7 Graphs and algorithms

ESSENTIAL UNDERSTANDINGS

- Geometry and trigonometry allow us to quantify the physical world, enhancing our spatial awareness in two and three dimensions.
- This branch provides us with the tools for analysis, measurement and transformation of quantities, movements and relationships.

In this chapter you will learn...

- how to represent a network as a graph, or using an adjacency matrix
- about different types of graphs and networks
- about different ways of moving around a graph
- how to use an adjacency matrix or a transition matrix for a graph
- about the PageRank algorithm
- how to find a minimum spanning tree for a graph
- how to solve the Chinese postman problem
- how to solve the travelling salesman problem.

CONCEPTS

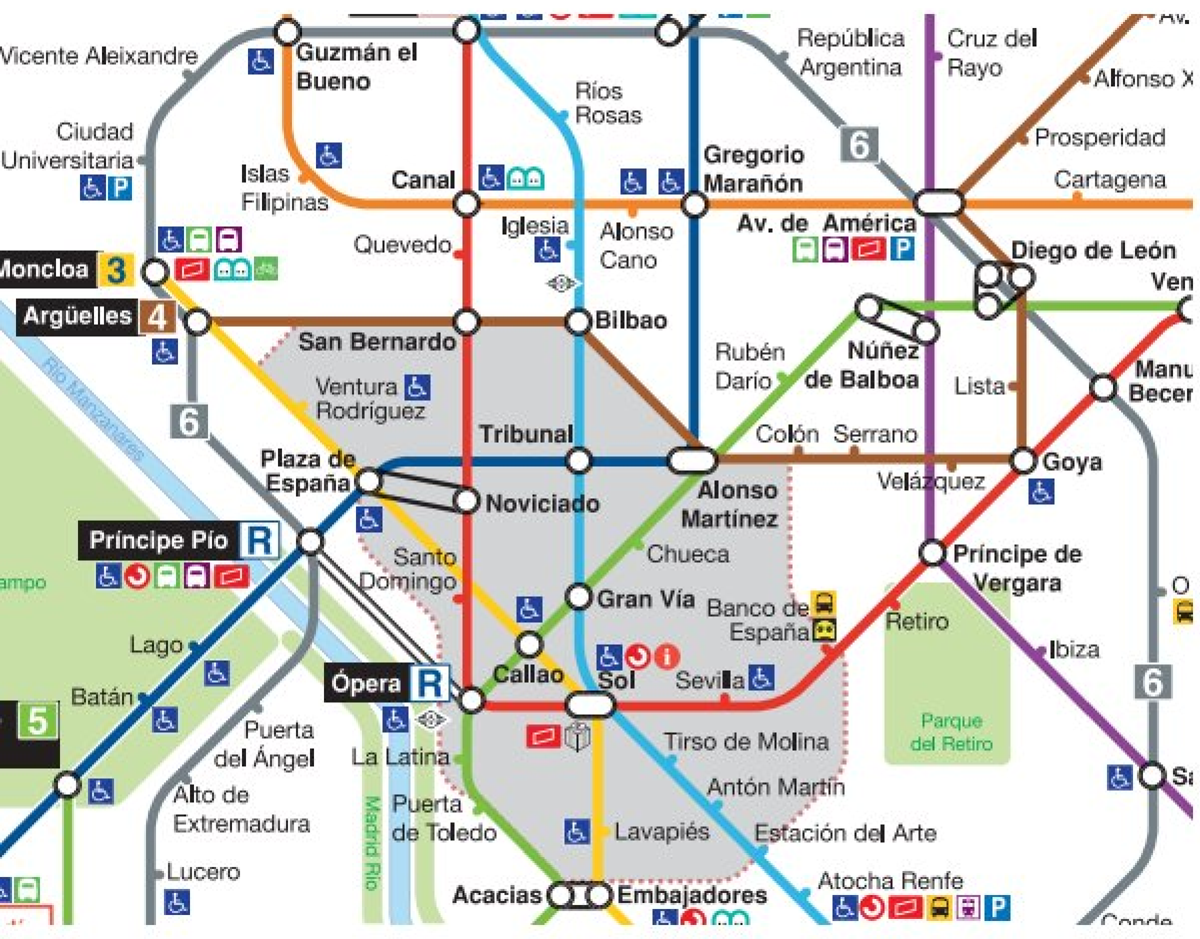
The following concepts will be addressed in this chapter:

- Graph theory algorithms allow us to **represent** networks and to **model** complex real-world problems.

LEARNER PROFILE – Open-minded

How useful is academic education? Is the mathematics chosen to be assessed in this course appropriate and beneficial?

Figure 7.1 What information is being represented in each of these images? What information has been left out?



PRIOR KNOWLEDGE

Use your calculator to find $\mathbf{M}^5$, where

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0.5 & 1.2 \\ 2 & 1.5 & 1.3 & 0 \\ 0 & 2 & 1 & 0.6 \\ 1.2 & 0.8 & 0 & 1 \end{pmatrix}$$

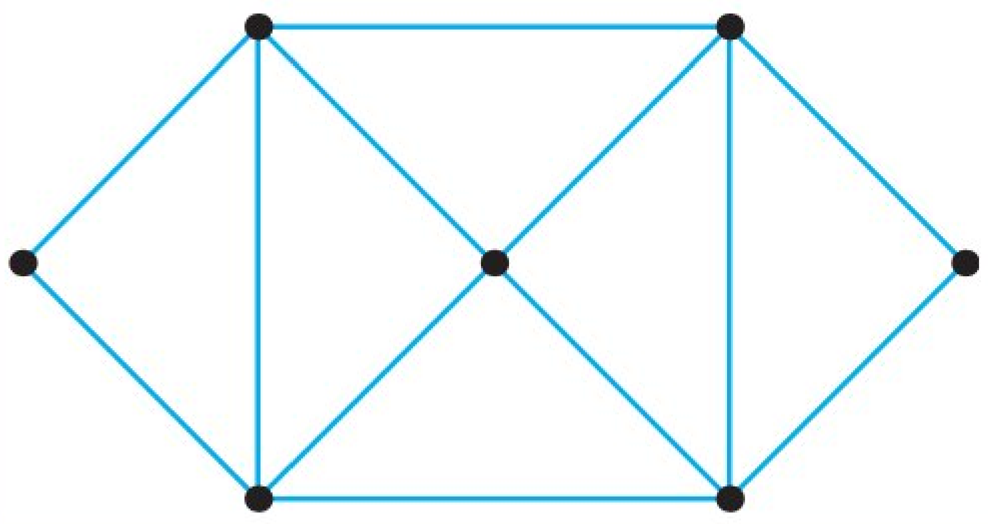
Many real-life systems can be represented as a network of connections between individual components. The mathematical study of such networks is called graph theory, and involves studying the number of connections, ways of moving around a network and the costs or distances involved. It has many applications, from finding optimal travel routes to designing internet search engines.

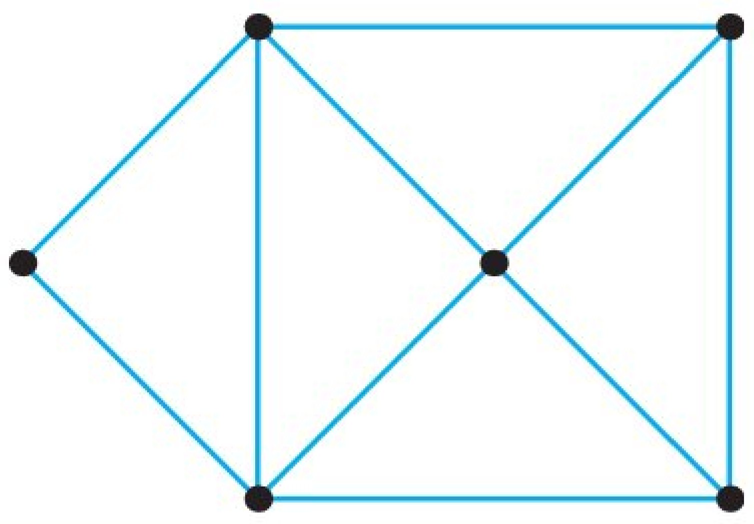
Starter Activity

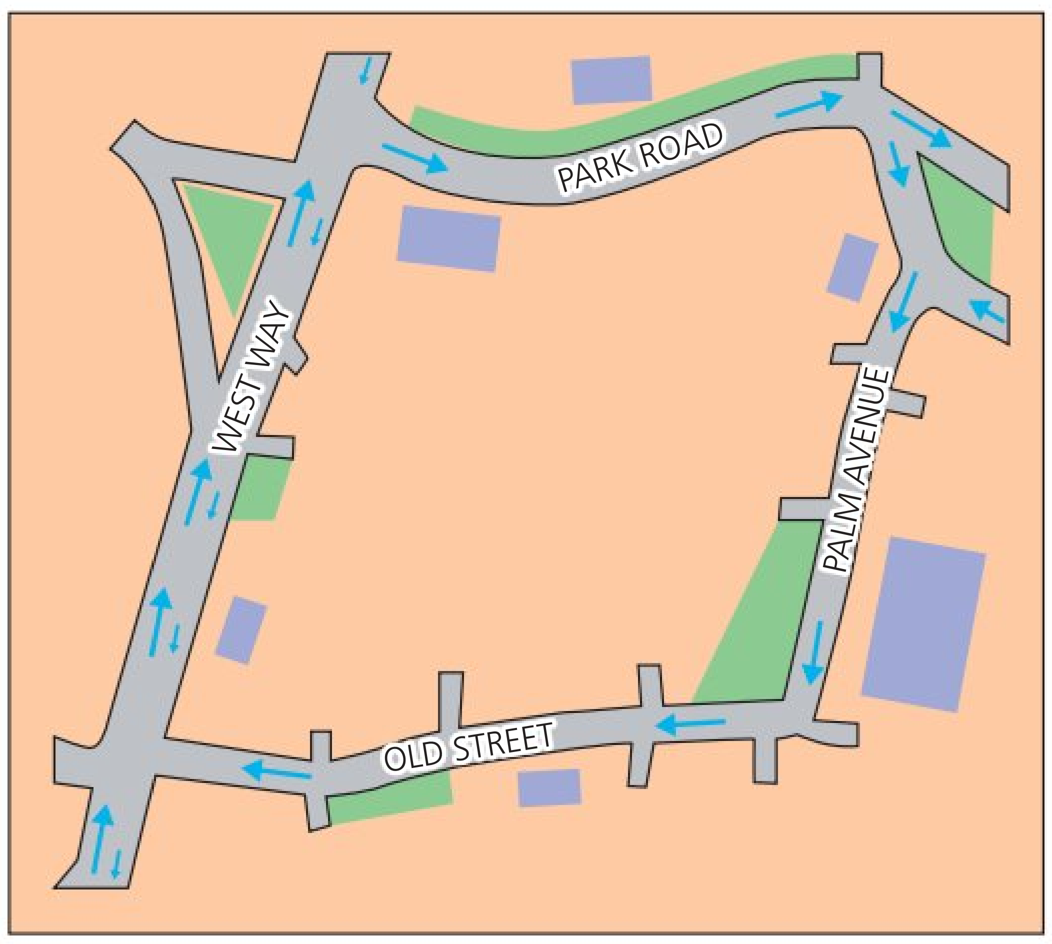
Look at the images in Figure 7.1. Discuss what information is shown in each image. Are there any other ways in which you could represent this information? What information that could be relevant to the situation is not shown in the image?

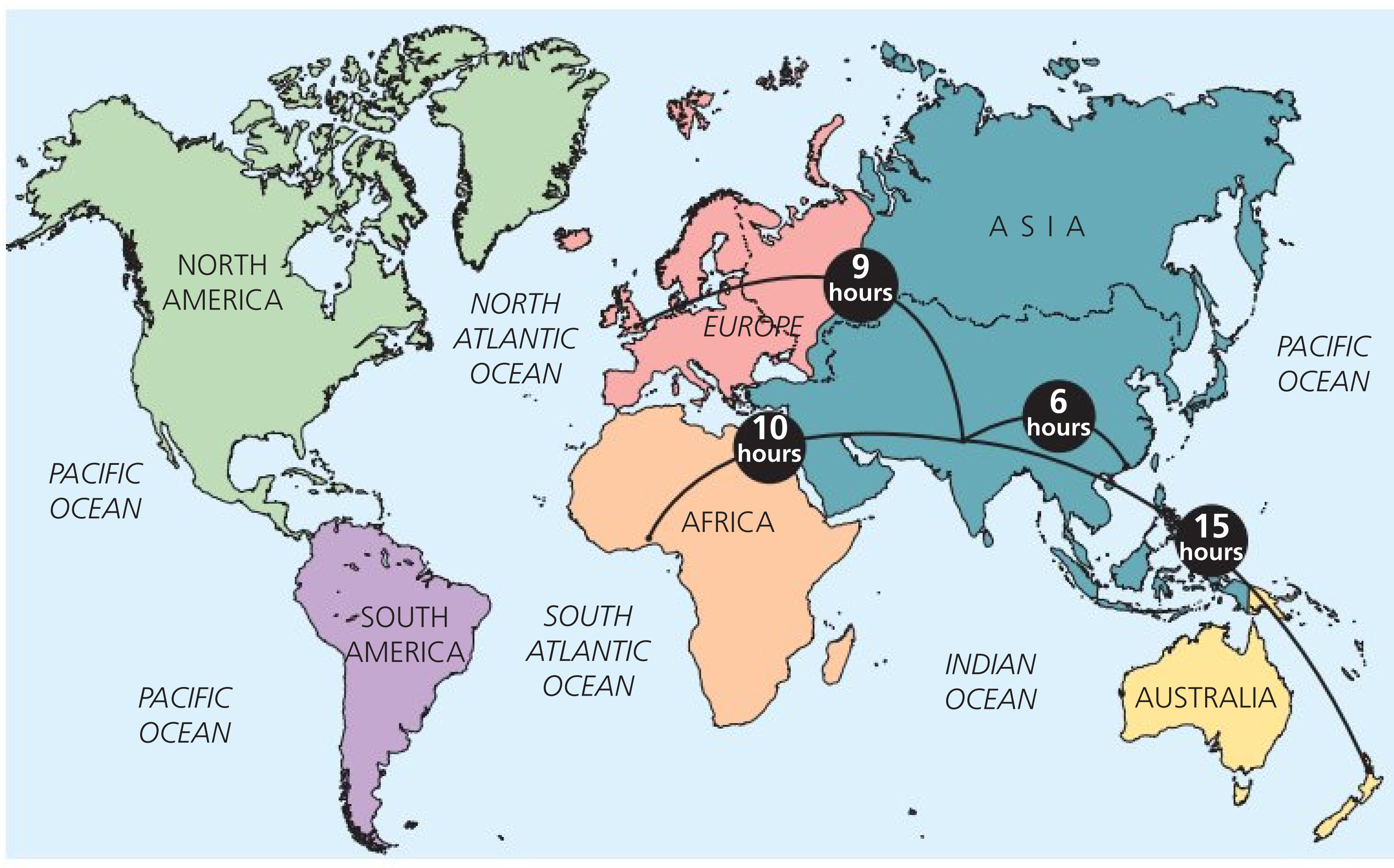
Now look at this problem:

Can you draw the following shapes without lifting the tip of your pen off the paper and without going over any lines twice? What if you have to start and finish at the same point?







