{d, e\}$

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

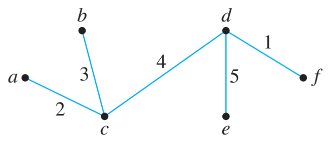
When Prim's algorithm is applied starting at a, edges are added in one of the following two orders:

1. $\{a, c\}, \{a, b\}, \{c, d\}, \{d, f\}, \{d, e\}$
2. $\{a, c\}, \{b, c\}, \{c, d\}, \{d, f\}, \{d, e\}$

Thus, as shown below, there are two distinct minimum spanning trees for this graph.



(a)



(b) ■

Dijkstra's Shortest Path Algorithm

Although the trees produced by Kruskal's and Prim's algorithms have the least possible total weight compared to all other spanning trees for the given graph, they do not always reveal the shortest distance between any two points on the graph. For instance, according to the complete route system shown in Figure 10.7.3, one can fly directly from Nashville to Minneapolis for a distance of 695 miles, whereas if you use the minimum spanning tree shown in Figure 10.7.5 the only way to fly from Nashville to Minneapolis is by going through Louisville, St. Louis, and Chicago, which gives a total distance of $151 + 242 + 262 + 355 = 1,010$ miles and the unpleasantness of three changes of plane.

In 1959 the computing pioneer, Edsgar Dijkstra (see Section 5.5), developed an algorithm to find the shortest path between a starting vertex and an ending vertex in a weighted graph in which all the weights are positive. It is somewhat similar to Prim's algorithm in that it works outward from a starting vertex a, adding vertices and edges one by one to construct a tree T. However, it differs from Prim's algorithm in the way it chooses the next vertex to add, ensuring that for each added vertex v, the length of the shortest path from a to v has been identified.

At the start of execution of the algorithm, each vertex u of G is given a label $L(u)$, which indicates the current best estimate of the length of the shortest path from a to u. $L(a)$ is initially set equal to 0 because the shortest path from a to a has length zero, but, because there is no previous information about the lengths of the shortest paths from a to any other vertices of G, the label $L(u)$ of each vertex u other than a is initially set equal to a number, denoted ∞, that is greater than the sum of the weights of all the edges of G. As execution of the algorithm progresses, the values of $L(u)$ are changed, eventually becoming the actual lengths of the shortest paths from a to u in G.

Because T is built up outward from a, at each stage of execution of the algorithm the only vertices that are candidates to join T are those that are adjacent to at least one vertex of T. Thus at each stage of Dijkstra's algorithm, the graph G can be thought of as divided into three parts: the tree T that is being built up, the set of "fringe" vertices that are adjacent to at least one vertex of the tree, and the rest of the vertices of G. Each fringe vertex is a candidate to be the next vertex added to T. The one that is chosen is the one for which the length of the shortest path to it from a through T is a minimum among all the vertices in the fringe.

An essential observation underlying Dijkstra's algorithm is that after each addition of a vertex v to T, the only fringe vertices for which a shorter path from a might be found are those that are adjacent to v *[because the length of the path from a to v was a minimum among all the paths from a to vertices in what was then the fringe].* So after each addition of a vertex v to T, each fringe vertex u adjacent to v is examined and two numbers are

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

compared: the current value of $L(u)$ and the value of $L(v) + w(v, u)$, where $L(v)$ is the length of the shortest path to v (in T) and $w(v, u)$ is the weight of the edge joining v and u. If $L(v) + w(v, u) < L(u)$, then the value of $L(u)$ is changed to $L(v) + w(v, u)$.

At the beginning of execution of the algorithm, the tree consists only of the vertex a, and $L(a) = 0$. When execution terminates, $L(z)$ is the length of a shortest path from a to z.

As with Kruskal's and Prim's algorithms for finding minimum spanning trees, there is a simple but dramatically inefficient way to find the shortest path from a to z: compute the lengths of all the paths and choose one that is shortest. The problem is that even for relatively small graphs using this method to find a shortest path could require billions of years, whereas Dijkstra's algorithm could do the job in a few seconds.

