Stein et al.

6 Graphs

6.1 GRAPHS

In this chapter, we study graphs, which are a fundamental topic in discrete mathematics and computer science. As we will see, we can use graphs to model many common situations and to naturally describe many algorithms. Graphs are also an ideal venue for developing a deeper understanding of proof by induction, especially strong induction.

Exercise 6.1-1 Figure 6.1 shows a stylized map of some cities in the eastern United States (Boston, New York, Pittsburgh, Cincinnati, Chicago, Memphis, New Orleans, Atlanta, Washington, DC, and Miami). A company has major offices with data-processing centers in each of these cities, and as its operations have grown, it has leased dedicated communication lines between certain pairs of these cities to allow for efficient communication among the computer systems. Each blue dot in the figure stands for a data center, and each line stands for a dedicated communication link. What is the minimum number of links that could be used to send a message from B (Boston) to NO (New Orleans)? Give a route with this number of links.

Exercise 6.1-2 Which city (or cities) has (or have) the most communication links emanating from it (or them)?

Exercise 6.1-3 What is the total number of communication links in the figure?

Figure 6.1 is a drawing of what we call a "graph." A **graph** consists of a set of **vertices**[1] and a set of **edges**[2] and has the property that each edge has two

[1] Another common name for vertex is *node*.

[2] Another common name for edge is *arc*, though some authors restrict this usage to directed graphs.

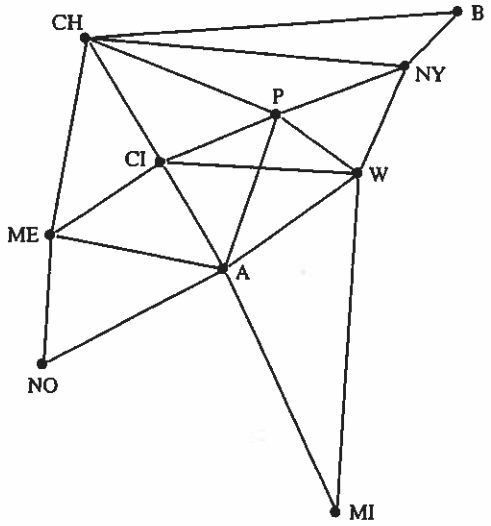


Figure 6.1: A stylized map of some eastern U.S. cities

(not necessarily different) vertices, called its **endpoints**, associatec
We say that the edge **joins** the endpoints, and we say that two e
are **adjacent** if they are joined by an edge. When a vertex is an
of an edge, we say that the edge and the vertex are **incident**. Seve
examples of graphs are given in Figure 6.2. Graphs model situ
which there are relationships among pairs of objects. In Figure
objects are the cities, and the relationship is being joined by a com
tion link. More generally, we represent objects as vertices, and we r
a relationship between two objects as an edge connecting their
Other examples include a graph in which the vertices represent b
species and two vertices are joined by an edge if their species

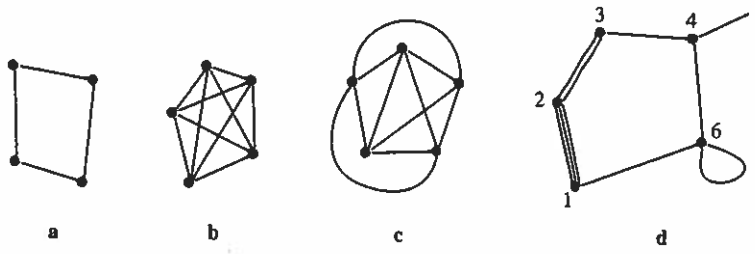


Figure 6.2: Some examples of graphs

common ancestor, or a graph in which the vertices represent people and an edge is drawn between two vertices if the people attended the same school.

The relationships we have mentioned are all symmetric; that is, whatever relationship may exist between two vertices a and b also exists between b and a. Graphs model symmetric relationships. One can also study directed graphs, which model relationships that are not necessarily symmetric. Much of what we do for graphs holds for directed graphs as well; we do not pursue the idea of directed graphs in this book.