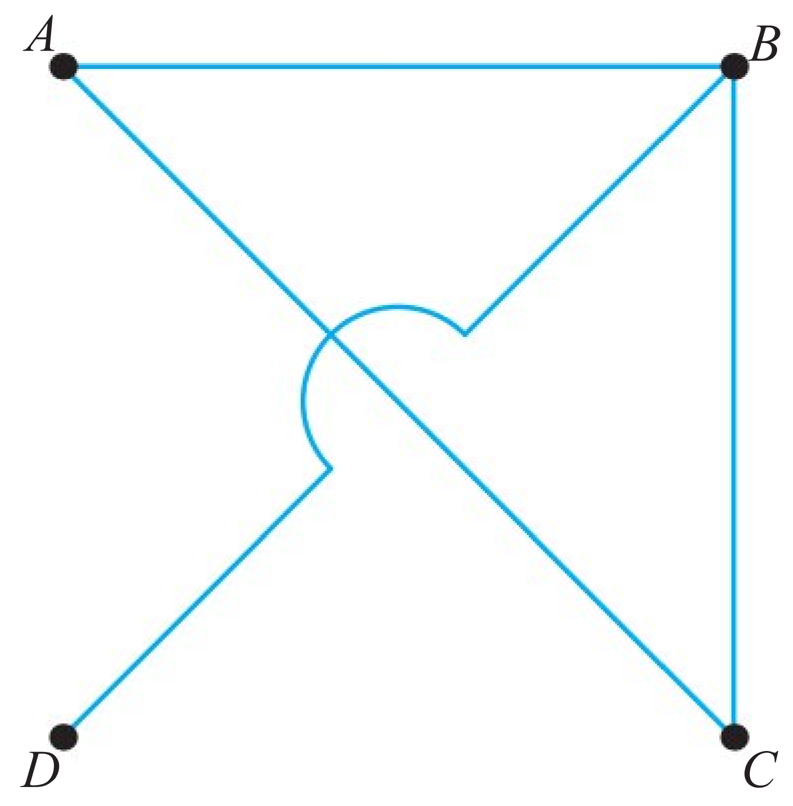


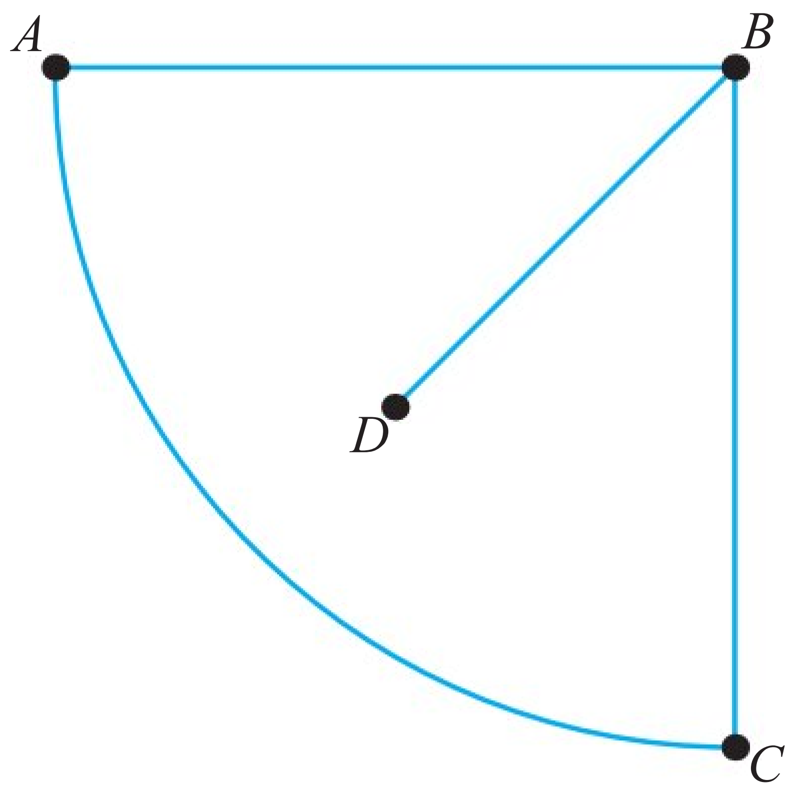
7A Introduction to graph theory

Definitions

A **graph** consists of **vertices** connected by **edges**. Two vertices are called **adjacent** if they are joined by an edge. Two edges are called adjacent if they share a common vertex.

Graphs are often represented by showing vertices as point and edges as lines between them. When we draw a graph, the position of the vertices and the shape of the edges is not relevant; the only thing that matters is which vertices are adjacent. So the two diagrams you see here represent the same graph.





The vertices of the graph are usually labelled by capital letters. The graph has vertices *A*, *B*, *C* and *D* and edges *AB*, *AC*, *BC* and *BD*.

It is often important to know the number of edges coming out of each vertex. This is called the **degree** of a vertex. For example, in the graph shown,

$$\deg(A) = 2,\ \deg(B) = 3,\ \deg(C) = 2,\ \deg(D) = 1$$

Tip

The edges may intersect at points other than vertices, so it is important to label the vertices clearly.

Graph theory is a relatively new branch of mathematics, with its beginnings in the eighteenth century and many important developments in the early twentieth century. Its development was mainly motivated by applications and it has had contributions from many different communities – mathematicians, computer scientists, chemists, even linguists. For those reasons some of the terminology is not firmly established and standardized yet, so you may find slightly different definitions in other books. For example, vertices and edges are also referred to as nodes and arcs. The terminology we use here is what will be used in your IB examination.

You are the Researcher

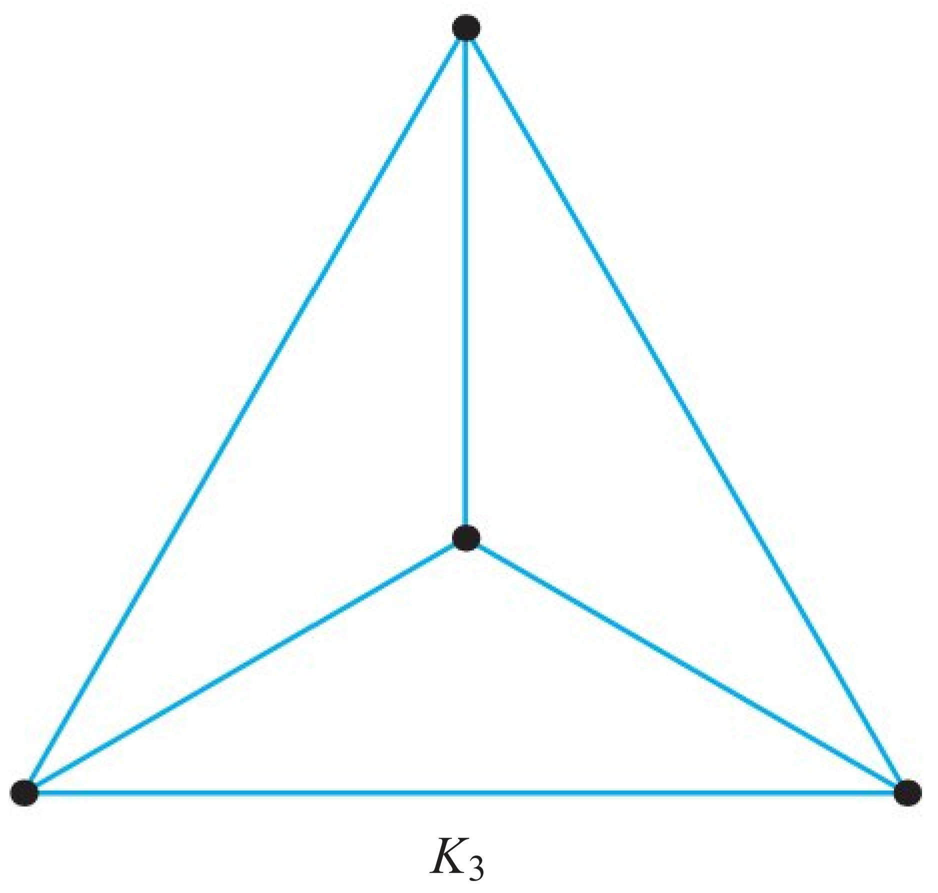
There are many interesting relationships connecting the number of edges, number of vertices and their degrees in a graph. One of the most important is the handshaking lemma, which states that the sum of the degrees of all the vertices in a graph is equal to twice the number of edges.

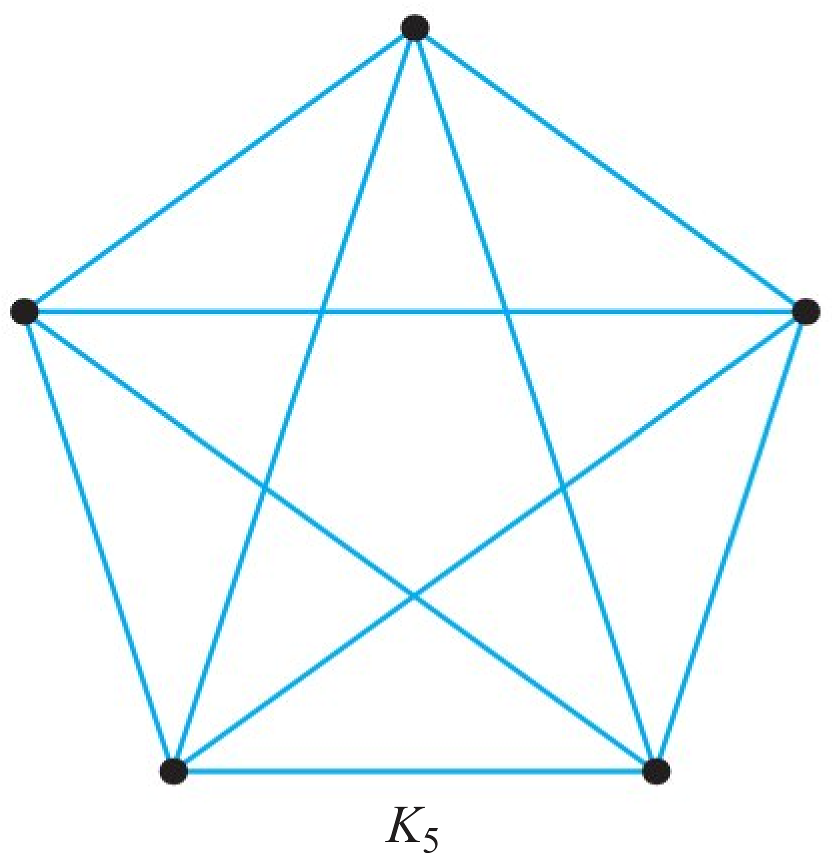
Types of graphs

Different types of graphs are appropriate for different practical situations.

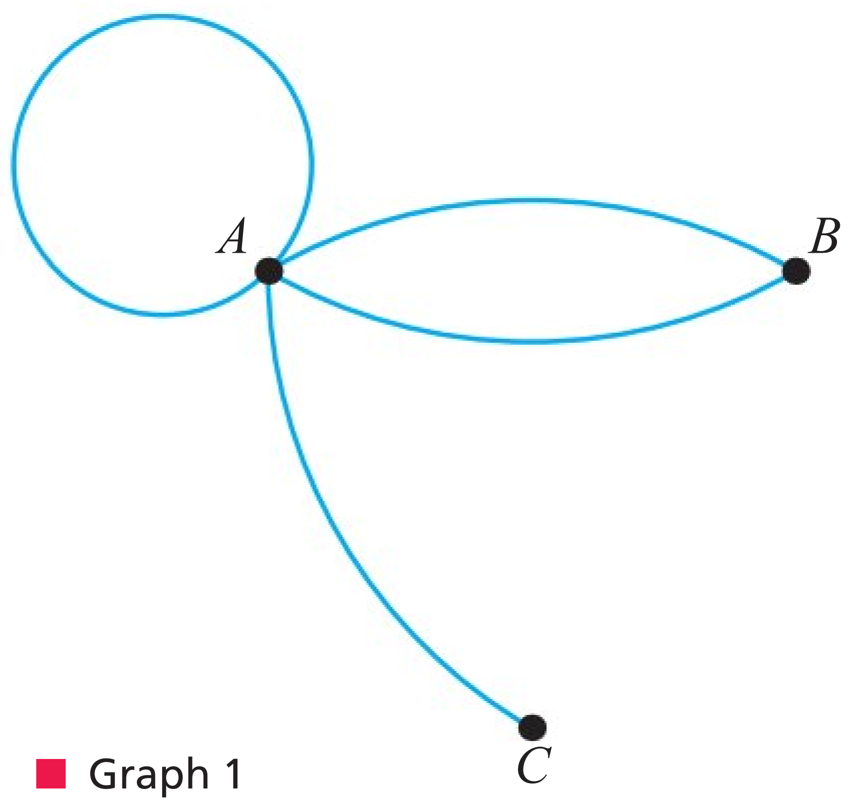
Sometimes every vertex is connected to every other vertex by exactly one edge. This is called a **complete graph**. A symbol for a complete graph with n vertices is K_n.

You can use your knowledge of counting selections to show that the number of edges in K_n is $\binom{n}{2}$.