Algorithm 10.7.3 Dijkstra

Input: *G [a connected simple graph with a positive weight for every edge], ∞ [a number greater than the sum of the weights of all the edges in the graph], $w(u, v)$ [the weight of edge $\{u, v\}$], a [the starting vertex], z [the ending vertex]*

Algorithm Body:

```
1.  Initialize $T$ to be the graph with vertex $a$ and no edges. Let $V(T)$ be the set of
    vertices of $T$, and let $E(T)$ be the set of edges of $T$.

2.  Let $L(a) = 0$, and for all vertices in $G$ except $a$, let $L(u) = \infty$.
        [The number $L(x)$ is called the label of $x$.]

3.  Initialize $v$ to equal $a$ and $F$ to be $\{a\}$.
        [The symbol $v$ is used to denote the vertex most recently added to $T$.]

4.  while ($z \notin V(T)$)

    4a. $F := (F - \{v\}) \cup \{$vertices that are adjacent to $v$ and are not in $V(T)\}$
        [The set $F$ is called the fringe. Each time a vertex is added to $T$, it is removed
        from the fringe and the vertices adjacent to it are added to the fringe if they are
        not already in the fringe or the tree $T$.]
    4b. For each vertex $u$ that is adjacent to $v$ and is not in $V(T)$,
        if $L(v) + w(v, u) < L(u)$ then

                    $L(u) := L(v) + w(v, u)$

                    $D(u) := v$

        [Note that adding $v$ to $T$ does not affect the labels of any vertices in the fringe $F$
        except those adjacent to $v$. Also, when $L(u)$ is changed to a smaller value, the
        notation $D(u)$ is introduced to keep track of which vertex in $T$ gave rise to the
        smaller value.]
    4c. Find a vertex $x$ in $F$ with the smallest label
        Add vertex $x$ to $V(T)$, and add edge $\{D(x), x\}$ to $E(T)$
        $v := x$ [This statement sets up the notation for the next iteration of the loop.]

    end while
```

Output: $L(z)$ *[$L(z)$, a nonnegative integer, is the length of the shortest path from a to z.]*

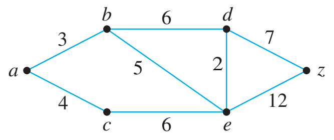
Note The unique path in the tree T from a to z is the shortest path in G from a to z.

The action of Dijkstra's algorithm is illustrated by the flow of the drawings in Example 10.7.5.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

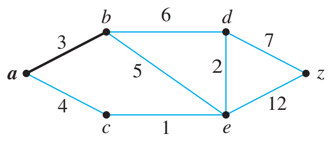
Example 10.7.5 Action of Dijkstra's Algorithm

Show the steps in the execution of Dijkstra's shortest path algorithm for the graph shown below with starting vertex a and ending vertex z.



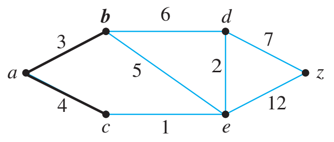
Solution

Step 1: Going into the **while** loop: $V(T) = \{a\}$, $E(T) = \emptyset$, and $F = \{a\}$



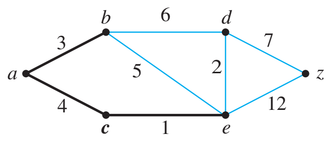
During iteration:
$F = \{b, c\}$, $L(b) = 3$, $L(c) = 4$.
Since $L(b) < L(c)$, b is added to $V(T)$ and $\{a, b\}$ is added to $E(T)$.