To **draw** a graph, we draw a point (in our case, a blue circle) in the plane for each vertex; then, for each edge, we draw a (possibly curved) line between the vertices that corresponds to the endpoints of the edge. The only vertices that may be touched by the line representing an edge are the endpoints of that edge. Notice that Figure 6.2d has three edges joining vertices 1 and 2, two edges joining vertices 2 and 3, and one edge joining vertex 6 to itself. This last edge has two identical endpoints. Note that in Figure 6.2, sometimes the vertices are labeled, and sometimes they aren't. We label the vertices to give them meaning, as in Figure 6.1, or when we know we will refer to them, as in Figure 6.2d.

Figure 6.1 and the first three graphs in Figure 6.2 are examples of **simple graphs**, which are graphs that have at most one edge joining each pair of distinct vertices and no edges joining a vertex to itself.[3] If there is an edge joining vertex x and vertex y in a simple graph, we denote it by $\{x, y\}$. Thus, in Figure 6.1, $\{P, W\}$ denotes the edge between Pittsburgh and Washington, DC. Sometimes it will be helpful to have a symbol to stand for a graph. The phrase "Let $G = (V, E)$" is shorthand for "Let G stand for a graph with vertex set V and edge set E." We say that Figure 6.2d has a loop at vertex 6 and multiple edges joining vertices 1 and 2 and vertices 2 and 3. More precisely, an edge that joins a vertex to itself is called a **loop**, and if there is more than one edge joining x and y, then the graph is said to have **multiple edges** between those two vertices.

Figures 6.2b and 6.2c are different drawings of the same graph, which consists of five vertices and one edge between each pair of distinct vertices. It is called the complete graph on five vertices and is denoted by K_5. In general, a **complete graph** on n vertices is a graph with n vertices that has an edge between each pair of vertices. We use K_n to stand for a complete

[3]The terminology of graph theory has not yet been standardized perhaps because it is a relatively young subject. The terminology we are using here is the most popular terminology in computer science. However, some graph theorists would reserve the word *graph* for what we have just called a *simple graph* and would use the word *multigraph* for what we have called a *graph*.

graph on n vertices. Figures 6.2b and 6.2c illustrate that there are different ways of drawing a given graph. The two drawings also d strate two different ideas: Figure 6.2b shows that each vertex is ac to each other vertex and suggests that there is a high degree of sym Figure 6.2c shows that it is possible to draw the graph so that on pair of edges crosses; other than the one place where two edges cro only places where edges touch each other are at their endpoints. In is impossible to draw K_5 so that no edges cross, a fact that we explai in this chapter.

In Exercise 6.1-1, the links referred to are edges of the graph, and the are the vertices of the graph. It is possible to get from the vertex for I to the vertex for New Orleans by using three communication links, n the edge from Boston to Chicago, the edge from Chicago to Memph the edge from Memphis to New Orleans. A **path** in a graph is an alter sequence of vertices and edges such that

- it starts and ends with a vertex,
- each edge joins the vertex before it in the sequence to the vert after it in the sequence, and
- no vertex appears more than once in the sequence.

If a is the first vertex in the path and b is the last vertex in the path, tl say the path is from a to b. Thus, the path from Boston to New Orl

B{B, CH}CH{CH, ME}ME{M E, NO}NO.

Because the graph is simple, there is exactly one edge between suc vertices in this list. Therefore, we can also use the shorter n B, CH, ME, NO to describe the same path. The **length of a path** number of edges it has; so our path from Boston to New Orlea length 3. By inspecting the map, we see that there is no shorter pat Boston to New Orleans. The length of a shortest path between two v in a graph is called the **distance** between them. Thus, the distanc Boston to New Orleans in the graph of Figure 6.1 is three.

Some applications lead us to pathlike sequences in which vertices repeated. A **walk** satisfies the first two conditions for a path but ne satisfy the third.[4] The **length of a walk** is the number of edges it h

The following lemma will prove useful later.

[4]Some texts use the word "path" for what we have just defined as a walk and use phrase "simple path" for what we have defined as a path.

Lemma 6.1 If there is a walk between two distinct vertices x and y of a graph G, then there is a path between x and y in G.