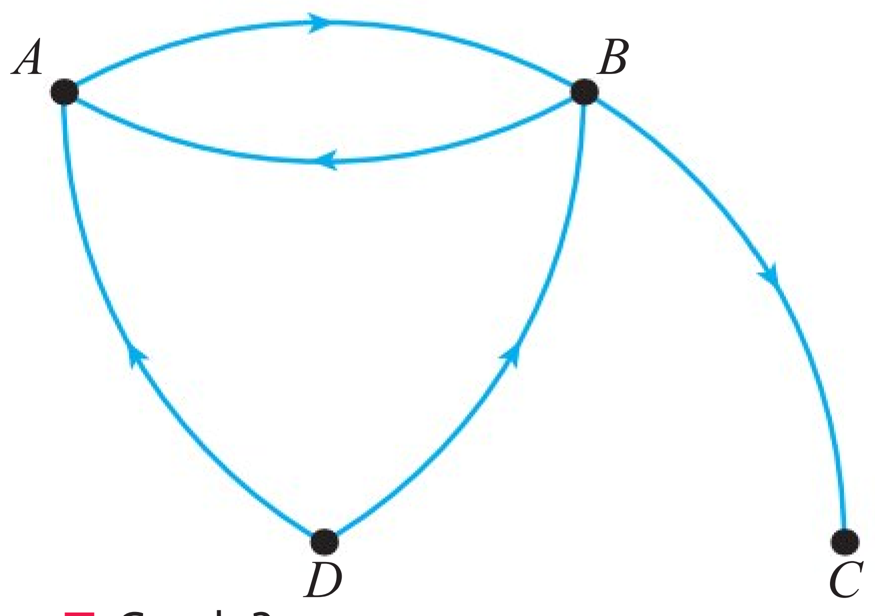
In some situations, it is possible to have more than one edge between two vertices, or for a vertex to be joined to itself, whereas in other situations this is not allowed. A **simple graph** has no multiple edges and no vertex is joined to itself. Graph 1, shown alongside, is not simple, but the graphs K_3 and K_5 above are simple graphs.



■ Graph 1

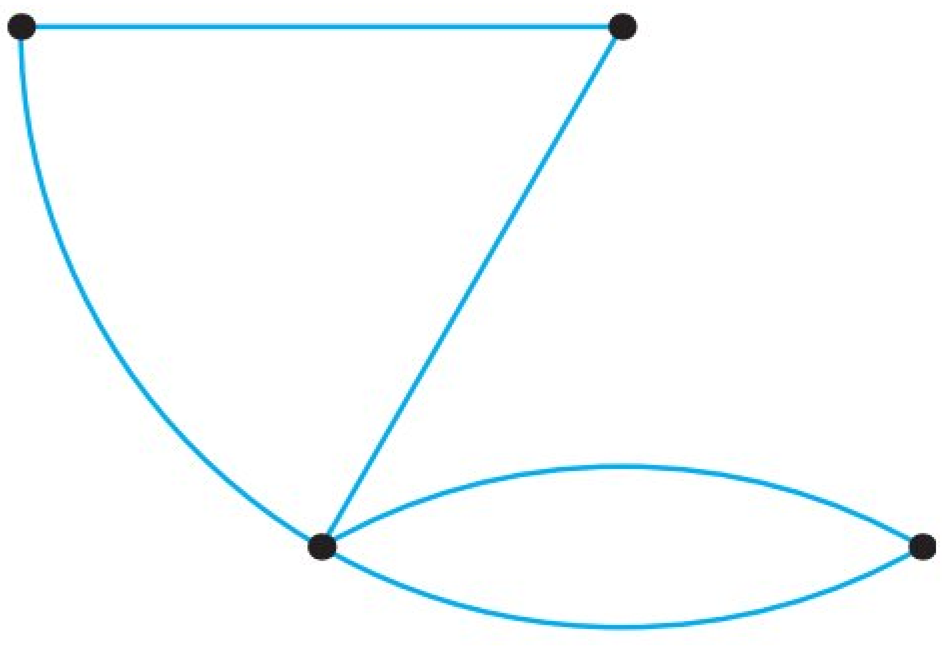
Sometimes we can only move in one direction between vertices, for example when modelling a road network with one-way roads. In that case, we can put arrows on the edges to indicate the allowed direction. The resulting graph is called a **directed graph**, or **digraph**. In Graph 2, shown on the right, it is possible to get from D to A but not from A to D, while both directions between A and B are allowed.

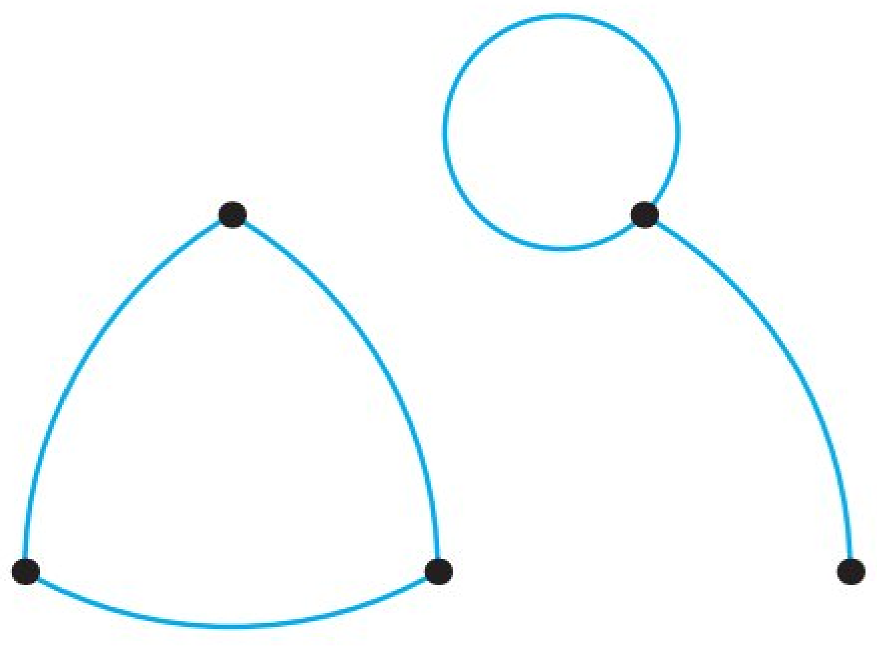
For a directed graph we can distinguish between **in-degree** and **out-degree** of each vertex. For example, vertex A in Graph 2 has in-degree of 2 and out-degree of 1.



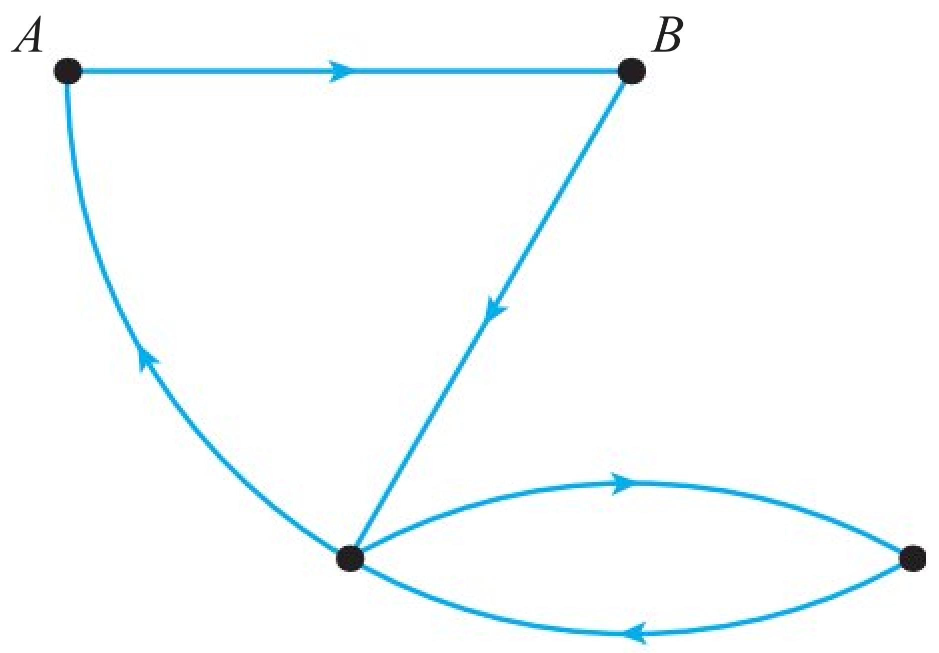
■ Graph 2

Graphs are often used to model situations where it is important to know how to get from one vertex to another. A graph is called **connected** if all pairs of vertices are connected (directly or indirectly). This means that the graph cannot be split into two parts.

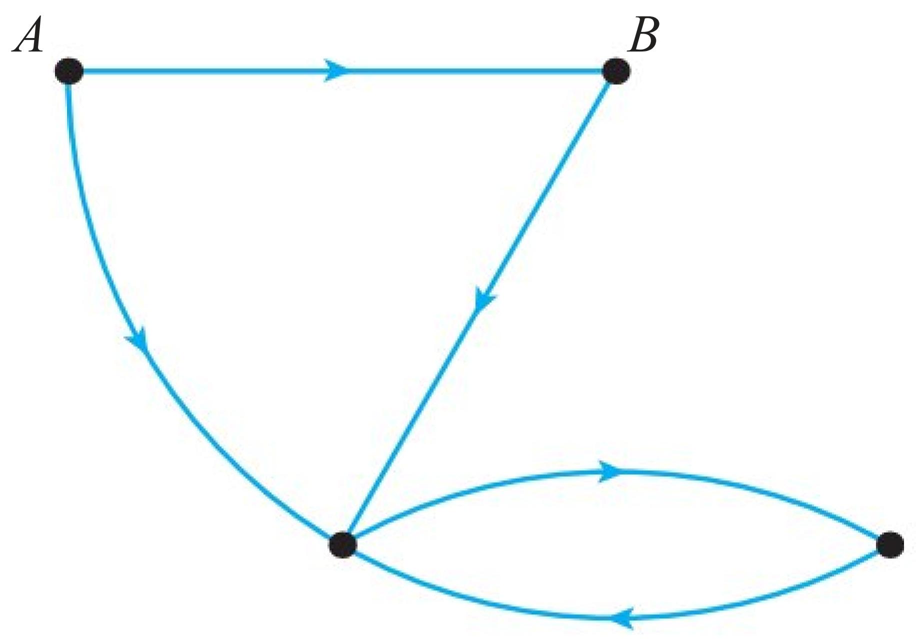

■ Connected


■ Not connected

In a directed graph it may be possible to get from A to B, but not from B to A. If all pairings of any two vertices are connected (in both directions) the graph is called **strongly connected**.



■ Strongly connected



■ Not strongly connected (no path from B to A)

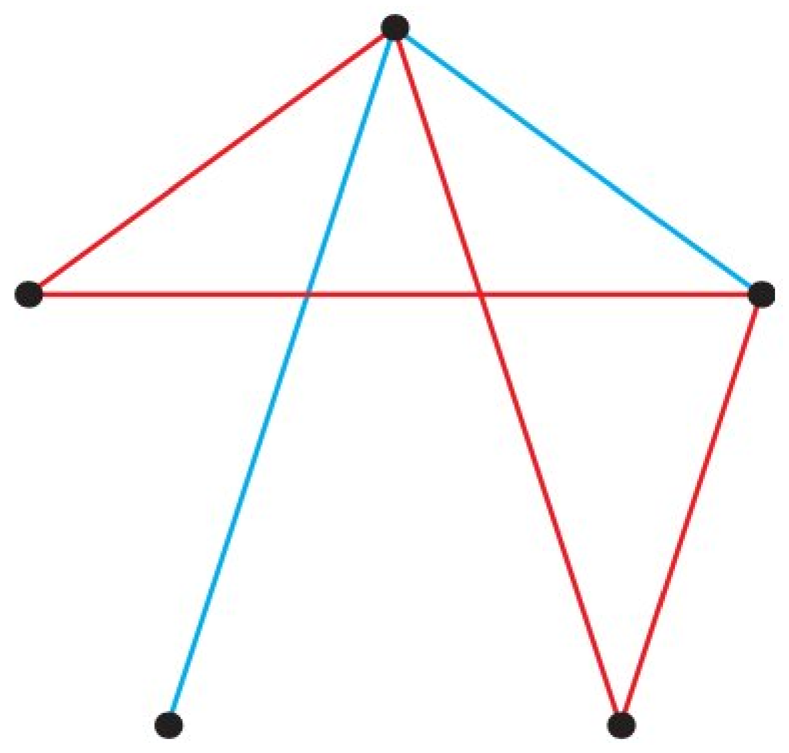
You are the Researcher

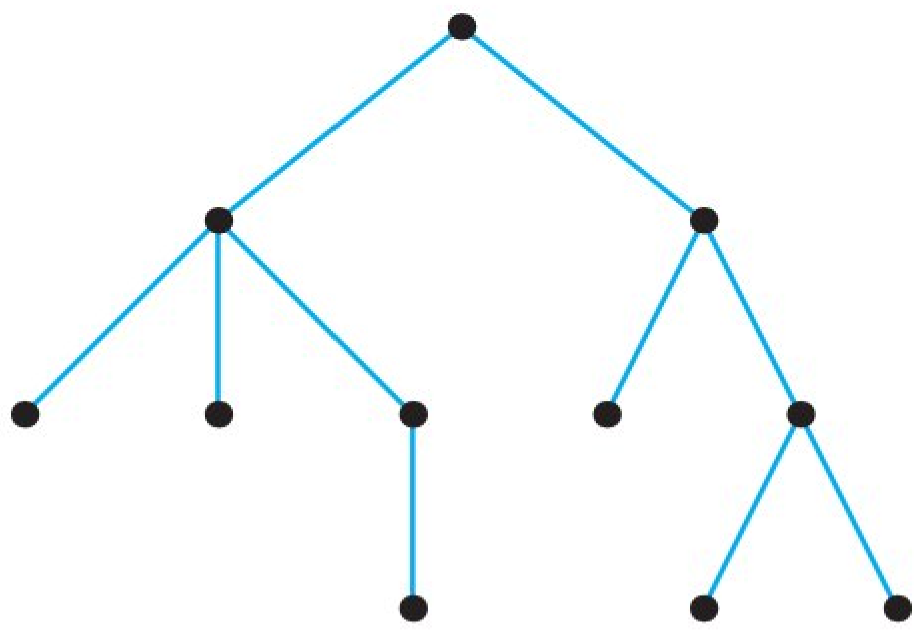
Another important type of graph is planar graphs, which can be drawn without edges crossing. There are many interesting results concerning the possible number of vertices and edges in a planar graph, including the Euler characteristic that also applies to polyhedra.