Step 2: Going into the **while** loop: $V(T) = \{a, b\}$, $E(T) = \{\{a, b\}\}$



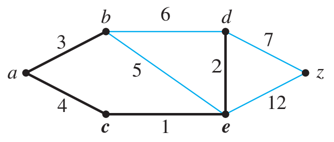
During iteration:
$F = \{c, d, e\}$, $L(c) = 4$, $L(d) = 9$, $L(e) = 8$.
Since $L(c) < L(d)$ and $L(c) < L(e)$, c is added to $V(T)$ and $\{a, c\}$ is added to $E(T)$.

Step 3: Going into the **while** loop: $V(T) = \{a, b, c\}$, $E(T) = \{\{a, b\}, \{a, c\}\}$



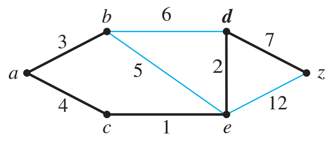
During iteration:
$F = \{d, e\}$, $L(d) = 9$, $L(e) = 5$
$L(e)$ becomes 5 because *ace*, which has length 5, is a shorter path to e than *abe*, which has length 8.
Since $L(e) < L(d)$, e is added to $V(T)$ and $\{c, e\}$ is added to $E(T)$.

Step 4: Going into the **while** loop: $V(T) = \{a, b, c, e\}$,
$E(T) = \{\{a, b\}, \{a, c\}, \{c, e\}\}$



During iteration:
$F = \{d, z\}$, $L(d) = 7$, $L(z) = 17$
$L(d)$ becomes 7 because *aced*, which has length 7, is a shorter path to d than *abd*, which has length 9.
Since $L(d) < L(z)$, d is added to $V(T)$ and $\{e, d\}$ is added to $E(T)$.

Step 5: Going into the **while** loop: $V(T) = \{a, b, c, e, d\}$,
$E(T) = \{\{a, b\}, \{a, c\}, \{c, e\}, \{e, d\}\}$



During iteration: $F = \{z\}$, $L(z) = 14$
$L(z)$ becomes 14 because *acedz*, which has length 14, is a shorter path to d than *abdz*, which has length 17.
Since z is the only vertex in F, its label is a minimum, and so z is added to $V(T)$ and $\{e, z\}$ is added to $E(T)$.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Execution of the algorithm terminates at this point because $z \in V(T)$. The shortest path from a to z has length $L(z) = 14$. ■

Keeping track of the steps in a table is a convenient way to show the action of Dijkstra's algorithm. Table 10.7.1 does this for the graph in Example 10.7.5.

Table 10.7.1

Step	$V(T)$	$E(T)$	F	$L(a)$	$L(b)$	$L(c)$	$L(d)$	$L(e)$	$L(z)$
0	$\{a\}$	$\emptyset$	$\{a\}$	**0**	∞	∞	∞	∞	∞
1	$\{a\}$	$\emptyset$	$\{b, c\}$	0	**3**	4	∞	∞	∞
2	$\{a, b\}$	$\{\{a, b\}\}$	$\{c, d, e\}$	0	3	**4**	9	8	∞
3	$\{a, b, c\}$	$\{\{a, b\}, \{a, c\}\}$	$\{d, e\}$	0	3	4	9	**5**	∞
4	$\{a, b, c, e\}$	$\{\{a, b\}, \{a, c\}, \{c, e\}\}$	$\{d, z\}$	0	3	4	**7**	5	17
5	$\{a, b, c, e, d\}$	$\{\{a, b\}, \{a, c\}, \{c, e\}, \{e, d\}\}$	$\{z\}$	0	3	4	7	5	**14**
6	$\{a, b, c, e, d, z\}$	$\{\{a, b\}, \{a, c\}, \{c, e\}, \{e, d\}, \{e, z\}\}$							

It is clear that Dijkstra's algorithm keeps adding vertices to I until it has added z. The proof of the following theorem shows that when the algorithm terminates, the label z goes the length of the shortest path to it from a.