Proof If the walk is a path, then we are done. If not, let z be a vertex that appears more than once in the walk from x to y. We create a shorter walk by removing the part of the walk between the first and last occurrences of z in the walk, including the last z but not the first. Then z will appear only once in the new walk. This process can be repeated until there are no vertices that appear more than once. At that point, the walk is a path.

The Degree of a Vertex

In Exercise 6.1-2, the city with the most communication links is Atlanta (A). We say the vertex A has degree 6, because six edges are incident to it. More generally, the **degree** of a vertex in a graph is the number of times it is incident with edges of the graph; that is, the degree of a vertex x is the number of edges between x and other vertices plus twice the number of loops at vertex x. In Figure 6.2d, vertex 2 has degree 5, and vertex 6 has degree 4.

Exercise 6.1-4 In a graph like the one in Figure 6.1, it is somewhat difficult to count the edges, because you might forget which ones you've counted and which ones you haven't. Is there a relationship between the number of edges in a graph and the degrees of the vertices? If so, find it. (*Hint:* Computing degrees of vertices and number of edges in some relatively small examples of graphs should help you discover a formula.)

In Exercise 6.1-4, examples such as those in Figure 6.2 convince us that the sum of the degrees of the vertices is twice the number of edges. How can we prove this? One way is to count the total number of incidences between vertices and edges. Each edge has exactly two incidences, so the total number of incidences is twice the number of edges. But the degree of a vertex is the number of incidences it has, so the sum of the degrees of the vertices is also the total number of incidences. Therefore, the sum of the degrees of the vertices of a graph is twice the number of edges. Thus, to compute the number of edges of a graph, we can sum the degrees of the vertices and divide by 2. There is another proof of this result that uses induction.

Theorem 6.2 Suppose a graph has a finite number of edges. Then the sum of the degrees of the vertices is twice the number of edges.

Proof The proof proceeds by induction on the number of edge: graph. If a graph has no edges, then each vertex has degree 0 and t of the degrees is 0, which is twice the number of edges. Now supp $e > 0$ and that the theorem is true whenever a graph has fewer than Let G be a graph with e edges and let ϵ be an edge of G.[5] Let G graph (on the same vertex set as G) that we get by deleting ϵ from t set E of G. Then G' has $e - 1$ edges, and so, by our inductive hyp the sum of the degrees of the vertices of G' is $2(e - 1)$. Now there possible cases. Either e was a loop, in which case one vertex of degree two less in G' than it has in G, or e has two distinct endp which case exactly two vertices of G' have degree one less than thei in G. Thus, in both cases, the sum of the degrees of the vertices in G less than the sum of the degrees of the vertices in G. Therefore, the the degrees of the vertices in G is $(2e - 2) + 2 = 2e$. Thus, the trut theorem for graphs with $e - 1$ edges implies the truth of the theo graphs with e edges. Therefore, by the principle of mathematical in the theorem is true for a graph with any finite number of edges.

There are several instructive points in the proof of Theorem 6.: because it wasn't clear from the outset whether we would need strong or weak induction, we made the inductive hypothesis that w normally make for strong induction. However, in the course of th we saw that we only needed to use weak induction; so we wrote our sion accordingly. This is not a mistake. We used our inductive hy correctly; we just didn't need to use it for every possible value it c

Second, instead of saying that we would take a graph with $e - 1$ ed add an edge to get a graph with e edges, we said that we woul graph with e edges and remove an edge to get a graph with $e -$ We proceeded in this manner because we need to prove that the resu for *every* graph with e edges. By using the second approach, we the need to say that "every graph with e edges may be built up from with $e - 1$ edges by adding an edge," because in the second appro started with an arbitrary graph on e edges. In the first approach, w have proved that the theorem was true for all graphs that could from an $(e - 1)$-edge graph by adding an edge, and we would hav say explicitly that every graph with e edges could be built in this

[5] Because it is very handy to have e stand for the number of edges of a graph, we Greek letters such as epsilon (ϵ) to stand for the edges of a graph. It is also handy to stand for the number of vertices of a graph, so we use other letters near the en alphabet, such as w, x, y, and z, to stand for vertices.