Subgraphs and trees

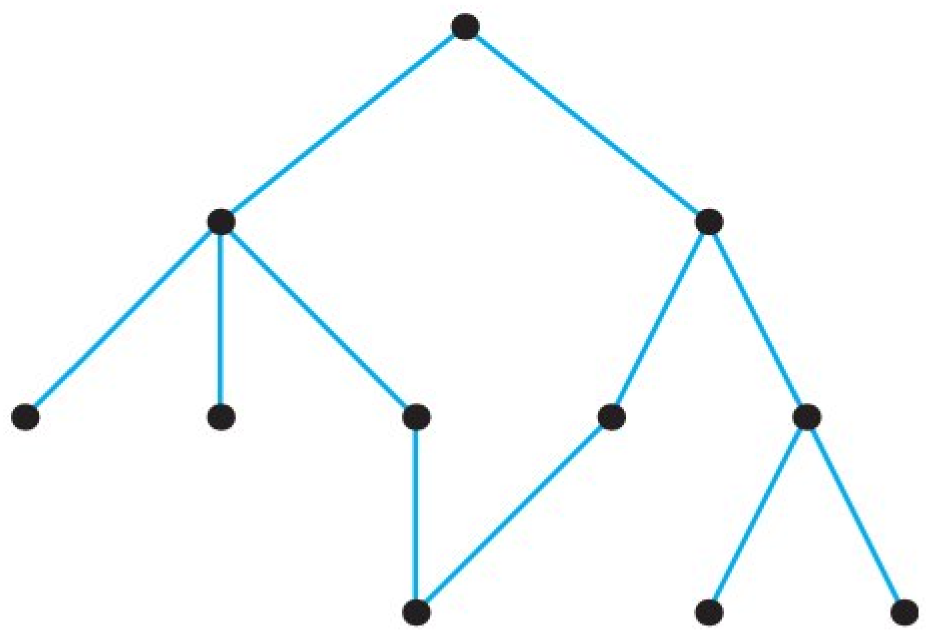
A **subgraph** of a given graph is a new graph formed by using only some of the edges of the original graph. In the diagram shown a subgraph is indicated by the red edges. Note that every simple graph with n vertices is a subgraph of the complete graph K_n.

A **tree** is a connected graph for which it is not possible to find a sequence of distinct edges that returns to the starting vertex (i.e. there are no closed paths).


■ A subgraph is shown in red



■ A tree



■ Not a tree

Finding a subgraph which is a tree has many applications, for example in the minimum connector problem which you will meet in Section C.

KEY POINT 7.1

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

Note that this is the smallest possible number of edges needed to make a graph with n vertices connected, so if any one edge is removed the graph is no longer connected.

WORKED EXAMPLE 7.1

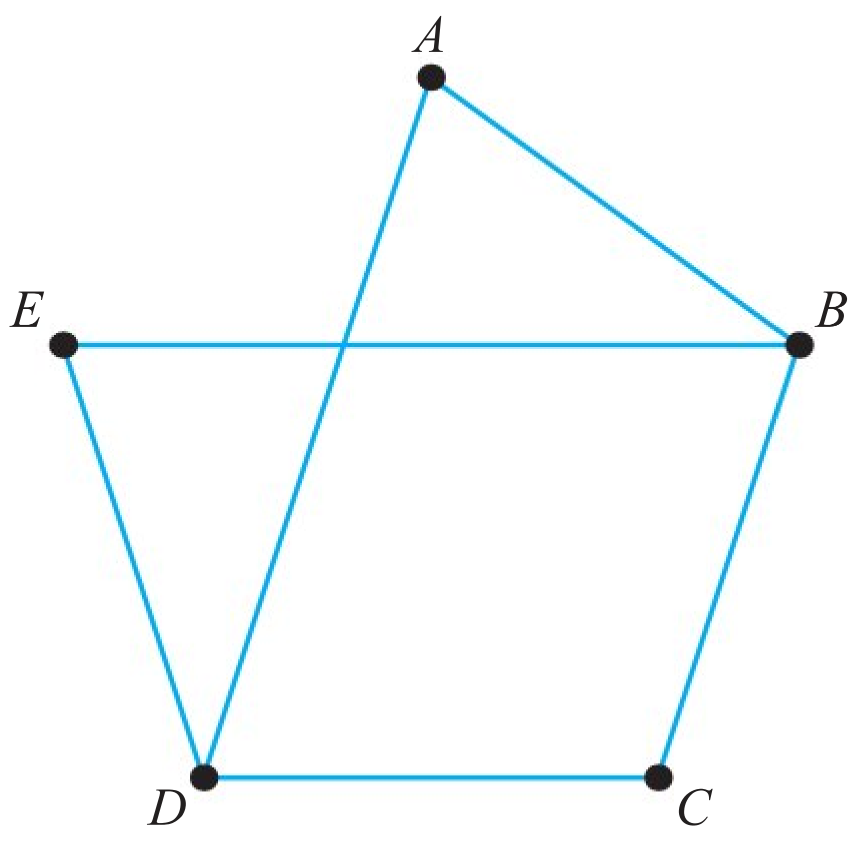
Which of the following terms describe the graph shown in the diagram? For those that do not, give a reason for your answer.

a Simple

b Connected

c Complete

d Tree



The graph contains no multiple edges between vertices, and no edges connecting a vertex to itself

a The graph is simple.

A connected graph cannot be split into two separate components

b The graph is connected.

A complete graph has an edge between each vertex and every other vertex

c The graph is not complete. For example, there is no edge between A and C.

A tree contains no sequences of vertices that return to the starting vertex

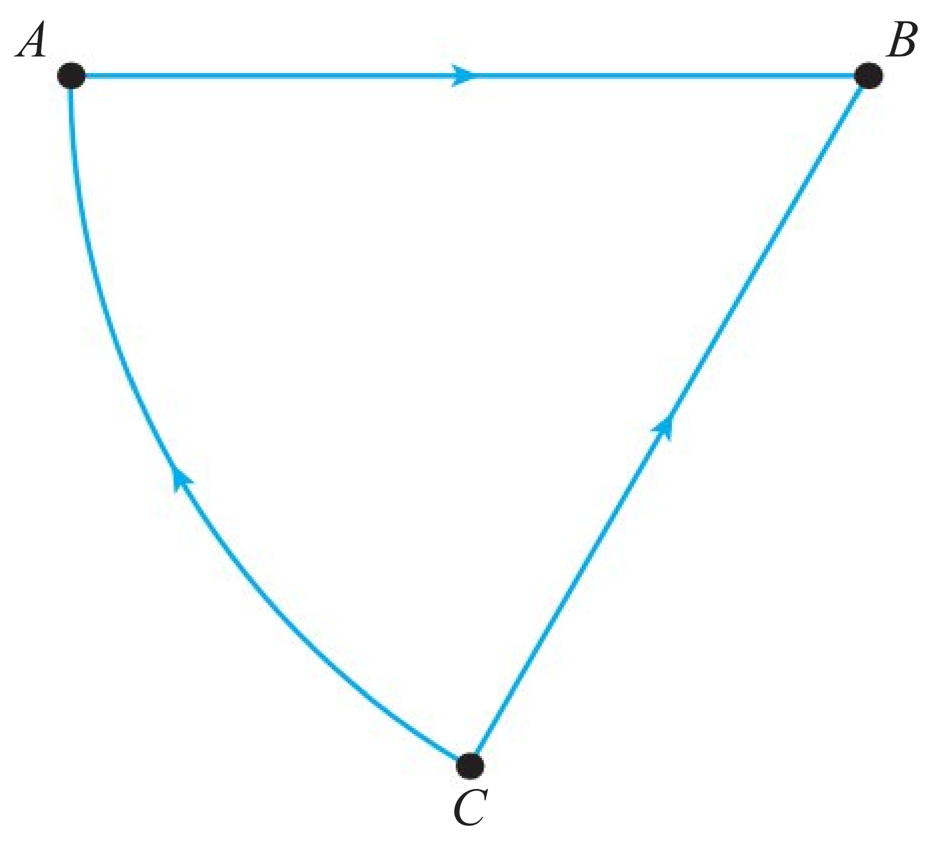
d The graph is not a tree, because $ABCDA$ returns to the starting vertex.

WORKED EXAMPLE 7.2

Which of the following terms describe the graph shown in the diagram? For those that do not, give a reason for your answer.

a Directed

b Strongly connected



There are arrows on the edges, indicating allowed directions

a The graph is directed.

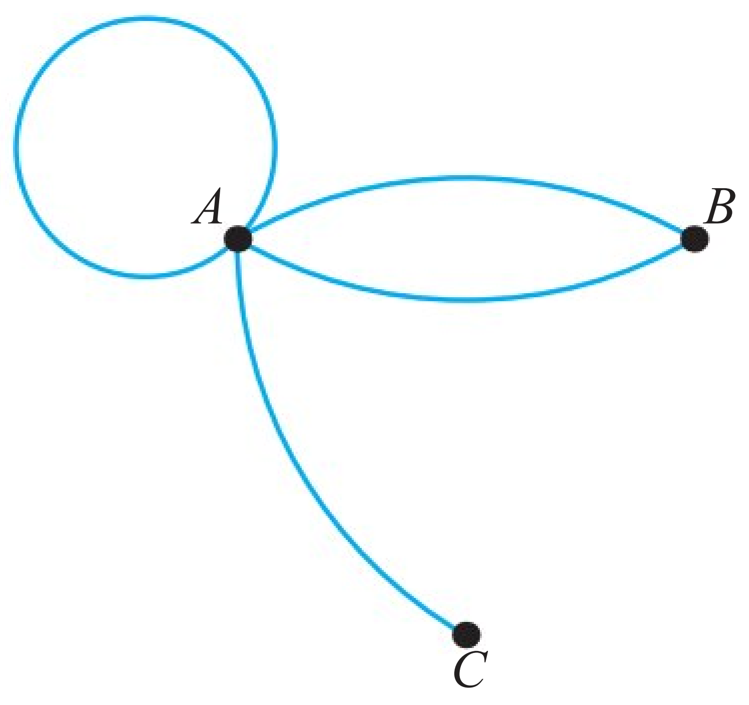
For a graph to be strongly connected, there must be a path between each and every two vertices, in both directions

b The graph is not strongly connected. For example, there is no path from B to A.

Adjacency matrices

If a graph is very large it is impractical to draw it. Instead, you can use a matrix to show the number of edges between each pair of vertices. This is called the **adjacency matrix** for the graph.

For example, the adjacency matrix for the graph below is shown alongside.



$$\begin{array}{c} A \\ B \\ C \end{array} \begin{pmatrix} 2 & 2 & 1 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Tip

Notice that the loop connecting A to itself contributes two connections in the adjacency matrix: the entry (A, A) is 2.

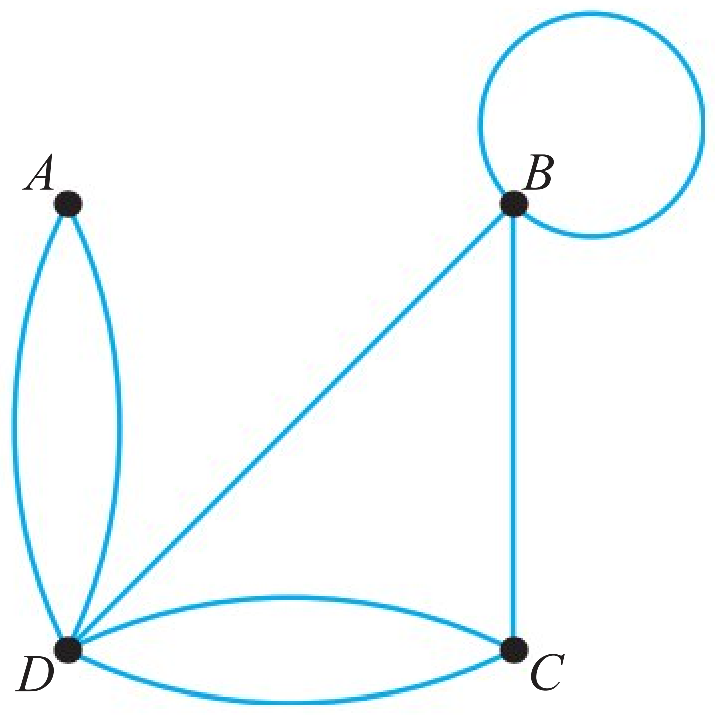
Each row/column of the matrix represents a vertex. The edges go from the vertex on the side to the vertex on top. This graph is undirected, so the matrix is symmetrical. For example, there are two edges from B to A, and one edge from C to A. The numbers on the diagonal show the number of edges from each vertex to itself.

WORKED EXAMPLE 7.3

a Write down the adjacency matrix for the following graph.

b Draw the graph represented by the following adjacency matrix:

$$\begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{pmatrix} 0 & 1 & 3 & 0 \\ 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 2 \\ 0 & 1 & 2 & 2 \end{pmatrix}$$



Look at each vertex in turn

a
$$\begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{pmatrix} 0 & 0 & 0 & 2 \\ 0 & 2 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 2 & 1 & 2 & 0 \end{pmatrix}$$

From the first row, there are no edges from A to itself, B or C; there are two edges from A to D. So the first row is 0, 0, 0, 2

From the second row, there are edges from B to B, C and D. Remember that the loop at B counts as two edges from B to B

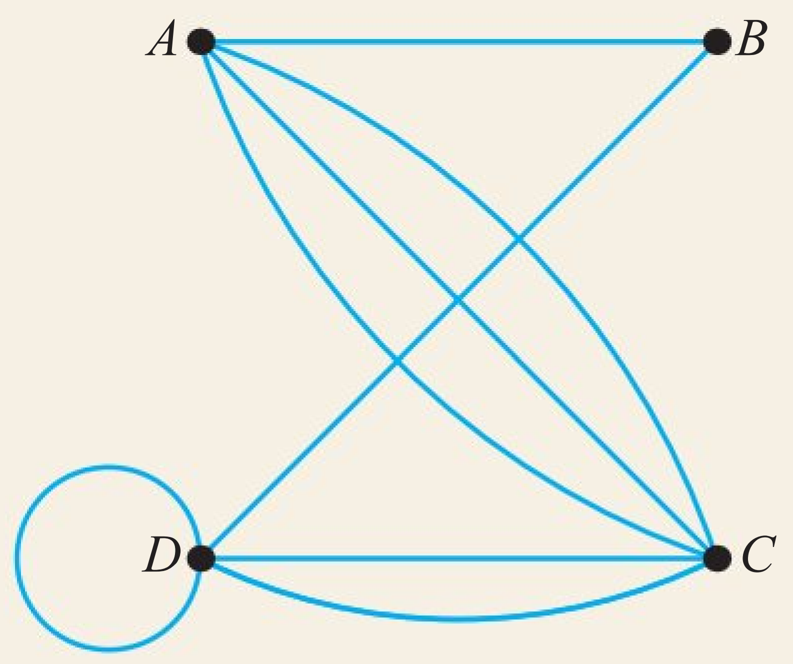
From C, there is one edge to B and two edges to D

Finally, from D, there are 2 edges to A, one edge to B, two edges to C and no edges to D

You should check that your matrix is symmetrical (as the graph is undirected)

Taking each vertex in turn, connect it to the other vertices by the number of edges indicated

b



You can work out the degree of each vertex by looking at the corresponding row (or column) of the adjacency matrix, and summing the elements.