Theorem 10.7.4 Correctness of Dijkstra's Algorithm

When a connected, simple graph with a positive weight for every edge is input to Dijkstra's algorithm with starting vertex a and ending vertex z, the output is the length of a shortest path from a to z.

Proof:

Let G be a connected, weighted graph with no loops or parallel edges and with a positive weight for every edge. Let T be the graph built up by Dijkstra's algorithm, and for each vertex u in G, let $L(u)$ be the label given by the algorithm to vertex u. For each integer $n \geq 0$, let the property $P(n)$ be the sentence

After the nth iteration of the while loop in Dijkstra's algorithm, (1) T is a tree, and (2) for every vertex v in T, $L(v)$ is the length of a shortest path in G from a to v. $\leftarrow P(n)$

We will show by mathematical induction that $P(n)$ is true for all integers n from 0 through the termination of the algorithm.

Show that P(0) is true: When $n = 0$, the graph T is a tree because it is defined to consist only of the vertex a and no edges. In addition, $L(a)$ is the length of the shortest path from a to a because the initial value of $L(a)$ is 0.

Show that for all integers k $\geq$ 0, if P(k) is true then P(k + 1) is also true: Let k be any integer with $k \geq 0$ and suppose that

After the kth iteration of the while loop in Dijkstra's algorithm, (1) T is a tree, and (2) for every vertex v in T, $L(v)$ is the length of a shortest path in G from a to v. $\leftarrow P(k)$ inductive hypothesis

We must show that

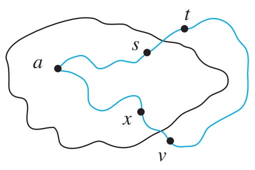
After the $(k + 1)$st iteration of the **while** loop in Dijkstra's algorithm, (1) T is a tree, and (2) for every vertex v in T, $L(v)$ is the length of a shortest path in G from a to v. $\leftarrow P(k + 1)$

continued on page 714

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

So suppose that after the $(k+1)$st iteration of the **while** loop in Dikjstra's algorithm, the vertex v and edge $\{x, v\}$ have been added to T, where x is in $V(T)$. Clearly the new value of T is a tree because adding a new vertex and edge to a tree does not create a circuit and does not disconnect the tree. By inductive hypothesis for each vertex y in the tree before the addition of v, $L(y)$ is the length of a shortest path from a to y. So it remains only to show that $L(v)$ is the length of a shortest path from a to v.

Now, according to the algorithm, the final value of $L(v) = L(x) + w(x, v)$. Consider *any* shortest path from a to v, and let $\{s, t\}$ be the first edge in this path to leave T, where $s \in V(T)$ and $t \notin V(T)$. This situation is illustrated below.



Let $LSP(a, v)$ be the length of a shortest path from a to v, and let $LSP(a, s)$ be the length of a shortest path from a to s. Observe that

$$\begin{aligned} LSP(a, v) &\geq LSP(a, s) + w(s, t) && \text{because the path from } t \text{ to } v \text{ has length} \geq 0 \\ &\geq L(s) + w(s, t) && \text{by inductive hypothesis because } s \text{ is a vertex in } T \\ &\geq L(x) + w(x, v) && t \text{ is in the fringe of the tree, and so if } L(s)+w(s,t) \text{ were less than } L(x)+w(x,v) \text{ then } t \text{ would have been added to } T \text{ instead of } v. \end{aligned}$$

On the other hand

$$L(x) + w(x, v) \geq LSP(a, v) \quad \text{because } L(x)+w(x,v) \text{ is the length of } a \text{ path from } a \text{ to } v \text{ and so it is greater than or equal to the length of the shortest path from } a \text{ to } v.$$

It follows that $$LSP(a, v) = L(x) + w(x, v),$$

and, since $$L(v) = L(x) + w(x, v),$$

$L(v)$ is the length of a shortest path from a to v. This completes the proof by mathematical induction.