In Exercise 6.1-3, the sum of the degrees of the vertices (working from left to right) is

$$2+4+5+5+6+5+2+5+4+2=40\ ,$$

and so the graph has 20 edges.

Connectivity

All of the examples we have seen so far have a property that is not common to all graphs—namely, that for every pair of vertices, there is a path between them.

Exercise 6.1-5 The company with the computer network in Figure 6.1 needs to reduce its expenses. It is currently leasing each of the communication lines shown in the figure. Because it can send information from one city to another through one or more intermediate cities in the graph, it decides to lease only the minimum number of communication lines it needs to be able to send a message from any city to any other city by using any number of intermediate cities. What is the minimum number of lines it needs to lease? Give two examples of subsets of the edge set with this number of edges (lines) that will allow communication between any two cities. Then give two examples of a subset of the edge set with this number of edges (lines) that will not allow communication between any two cities.

Some experimentation with the graph in Figure 6.1 convinces us that if we keep eight or fewer edges, there is no way to communicate among the cities (we explain this more precisely later on). However, we also see that there are quite a few sets of nine edges that suffice for communication among all the cities. Figure 6.3 shows two sets of nine edges each that allow communication among all the cities and two sets of nine edges each that do not allow communication among all the cities.

Notice that in Figures 6.3a and 6.3b, it is possible to get from any vertex to any other vertex by a path. A graph is called **connected** when, for each pair of vertices of the graph, there is a path between these two vertices. Notice that in Figure 6.3c, it is not possible to find a path from Atlanta to Boston, for example, and in Figure 6.3d, it is not possible to find a path from Miami to any of the other vertices. Thus, these last two graphs are not connected, and we call them **disconnected**. In Figure 6.3d, we say that Miami is an **isolated vertex**. We say two vertices are **connected** if there is a path between them. Thus, in Figure 6.3c, the vertices for Boston and New Orleans are connected.

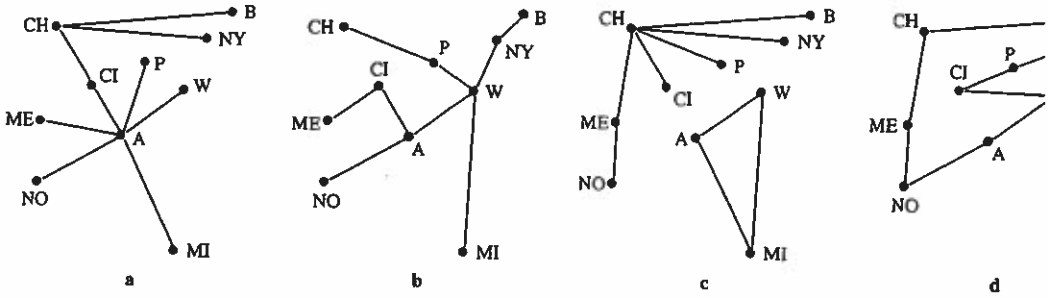


Figure 6.3: Selecting nine edges from the stylized map of some eastern cities

The relationship of being connected is an equivalence relation that the set of vertices into mutually exclusive classes; that is, it partit vertices of the graph. How do we know this? Connectivity is re symmetric, and transitive. A vertex x is connected to itself, so conr is reflexive. If there is a path from x to y then reversing direction path from y to x, showing that connectivity is symmetric. If there i from x to y and a path from y to z then we get a walk from x appending the paths. By Lemma 6.1 there must be a path from x connectivity is transitive.

We call the relationship of "being connected to" the **connectivity r** We call the blocks into which this relationship partitions the graph **tivity classes**. There is no edge of a graph between two vertices in d connectivity classes, because if there were, then everything in or would be connected to everything in the other class and the two would have to be the same. Thus, we also end up with a partitior edges into disjoint sets. If a graph has edge set E, and C is a conn class, then $E(C)$ denotes the set of edges whose endpoints are bot Because no edge connects vertices in different connectivity classe edge must be in some set $E(C)$. The graph consisting of a conn class C, together with the edges $E(C)$, is called a **connected com** of our original graph. From now on, our emphasis is on connecte ponents rather than on connectivity classes, and we describe a co component by listing its vertices. Note that Figures 6.3c and 6.3 have two connected components. In Figure 6.3c, the vertex sets of t nected components are {NO, ME, CH, CI, P, NY, B} and {A, W, Figure 6.3d, the connected components are {NO, ME, CH, B, NY, W, A} and {MI}. Two other examples of graphs with multiple co components are shown in Figure 6.4.