KEY POINT 7.2

For an undirected graph:

- the adjacency matrix is symmetrical about the leading diagonal
- the degree of a vertex equals the sum of the entries in the corresponding row (or column).

Additionally, if the graph is simple:

- the adjacency matrix contains only zeros and ones.

WORKED EXAMPLE 7.4

A graph has the following adjacency matrix:

$$\begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{pmatrix} 2 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 \\ 1 & 0 & 2 & 0 \end{pmatrix}$$

a List all the vertices adjacent to B.

b Find the degree of vertex A.

Row B has 1s in the columns corresponding to A and C

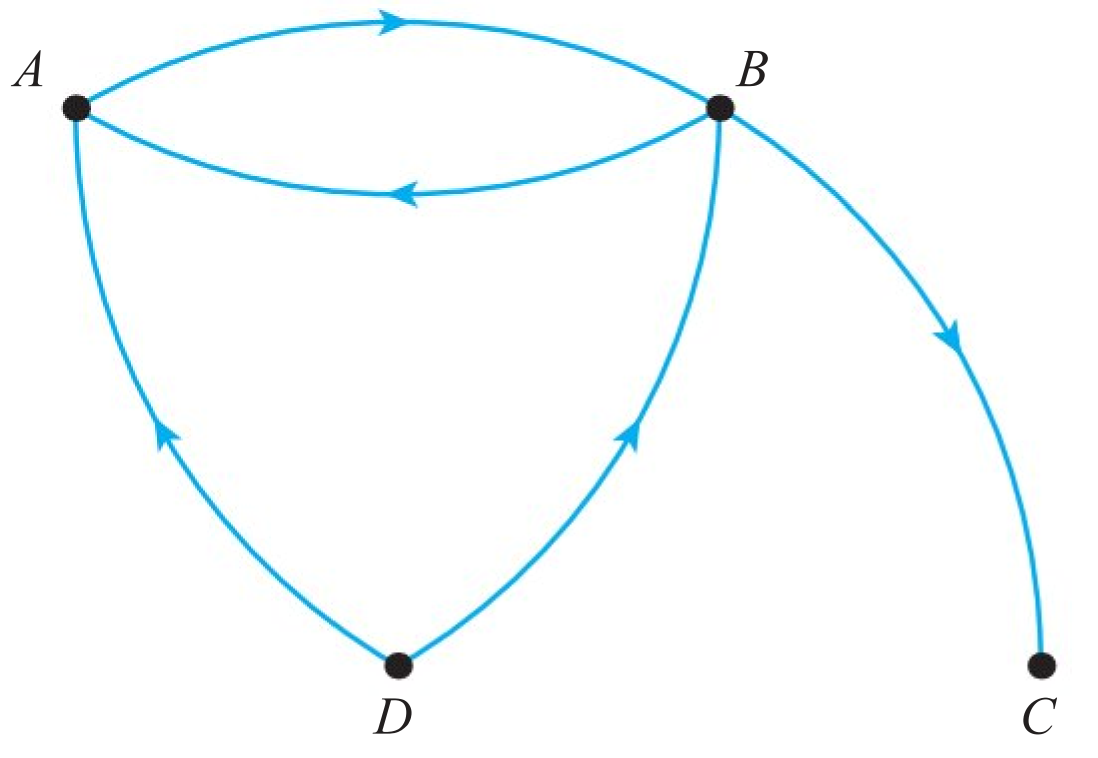
a A, C

This is the sum of all the numbers in row A

b $2 + 1 + 0 + 1 = 4$

Adjacency matrices can also be used to represent directed graphs. In this case, the matrix will not be symmetrical on the leading diagonal. The rows give the starting vertex and the columns the end vertex.

Graph 2 and its adjacency matrix are shown below.



$$\text{from}\quad \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 0 & 0 \\ B & 1 & 0 & 1 & 0 \\ C & 0 & 0 & 0 & 0 \\ D & 1 & 1 & 0 & 0 \end{array}$$

to

The out-degree of a vertex is the sum of the entries in its row, and the in-degree is the sum of the entries in its column.

TOK Links

The decision to use rows for the starting vertex and columns for the end vertex of an edge is purely a convention, with no underlying mathematical reason. Do such conventions help or hinder the learning of a new concept?

WORKED EXAMPLE 7.5

A directed graph is represented by the following adjacency matrix:

$$\begin{array}{c} A \\ B \\ C \\ D \end{array}\begin{pmatrix} 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 2 \\ 0 & 3 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

a Determine

i the out-degree of vertex A

ii the in-degree of vertex D.

b Draw the graph.

The edges starting from A are shown in the first row of the matrix: there is one edge from A to B and there are two edges from A to D

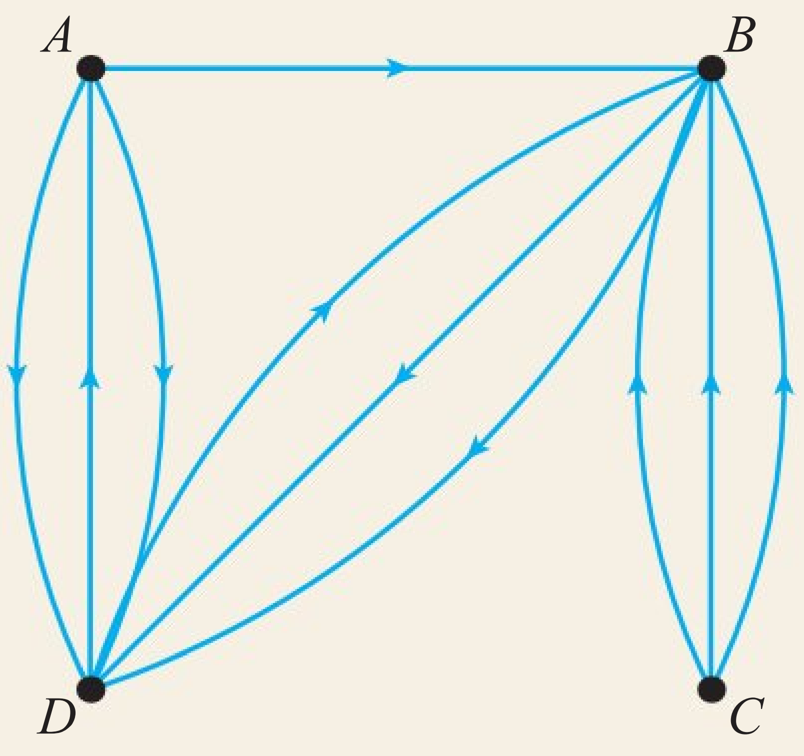
a i The out-degree of A is $0 + 1 + 0 + 2 = 3$.

The edges going into D are shown in the fourth column of the graph. There are two edges coming from A, two from B and one from C

ii The in-degree of D is $2 + 2 + 1 + 0 = 5$.

Start by marking the four vertices. Then use each row of the table to draw the edges from the corresponding vertex

b

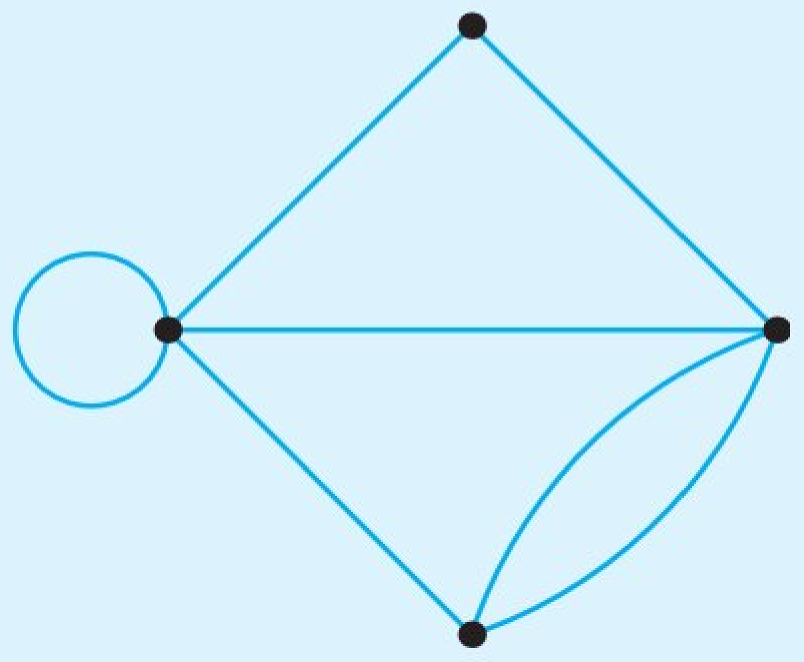


Exercise 7A

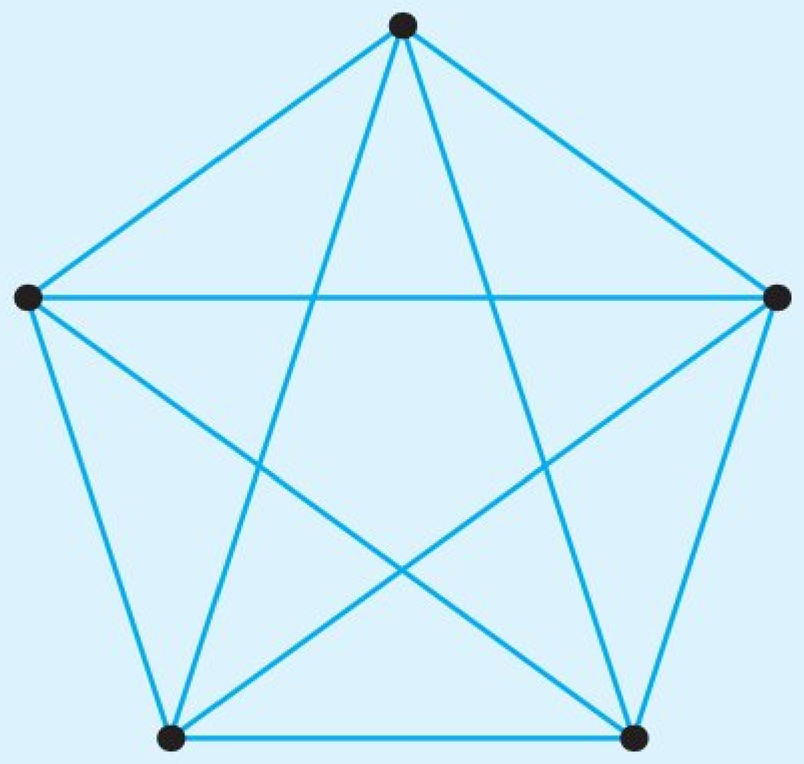
For questions 1 and 2, use the method demonstrated in Worked Example 7.1 to say which of the following terms describe each graph.