The algorithm terminates as soon as z is in T, and, since we have proved that the label of every vertex in the tree gives the length of the shortest path to it from a, then, in particular, $L(z)$ is the length of a shortest path from a to z.

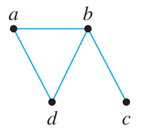
Test Yourself

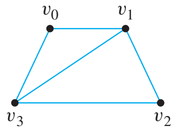
1. A spanning tree for a graph G is _____.
2. A weighted graph is a graph for which _____, and the total weight of the graph is _____.
3. A minimum spanning tree for a connected, weighted graph is _____.
4. In Kruskal's algorithm, the edges of a connected, weighted graph are examined one by one in order of _____ starting with _____.
5. In Prim's algorithm, a minimum spanning tree is built by expanding outward from an _____ in a sequence of _____.
6. In Dijkstra's algorithm, a vertex is in the fringe if it is _____ vertex in the tree that is being built up.
7. At each stage of Dijkstra's algorithm, the vertex that is added to the tree is a vertex in the fringe whose label is a _____.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

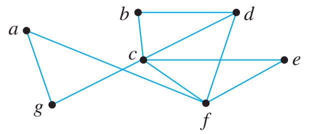
Exercise Set 10.7

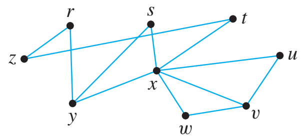
Find all possible spanning trees for each of the graphs in 1 and 2.

1. 

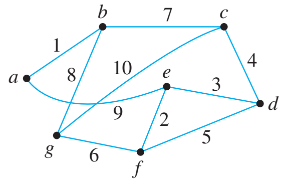
2. 

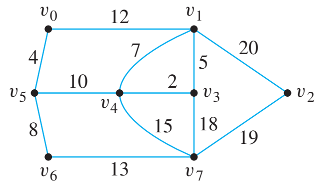
Find a spanning tree for each of the graphs in 3 and 4.

3. 

4. 

Use Kruskal's algorithm to find a minimum spanning tree for each of the graphs in 5 and 6. Indicate the order in which edges are added to form each tree.

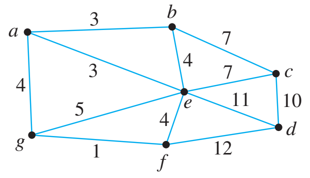
5. 

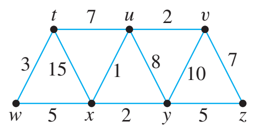
6. 

Use Prim's algorithm starting with vertex a or v_0 to find a minimum spanning tree for each of the graphs in 7 and 8. Indicate the order in which edges are added to form each tree.

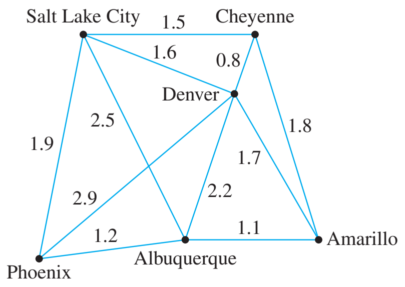
7. The graph of exercise 5. 8. The graph of exercise 6.

For each of the graphs in 9 and 10, find all minimum spanning trees that can be obtained using (a) Kruskal's algorithm and (b) Prim's algorithm starting with vertex a or t. Indicate the order in which edges are added to form each tree.

9. 

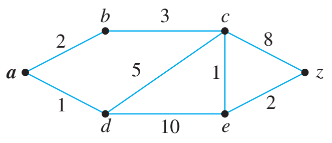
10. 

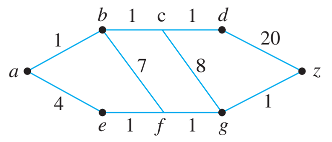
11. A pipeline is to be built that will link six cities. The cost (in hundreds of millions of dollars) of constructing each potential link depends on distance and terrain and is shown in the weighted graph below. Find a system of pipelines to connect all the cities and yet minimize the total cost.