Figure 6.4: A simple graph G_1 with three connected components and a graph G_2 with four connected components

Cycles

In Figures 6.3c and 6.3d, we see a feature that we don't see in Figures 6.3a and 6.3b, namely, a walk that leads from a vertex back to itself. A walk with at least one edge that starts and ends at the same vertex, but that has no other repeated vertices or edges, is called a **cycle**. Similarly, a walk that starts and ends with the same vertex is called a **closed walk**. The closed walks in Figures 6.3c and 6.3d are cycles A, W, M, A and NO, ME, CH, B, NY, P, CI, W, A, NO, respectively. We don't normally distinguish which point on a cycle is the starting point; for example, we consider the cycle A, W, MI, A to be the same as the cycle W, MI, A, W.

Let's compare Figures 6.3d and 6.1. In both graphs, NO, ME, CH, B, NY, P, CI, W, A, NO is a cycle. In Figure 6.3d, the only edges on the set of vertices in the cycle are the edges of the cycle. In contrast, some vertices in the cycle of Figure 6.1 are joined by other edges, too. We wish to be able to distinguish between these two cases.

In general, a graph H is called a **subgraph** of the graph G if all the vertices and edges of H are vertices and edges of G. In other words, $H = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. A graph H is called an **induced subgraph** of G if H is a subgraph of G and every edge of G connecting vertices of H is an edge of H. Thus, the graph G_1 in Figure 6.4 has an induced K_4 (complete graph on four vertices) and an induced cycle on three vertices (which also happens to be an induced K_3). It has a subgraph that is a cycle on four vertices, but it does not have an induced subgraph that is a cycle on four vertices. It has some induced paths on three vertices as well. Can you find one?

Notice that in graph G_2 of Figure 6.4, there are cycles with one edge and cycles with two edges. We call a graph a **cycle on n vertices**, or an **n-cycle**, and denote it by C_n if its vertex set is the vertex set of a cycle and its edge set is the edge set of that cycle. We say that a graph is a **path on n vertices** and denote it by P_n if its vertex set is the vertex set of a path and its edge set is the edge set of that path. Thus, Figure 6.2a is a drawing of C_4. Graph G_2 of Figure 6.4 has an induced P_3 and an induced C_2 as subgraphs.

Trees

The graphs in Figures 6.3a and 6.3b are called trees. We have re them slightly in Figure 6.5 to clarify why they are called trees. No the graphs drawn in Figures 6.3a and 6.3b and in Figures 6.5a and 6. connected and have no cycles.

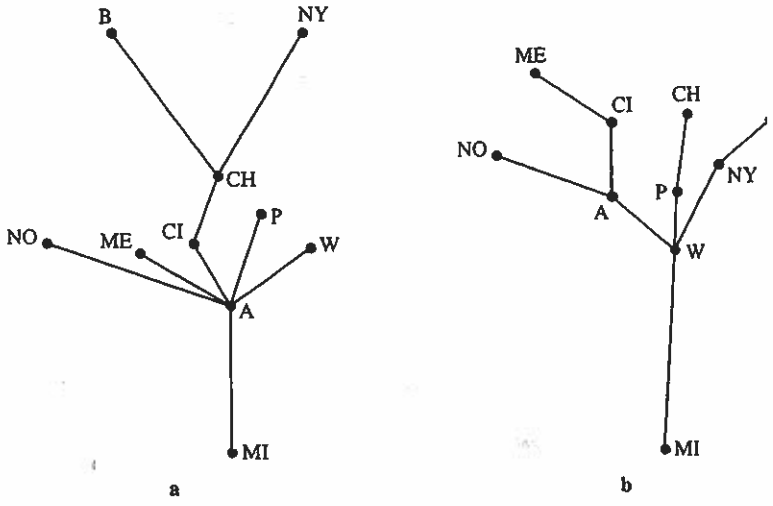


Figure 6.5: A visual explanation of the name tree

Definition 6.1

A connected graph with no cycles is called a tree.[6]

Note that the graph with one vertex and no edges is, by this defini tree.