i complete **ii** connected **iii** simple **iv** tree

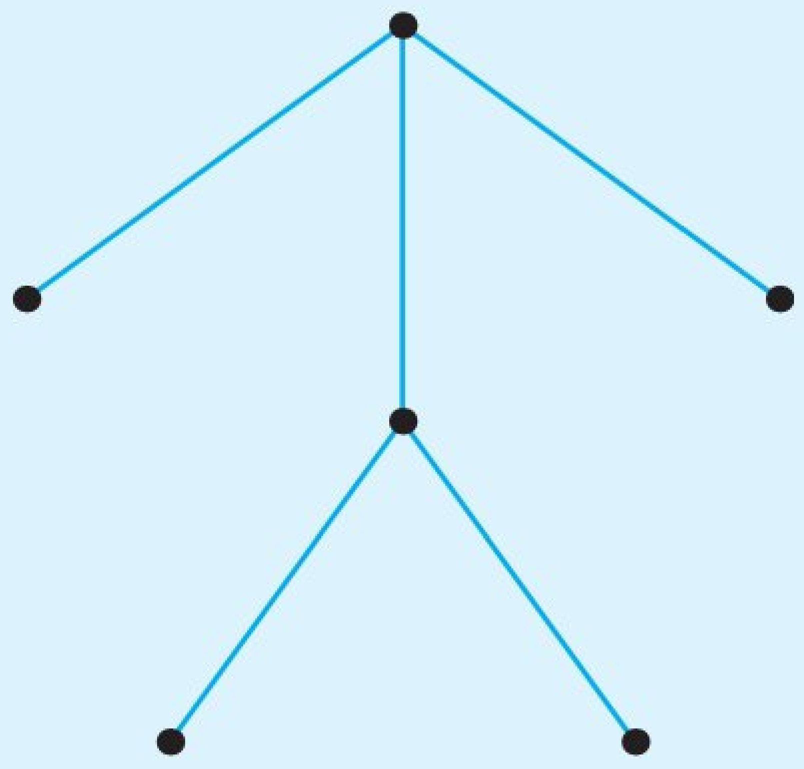
1 **a**



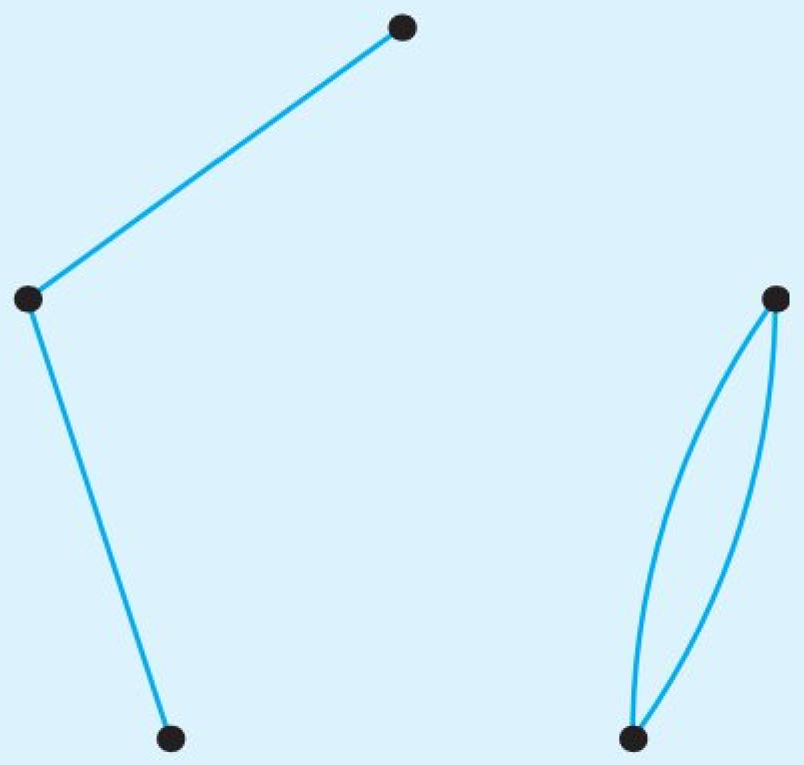
b



2 **a**

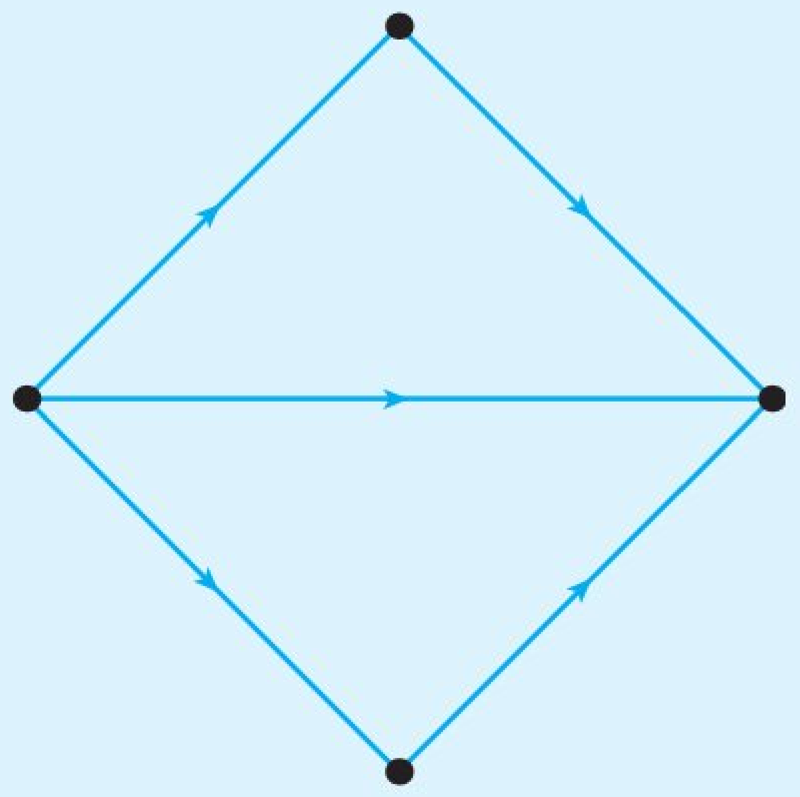


b

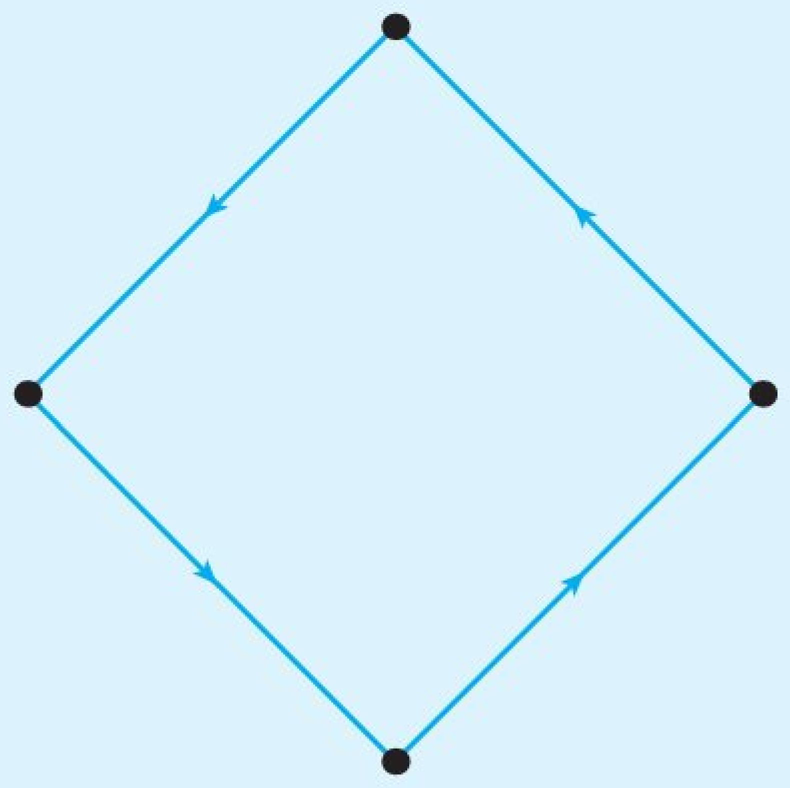


For questions 3 and 4, use the method demonstrated in Worked Example 7.2 to decide whether these directed graphs are strongly connected.

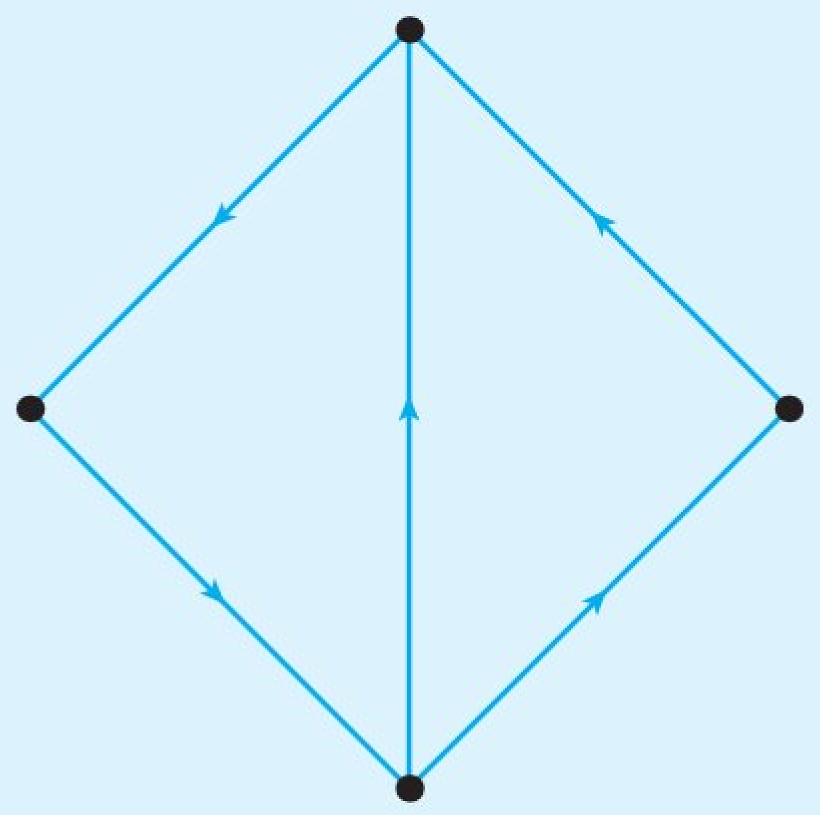
3 **a**



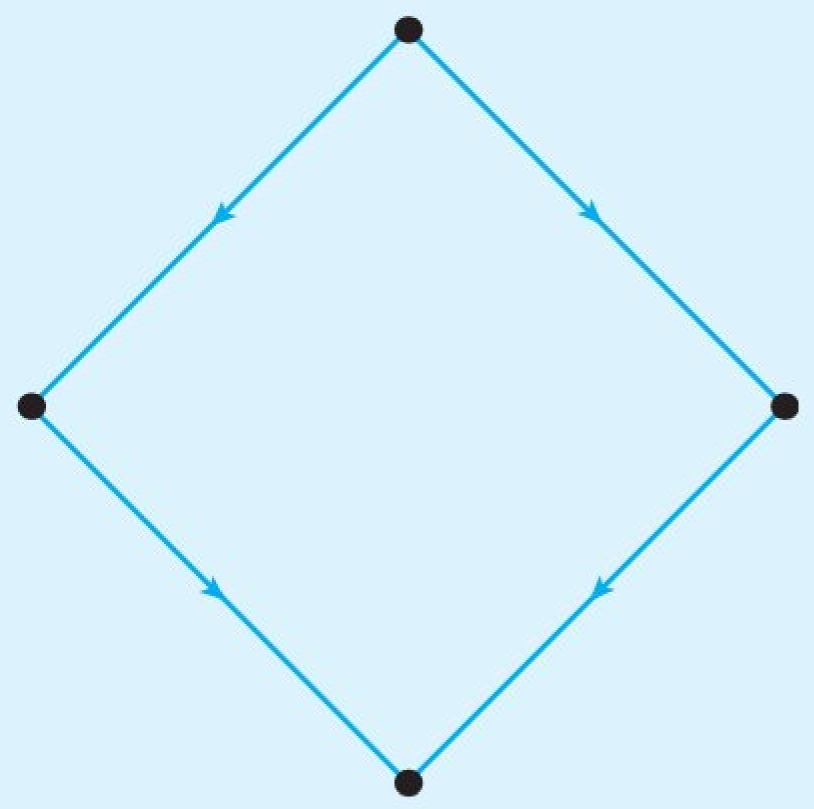
b



4 **a**

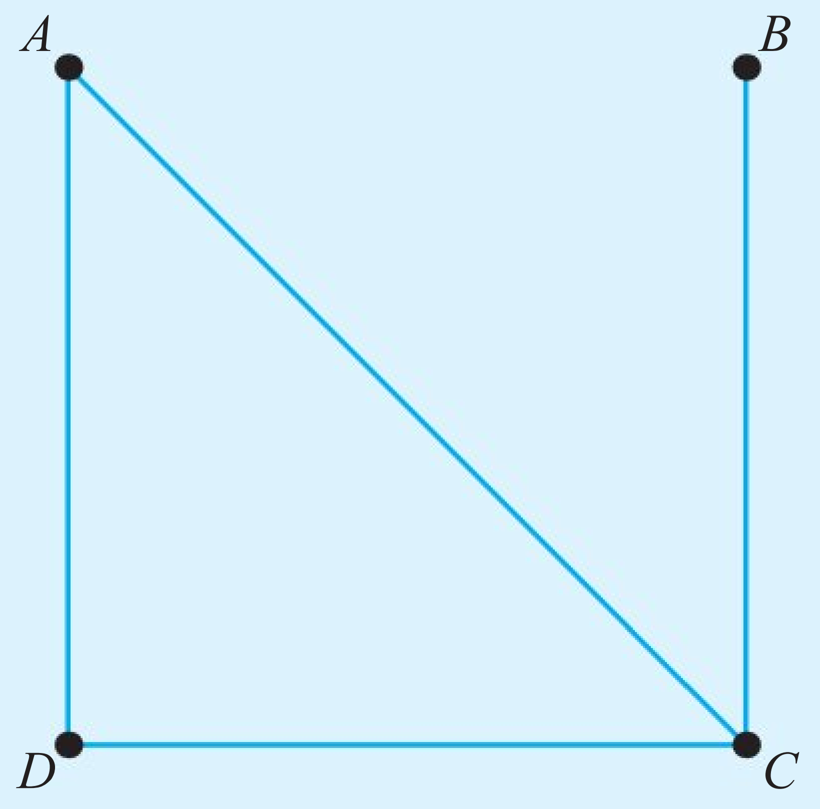


b

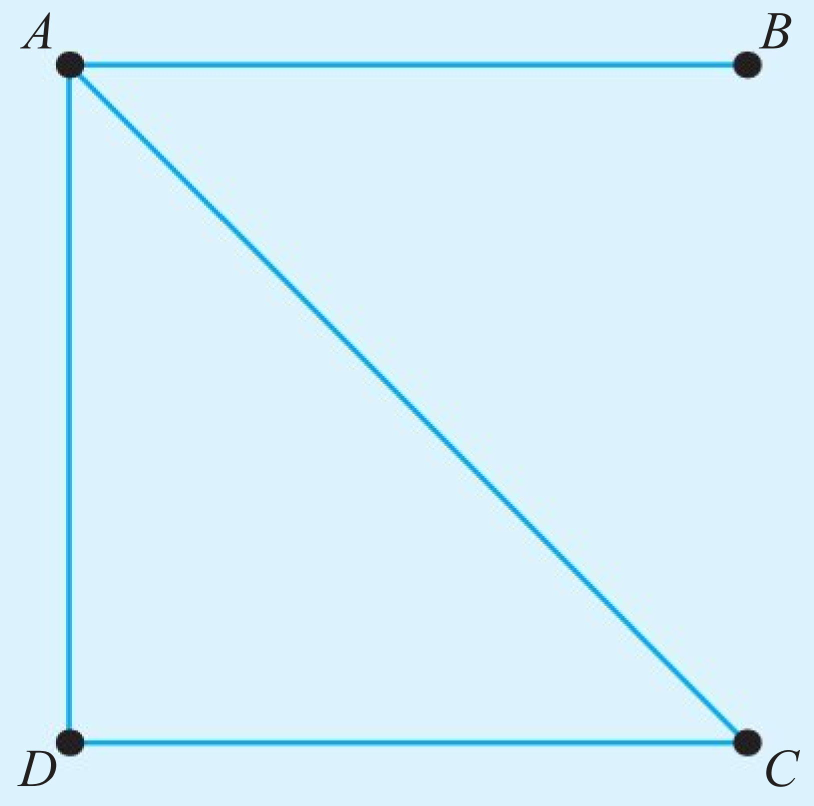


For questions 5 and 6, use the method demonstrated in Worked Example 7.3a to construct an adjacency matrix for each graph.