12. Use Dijkstra's algorithm for the airline route system of Figure 10.7.3 to find the shortest distance from Nashville to Minneapolis. Make a table similar to Table 10.7.1 to show the action of the algorithm.

Use Dijkstra's algorithm to find the shortest path from a to z for each of the graphs in 13–16. In each case make tables similar to Table 10.7.1 to show the action of the algorithm.

13. 

14. 

15. The graph of exercise 9 with $a = a$ and $z = f$

16. The graph of exercise 10 with $a = u$ and $z = w$

17. Prove part (2) of Proposition 10.7.1: Any two spanning trees for a graph have the same number of edges.

18. Given any two distinct vertices of a tree, there exists a unique path from one to the other.

a. Give an informal justification for the above statement.

✶ **b.** Write a formal proof of the above statement.

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

19. Prove that if G is a graph with spanning tree T and e is an edge of G that is not in T, then the graph obtained by adding e to T contains one and only one set of edges that forms a circuit.

20. Suppose G is a connected graph and T is a circuit-free subgraph of G. Suppose also that if any edge e of G not in T is added to T, the resulting graph contains a circuit. Prove that T is a spanning tree for G.

21. **a.** Suppose T_1 and T_2 are two different spanning trees for a graph G. Must T_1 and T_2 have an edge in common? Prove or give a counterexample.
    b. Suppose that the graph G in part (a) is simple. Must T_1 and T_2 have an edge in common? Prove or give a counterexample.

H **22.** Prove that an edge e is contained in every spanning tree for a connected graph G if, and only if, removal of e disconnects G.

23. Consider the spanning trees T_1 and T_2 in the proof of Theorem 10.7.3. Prove that $w(T_2) \le w(T_1)$.

24. Suppose that T is a minimum spanning tree for a connected, weighted graph G and that G contains an edge e (not a loop) that is not in T. Let v and w be the endpoints of e. By exercise 18 there is a unique path in T from v to w. Let e' be any edge of this path. Prove that $w(e') \le w(e)$.

H **25.** Prove that if G is a connected, weighted graph and e is an edge of G (not a loop) that has smaller weight than any other edge of G, then e is in every minimum spanning tree for G.

✶ **26.** If G is a connected, weighted graph and no two edges of G have the same weight, does there exist a unique minimum spanning tree for G? Use the result of exercise 19 to help justify your answer.

✶ 27. Prove that if G is a connected, weighted graph and e is an edge of G that (1) has greater weight than any other edge of G and (2) is in a circuit of G, then there is no minimum spanning tree T for G such that e is in T.

28. Suppose a disconnected graph is input to Kruskal's algorithm. What will be the output?

29. Suppose a disconnected graph is input to Prim's algorithm. What will be the output?

30. Prove that if a connected, weighted graph G is input to Algorithm 10.7.4 (shown below), the output is a minimum spanning tree for G.

Algorithm 10.7.4

```
Input: G [a connected graph]
Algorithm Body:
1. T := G.
2. E := the set of all edges of G, m := the number of edges
   of G.
3. while (m > 0)
    3a. Find an edge e in E that has maximal weight.
    3b. Remove e from E and set m := m − 1.
    3c. if the subgraph obtained when e is removed from
          the edge set of T is connected then remove e from
          the edge set of T
    end while
Output: T [a minimum spanning tree for G]
```

31. Modify Algorithm 10.7.3 so that the output consists of the sequence of edges in the shortest path from a to z.

Answers for Test Yourself

1. a subgraph of G that contains every vertex of G and is a tree. 2. each edge has an associated positive real number weight; the sum of the weights of all the edges of the graph 3. a spanning tree that has the least possible total weight compared to all other spanning trees for the graph 4. weight; an edge of least weight 5. initial vertex; adjacent vertices and edges 6. adjacent to a 7. minimum among all those in the fringe

Copyright 2010 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.