Other Properties of Trees

Our definition of a tree left out several other properties of trees t could have discovered by a further analysis of Figure 6.3.

Exercise 6.1-6 Given any two vertices in a tree, how many distinct paths are there b these two vertices?

[6]The student who has experience with rooted trees, binary trees, or binary search t should note that we are not talking about these kinds of trees in this section. They subject of the next section.

Exercise 6.1-7 Is it possible to delete an edge from a tree and have it remain connected?

Exercise 6.1-8 If $G = (V, E)$ is a graph, and we add an edge that joins vertices of V, what can happen to the number of connected components?

Exercise 6.1-9 How many edges does a tree with v vertices have?

Exercise 6.1-10 Does every tree have a vertex of degree 1? If the answer is "yes," explain why. If the answer is "no," try to find additional conditions that will guarantee that a tree satisfying these conditions has a vertex of degree 1.

For Exercise 6.1-6, suppose we have two distinct paths from a vertex x to a vertex y. The paths begin with the same vertex x and might have some more edges in common, as in Figure 6.6. Let w be the last vertex after (or including) x that the paths share before they become different. For visualizing the argument, let us focus on the path that goes upward at the vertices marked w and t and the path that goes downward at these two vertices. The paths must come together again at y, though they might come together earlier. Let z be the first vertex the paths have in common after w. Then there are two paths from w to z that have only w and z in common. Taking one of these paths from w to z and the other from z to w gives us a cycle, and so the graph is not a tree. We have shown that if a graph has two distinct paths from x to y, then it is not a tree. By contrapositive inference, then, if a graph is a tree, it does not have two distinct paths between two vertices x and y. We state this result as a theorem.

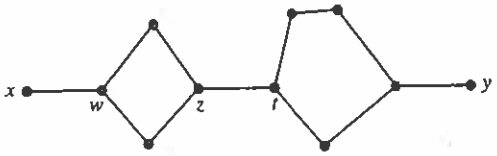


Figure 6.6: A graph with multiple paths from x to y

Theorem 6.3 There is exactly one path between each pair of vertices in a tree.

Proof By the definition of a tree, there is at least one path between each pair of vertices. By our argument above, there is at most one path between each pair of vertices. Thus, there is exactly one path.

For Exercise 6.1-7, note that if ϵ is an edge from x to y, then x, ϵ, unique path from x to y in the tree. Suppose we delete ϵ from the ed the tree. If there were still a path from x to y in the resulting grapl would also be a path from x to y in the tree, which would contradict 6.3. Thus, the only possibility is that there is no path between x and resulting graph; thus, it is not connected and is therefore not a tree.

For Exercise 6.1-8, if the endpoints are in the same connected con then the number of connected components will not change. If the end the edge are in different connected components, then the number of cc components can decrease by one. Because an edge has two endpoi impossible for the number of connected components to decrease by m one when we add an edge. This paragraph and the previous one lead following useful lemma.

Lemma 6.4 Removing one edge from the edge set of a tree gives a graph w connected components, each of which is a tree.

Proof Suppose that ϵ is an edge from x to y in a tree. We have see graph that we get by deleting ϵ from the edge set of the tree is not cc so the graph has at least two connected components. But adding the e in can only reduce the number of connected components by one. T the graph has exactly two connected components. Because neither cycles, both are trees.