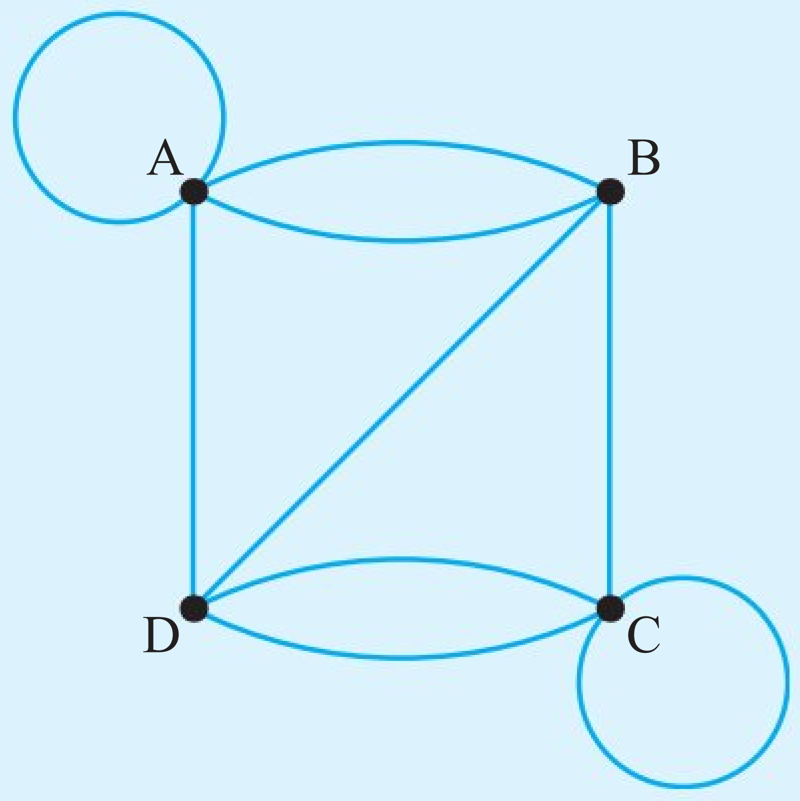
5 **a**



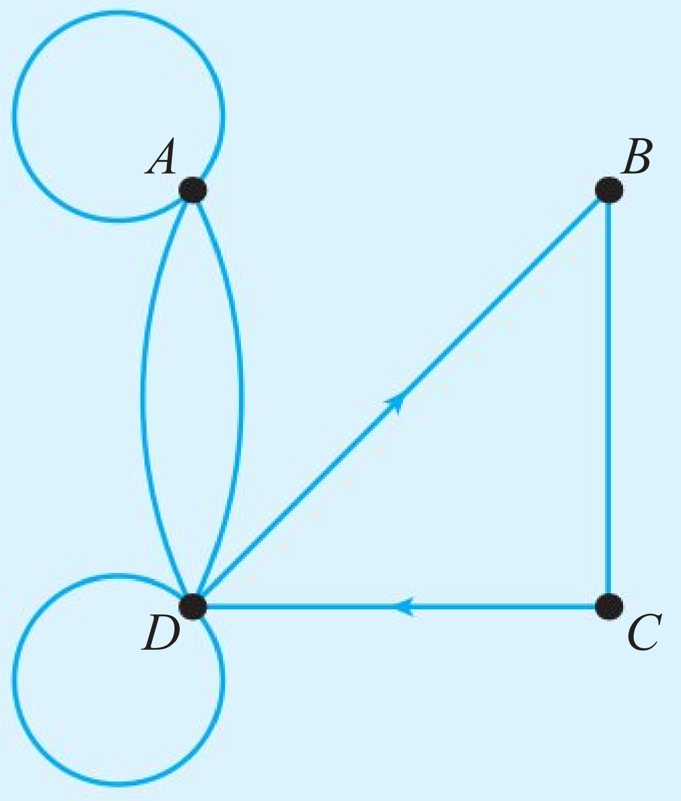
b



6 **a**



b



For questions 7 and 8, use the method demonstrated in Worked Example 7.3b to draw a graph with the given adjacency matrix.

7 **a**

$$\begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

b

$$\begin{array}{c} A \\ B \\ C \\ D \end{array} \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

8 **a**
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}2&1&0&1&2\\1&0&1&0&1\\0&1&0&2&1\\1&0&2&0&0\\2&1&1&0&2\end{pmatrix}$$

b
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}2&2&0&1&0\\2&0&1&0&1\\0&1&0&2&1\\1&0&2&0&0\\0&1&1&0&2\end{pmatrix}$$

For questions 9 and 10, use the method demonstrated in Worked Example 7.4 to answer the questions related to the graph represented by the given adjacency matrix.

9 **a**
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}0&1&0&1&0\\1&0&1&0&1\\0&1&0&0&1\\1&0&0&0&1\\0&1&1&1&0\end{pmatrix}$$

i List all the vertices adjacent to A.

ii List all the vertices adjacent to C.

iii State the degree of vertex B.

iv State the degree of vertex C.

b
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}0&1&0&1&1\\1&0&1&0&1\\0&1&0&1&1\\1&0&1&0&0\\1&1&1&0&0\end{pmatrix}$$

i List all the vertices adjacent to D.

ii List all the vertices adjacent to C.

iii State the degree of vertex D.

iv State the degree of vertex A.

10 **a**
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}0&3&0&1&0\\3&0&1&0&1\\0&1&0&2&1\\1&0&2&0&0\\0&1&1&0&0\end{pmatrix}$$

i List all the vertices adjacent to A.

ii List all the vertices adjacent to C.

iii State the degree of vertex D.

iv State the degree of vertex B.

b
$$\begin{array}{c}A\\B\\C\\D\\E\end{array}\begin{pmatrix}0&0&2&1&0\\0&0&1&1&1\\2&1&0&2&3\\1&1&2&0&0\\0&1&3&0&0\end{pmatrix}$$

i List all the vertices adjacent to B.

ii List all the vertices adjacent to D.

iii State the degree of vertex C.

iv State the degree of vertex A.

For questions 11 and 12, you are given an adjacency matrix for a directed graph. Use the method demonstrated in Worked Example 7.5 to state

i the out-degree of vertex C

ii the in-degree of vertex A.

11 **a**
$$\begin{array}{c}A\\B\\C\\D\end{array}\begin{pmatrix}0&1&0&0\\1&0&1&0\\1&0&0&1\\1&1&1&0\end{pmatrix}$$

b
$$\begin{array}{c}A\\B\\C\\D\end{array}\begin{pmatrix}0&1&0&1\\1&0&1&0\\0&1&0&1\\1&0&1&0\end{pmatrix}$$

12 **a**
$$\begin{array}{c}A\\B\\C\\D\end{array}\begin{pmatrix}0&1&0&2\\3&0&1&0\\2&1&0&1\\1&0&2&0\end{pmatrix}$$

b
$$\begin{array}{c}A\\B\\C\\D\end{array}\begin{pmatrix}0&2&1&1\\1&0&1&3\\0&1&0&1\\2&1&0&0\end{pmatrix}$$

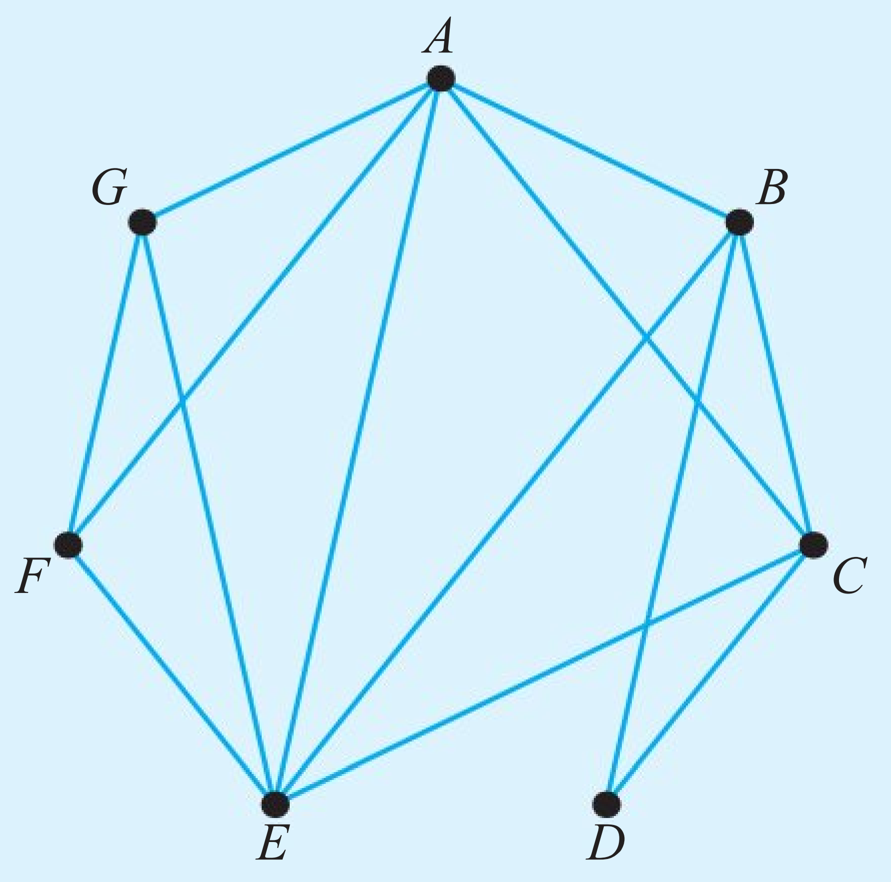
13 In the graph below, the vertices represent seven members of a social network and there is an edge between two vertices if they are friends.

a Write down an adjacency matrix for the graph.

b State the degree of vertex C, and interpret it in context.

c Interpret the meaning of the term ‘complete graph’ in this context.

d How many edges need to be added to make the above graph complete?

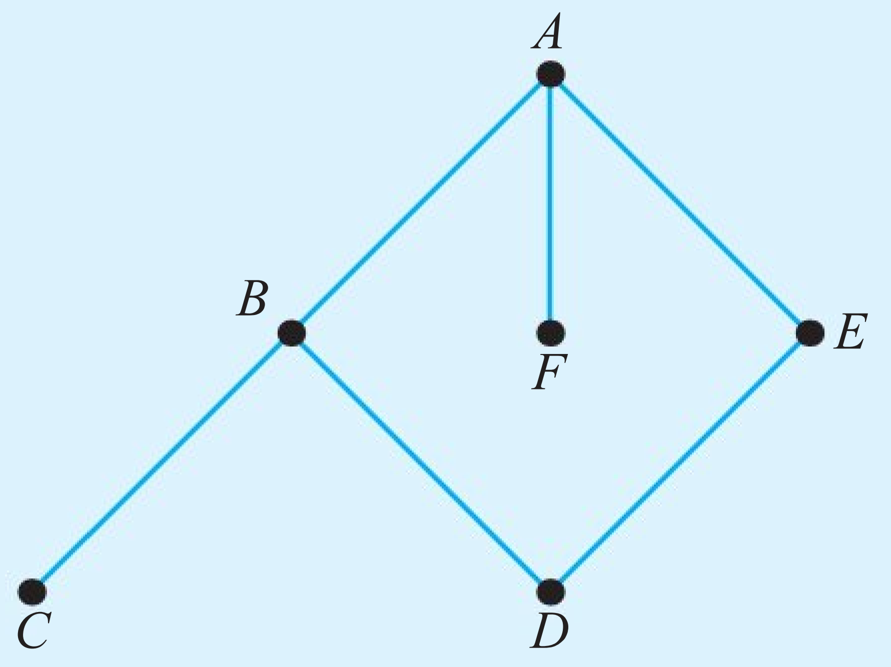


14 The graph shows power cables which connect a power source at A to devices B to F.

a Explain why the graph is not a tree.

b Explain why it is important for this graph to be connected.

c State one possible edge that can be removed for the graph to stay connected.



15 A network of one-way streets, connecting junctions A to E in a town, is represented by the following adjacency matrix:

$$\begin{array}{c} A \\ B \\ C \\ D \\ E \end{array}\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

a Draw the graph to represent this situation.

b Show that the graph is not strongly connected.

c Explain what that means in this context.

d The town council proposes to make one of the streets two-way in order to make the graph strongly connected. Which street should it be?

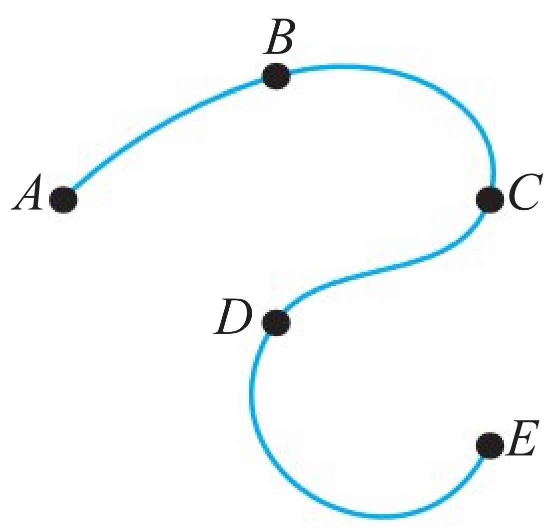
7B Moving around a graph

In many applications of graph theory we are interested in different ways of getting from one vertex to another. We may be required to return to the starting vertex, visit every vertex, or there may be restrictions on whether we are allowed to use each edge more than once. All these different ways of moving around a graph are given names.

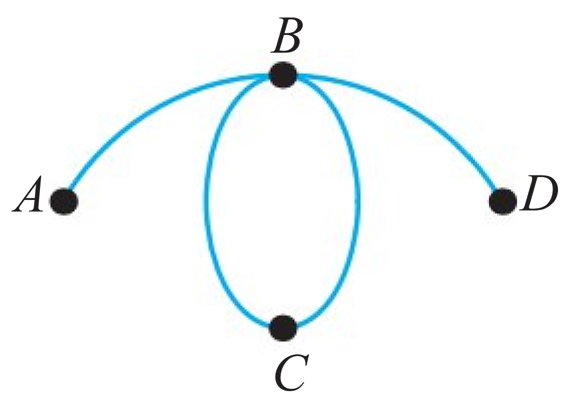
A **walk** is any sequence of adjacent edges.

A **trail** is a walk with no repeated edges.