In Exercise 6.1-9, our trees with ten vertices had nine edges. If we dr on two vertices, it will have one edge; if we draw a tree on three ve will have two edges. There are two different-looking trees on four ve shown in Figure 6.7, and each has three edges. On the basis of these e we conjecture that a tree on n vertices has $n - 1$ edges. One app proving this is to try to use induction. To do so, we have to see how every tree from smaller trees or how to take a tree and break it int ones. In either case, we then have to figure out how to use the tru conjecture for the smaller trees to imply its truth for the larger trees. A that people often make at this stage is to assume that every tree can from smaller ones by adding a vertex of degree 1. Although that is finite trees with more than one vertex (which is the point of Exercise



Figure 6.7: Two trees on four vertices

we haven't proved it yet, so we can't yet use it in proofs of other theorems. Another approach to using induction is to ask whether there is a natural way to break a tree into two smaller trees. There is a way; as we just showed in Lemma 6.4, if you remove an edge ϵ from the edge set of a tree, you get two connected components that are trees. We may assume inductively that the number of edges of each of these trees is one less than its number of vertices. Thus, if the graph with these two connected components has v vertices, then it has $v - 2$ edges. Adding ϵ back in gives us a graph with $v - 1$ edges—so, except for the fact that we have not done a base case, we have proved the following theorem.[7]

Theorem 6.5 For all integers $v \geq 1$, a tree with v vertices has $v - 1$ edges.

Proof If a tree has one vertex, it can have no edges, as any edge would have to connect that vertex to itself and would thus give a cycle. A tree with two or more vertices must have an edge in order to be connected. Before the statement of the theorem, we showed how to use the deletion of an edge to complete an inductive proof that a tree with v vertices has $v - 1$ edges. Therefore, for all $v \geq 1$, a tree with v vertices has $v - 1$ edges.

Finally, for Exercise 6.1-10, we can now give a contrapositive argument to show that a finite tree with more than one vertex has a vertex of degree 1. Suppose that G is a graph that is connected and that all vertices of G have degree 2 or more. Then the sum of the degrees of the vertices is at least $2v$, and so, by Theorem 6.2, the number of edges is at least v. Therefore, by Theorem 6.5, G is not a tree. Then, by contrapositive inference, if T is a tree, then T must have at least one vertex of degree 1. This corollary to Theorem 6.5 is so useful that we state it formally.

Corollary 6.6 A finite tree with more than one vertex has at least one vertex of degree 1.

Important Concepts, Formulas, and Theorems

1. *Graph.* A *graph* consists of a set of *vertices* and a set of *edges* and has the property that each edge has two (not necessarily different) vertices, called its *endpoints*, associated with it.

[7] In Section 4.1, we mentioned that in certain applications of induction, it makes our proofs simpler if we try to understand how to break large instances of our problems into smaller ones, rather than trying to understand how to build smaller instances to get larger ones. This is one example where the approach is useful.

2. *Edge/adjacent.* We say that an edge in a graph *joins* its endpoi we say that two endpoints are *adjacent* if they are joined by an
3. *Incident.* When a vertex is an endpoint of an edge, we say that and the vertex are *incident.*
4. *Drawing a graph.* To *draw* a graph, we draw a point in the pla each vertex. For each edge, we draw a (possibly curved) line b the points that correspond to the endpoints of the edge. Lines t correspond to edges may only touch the vertices that are their endpoints.
5. *Simple graph.* A *simple graph* is one that has, at most, one edge each pair of distinct vertices and no edges joining a vertex to it
6. *Loop/multiple edges.* An edge that joins a vertex to itself is call *loop,* and we say that we have *multiple edges* between vertices if there is more than one edge joining x and y.
7. *Notation for a graph.* The phrase "Let $G = (V, E)$" is shorthan "Let G stand for a graph with vertex set V and edge set E."
8. *Notation for edges.* In a simple graph, we use the notation $\{x, y$ edge from x to y. In any graph, when we want to use a letter to an edge, we use a Greek letter like ϵ so that we can save e to st the number of edges of the graph.
9. *Complete graph on* n *vertices.* A *complete graph* on n vertices graph with n vertices that has an edge between each pair of ver We use K_n to stand for a complete graph on n vertices.
10. *Walk.* We call an alternating sequence of vertices and edges in a a *walk* if it starts and ends with a vertex and each edge joins the before it (in the sequence) to the vertex after it (in the sequence
11. *Path.* A walk is called a *path* if it has no repeated vertices or ed
12. *Length/distance.* The *length* of a path is the number of edges. T *distance* between two vertices in a graph is the length of a short between them.
13. *Degree of a vertex.* The *degree* of a vertex in a graph is the num times it is incident with edges of the graph; that is, the degree o vertex x is the number of edges from x to other vertices plus tw number of loops at vertex x.
14. *Sum of degrees of vertices.* The sum of the degrees of the vertic graph with a finite number of edges is twice the number of edge
15. *Connected.* A graph is *connected* if, for each pair of vertices of graph, there is a path between them. We say that two vertices ar *connected* if there is a path between them; so, a graph is connec

each pair of its vertices are connected. The relationship of being connected is an equivalence relation that partitions the vertices of a graph into sets called *connectivity classes*.

16. *Connected component.* If C is a subset of the vertex set of a graph, then we use $E(C)$ to stand for the set of all edges *both* of whose endpoints are in C. The graph consisting of a connectivity class C of the connectivity relation, together with the edges $E(C)$, is called a *connected component* of our original graph.
17. *Closed walk.* A walk that starts and ends at the same vertex is called a *closed walk*.
18. *Cycle.* A walk whose first and last vertices are the same is called a *cycle* if it has at least one edge and all vertices of the walk, except the first and last, are distinct.
19. *Tree.* A connected graph with no cycles is called a tree.
20. *Important properties of trees.*
    a. There is a unique path between each pair of vertices in a tree.
    b. A tree on v vertices has $v - 1$ edges.
    c. Every finite tree with at least two vertices has a vertex of degree 1.

Problems

All problems with blue boxes have an answer or hint available at the end of the book.

1. Find a shortest path from vertex 1 to vertex 5 in Figure 6.8.

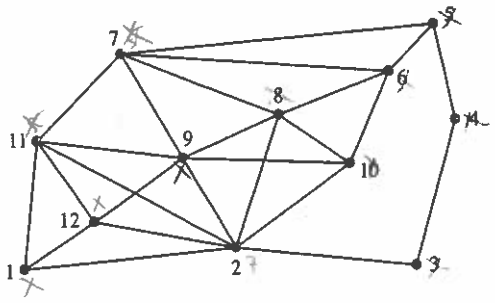


Figure 6.8

2. Find the longest path possible from vertex 1 to vertex 5 in Figure 6.8.

3. Find the vertex of largest degree in Figure 6.8. What is its degree?

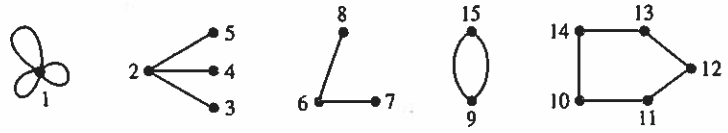


Figure 6.9: A graph with a number of connected components

4. How many connected components does the graph in Figure 6.9

5. Find all induced cycles in Figure 6.9.

6. What is the size of the largest induced K_n in Figure 6.9?

7. Find a largest induced K_n (in words, a largest complete subgrap Figure 6.8.

8. Find the size of a largest induced P_n in Figure 6.9.

9. A graph with no cycles is called a **forest**. Show that if a forest h vertices, e edges, and c connected components, then $v = e + c$.

10. What can you say about a five-vertex simple graph in which eve vertex has degree 4?

11. Draw K_6 so that only three pairs of edges cross.

12. Either prove true or find a counterexample: A graph is a tree if t one and only one path between each pair of vertices.

13. Are there connected graphs with v vertices and $v - 1$ edges that not trees?

14. Are there graphs with v vertices and $v - 1$ edges and no cycles t not trees? Give a proof to justify your answer.

15. Suppose that a graph G is connected, but, for each edge, deletin edge leaves a disconnected graph. What can you say about G? F it.

16. Show that each tree with four vertices can be drawn with one of two drawings in Figure 6.7.

17. Draw the minimum number of drawings of trees possible so tha tree with five vertices has one of those drawings. Explain why y have drawn all possible trees.

18. Draw the minimum number of drawings of trees possible so tha tree with six vertices is represented by exactly one of those draw Explaining why you have drawn all possible drawings is option

19. Find a longest induced cycle in Figure 6.8.