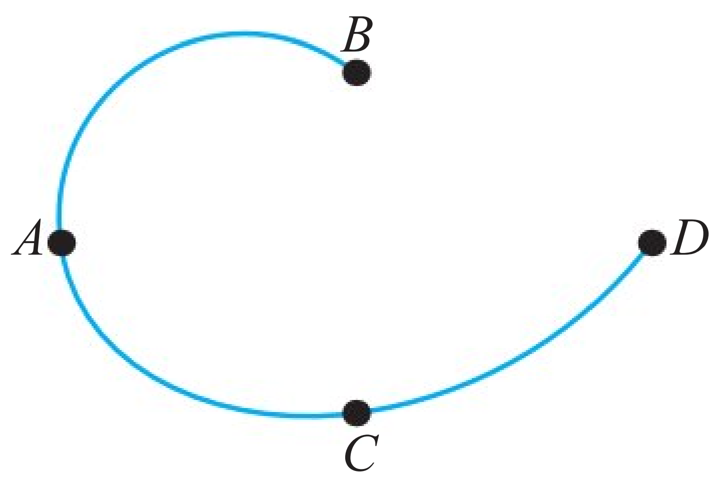
A **path** is a walk with no repeated vertices



■ *ABCDE* is a path (and therefore also a walk and a trail)



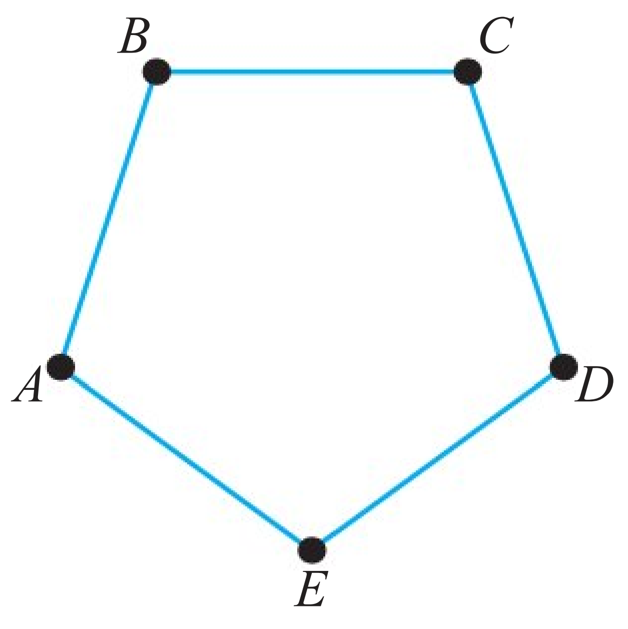
■ *ABCBD* is a trail (and therefore also a walk), but not a path



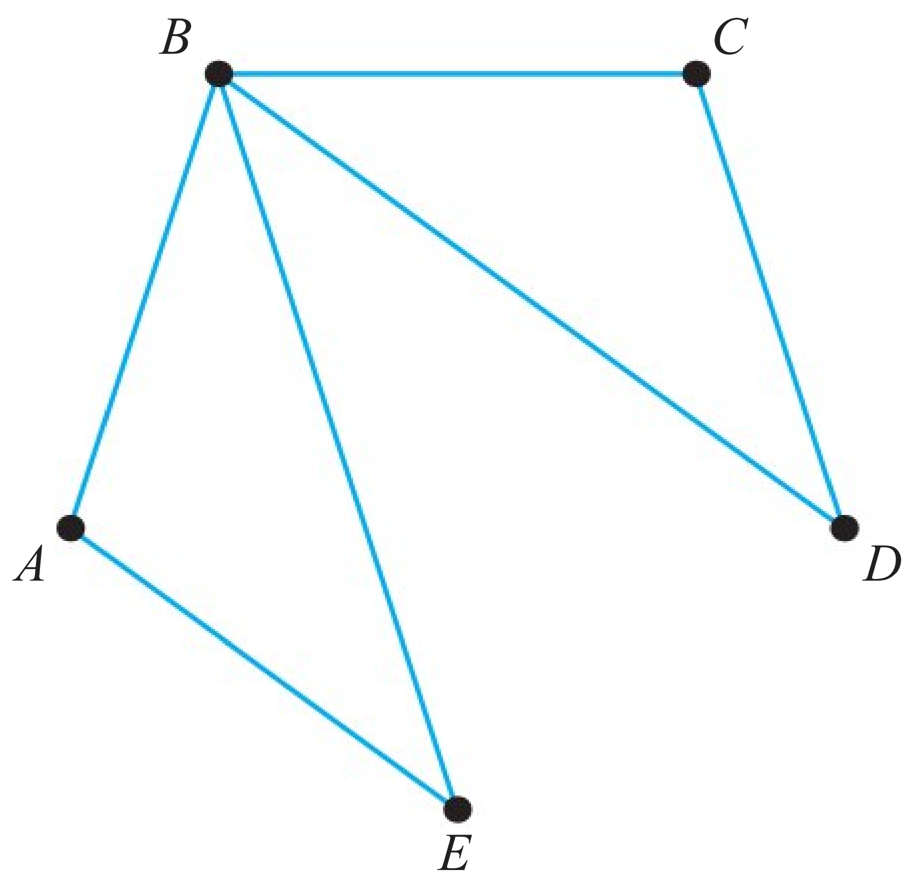
■ *ABACD* is a walk, but not a trail (and therefore also not a path)

A **cycle** is a walk that starts and ends at the same vertex and has no other repeated vertices (so it is a closed path). A tree has no cycles.

A **circuit** is a walk that starts and ends at the same vertex and has no repeated edges (so it is a closed trail).



■ *ABCDEA* is a cycle (and therefore also a circuit)



■ *ABCDBEA* is a circuit but not a cycle

The length of a walk is the number of its edges.

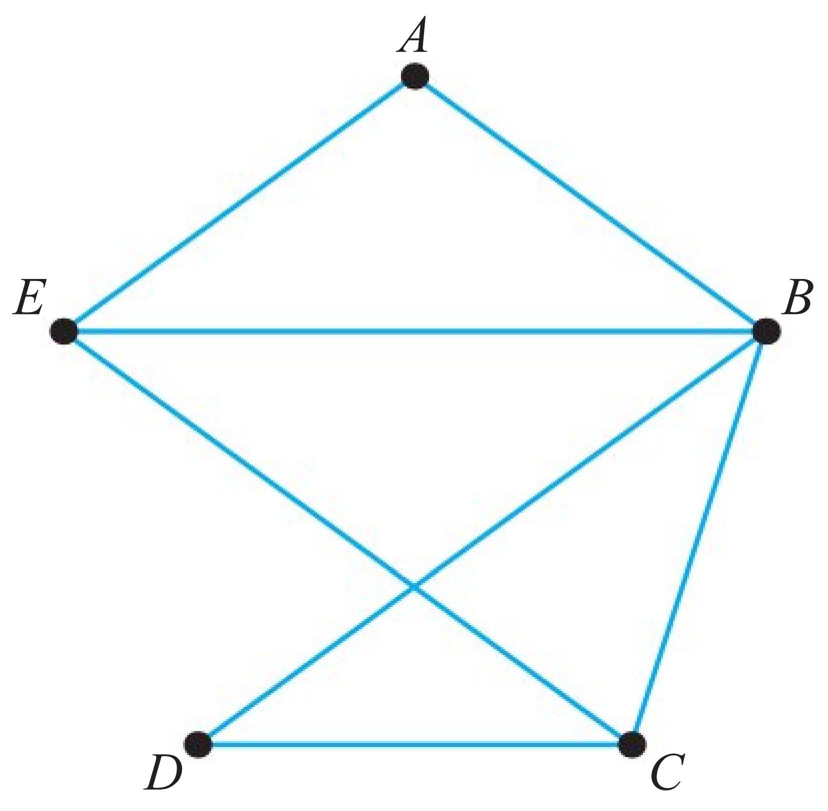
WORKED EXAMPLE 7.6

Consider this graph.

a List all paths of length 3 from E to C.

b List all the cycles of length 4 starting and ending at C.

c Find an example of a trail that uses each edge of the graph exactly once.



This can only be done by inspection. Think where we can go from E

A path of length 3 has four different vertices

a $EABC$, $EBDC$

A cycle of length 4 has four different vertices

Note that, in an undirected graph, $CBEAC$ is the same cycle as $CEABC$

b $CEABC$

Some vertices will appear more than once

c $EABECBDC$

Using an adjacency matrix to count walks

The entries in an adjacency matrix tell you the number of edges between two adjacent vertices. Do you think it is also possible to count the number of ways of moving between two vertices which are not adjacent? It turns out that this can be done using powers of the adjacency matrix.

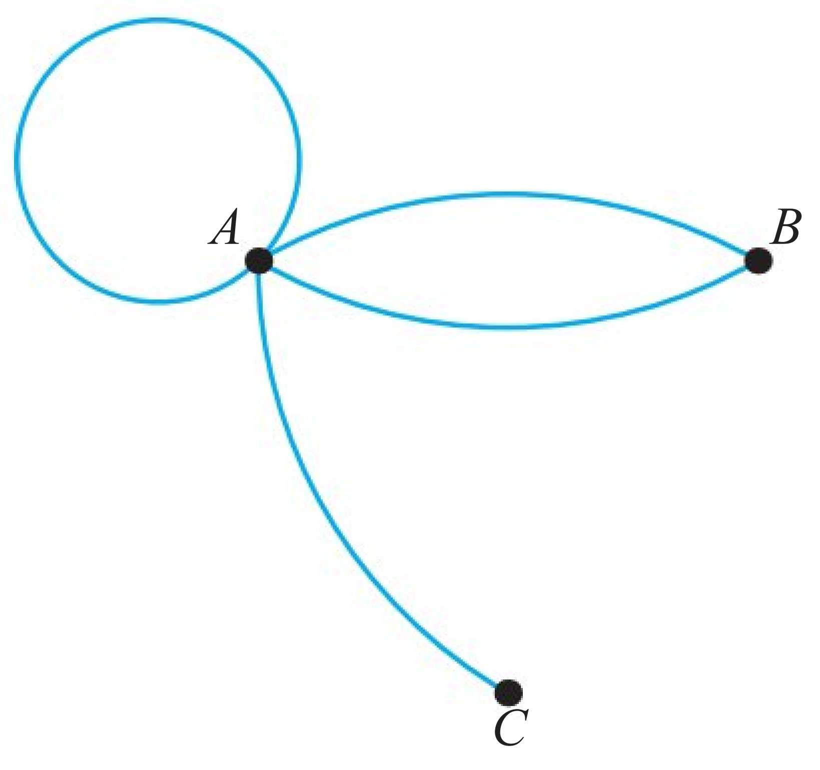
KEY POINT 7.3

For a graph with adjacency matrix A, the number of walks of length k between two vertices is given by the corresponding entry in the matrix $\mathbf{A}^k$.

There is one adjustment you may need to make to the adjacency matrix. If a vertex is connected to itself, there will be a ‘2’ on the diagonal of the matrix. In most contexts this needs to be changed to ‘1’ before raising the matrix to the required power. The only time when you would leave it as a ‘2’ would be if you wanted to treat clockwise and anticlockwise movements around the path as different walks.

WORKED EXAMPLE 7.7

Find the number of walks of length 5 between vertices A and B in this graph.



You need the fifth power of the adjacency matrix

The adjacency matrix is

Remember to write the entry in the top left corner as '1' rather than '2'

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

Use your GDC to evaluate the power of the matrix

Then

$$\mathbf{A}^5 = \begin{pmatrix} 96 & 82 & 41 \\ 82 & 44 & 22 \\ 41 & 22 & 11 \end{pmatrix}$$

The number of walks between A and B is the entry in the first row and second column

The required number of walks is 82.

Tip

You may want to list all the walks of length 2 from A to A and compare their number to the relevant entry in $\mathbf{A}^2$.

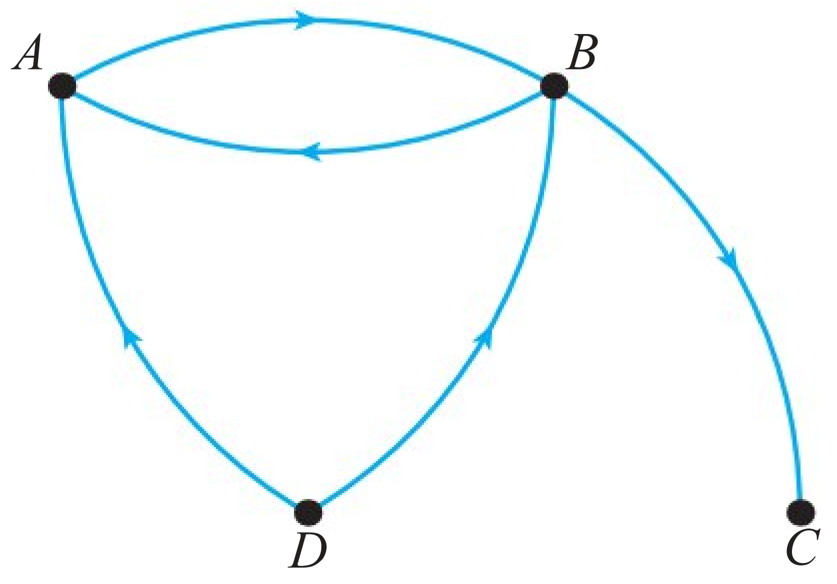
The result also applies to directed graphs.

WORKED EXAMPLE 7.8

For the graph shown, find the number of walks of length less than 4

a from A to D

b from D to A.



First write down the adjacency matrix. The rows are the 'from' vertices and the columns 'to' vertices

$$\mathbf{A}=\begin{pmatrix}0&1&0&0\\1&0&1&0\\0&0&0&0\\1&1&0&0\end{pmatrix}$$

You need the number of walks of length 1, 2 and 3. Therefore, you need to look at $\mathbf{A}$, $\mathbf{A}^2$ and $\mathbf{A}^3$

$$\mathbf{A}^2=\begin{pmatrix}1&0&1&0\\0&1&0&0\\0&0&0&0\\1&1&1&0\end{pmatrix},\ \mathbf{A}^3=\begin{pmatrix}0&1&0&0\\1&0&1&0\\0&0&0&0\\1&1&1&0\end{pmatrix}$$

Walks from A to D are given by the first row and fourth column

a From A to D: $0+0+0=0$

Walks from B to A are given by the second row and first column

b From B to A: $1+1+1=3$

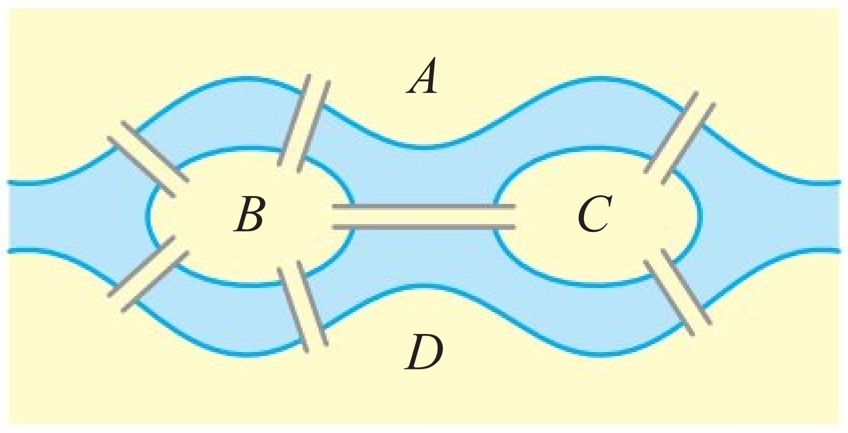
Tip

You can see the number of all walks of length less than 4 by considering the matrix $\mathbf{A}+\mathbf{A}^2+\mathbf{A}^3$.

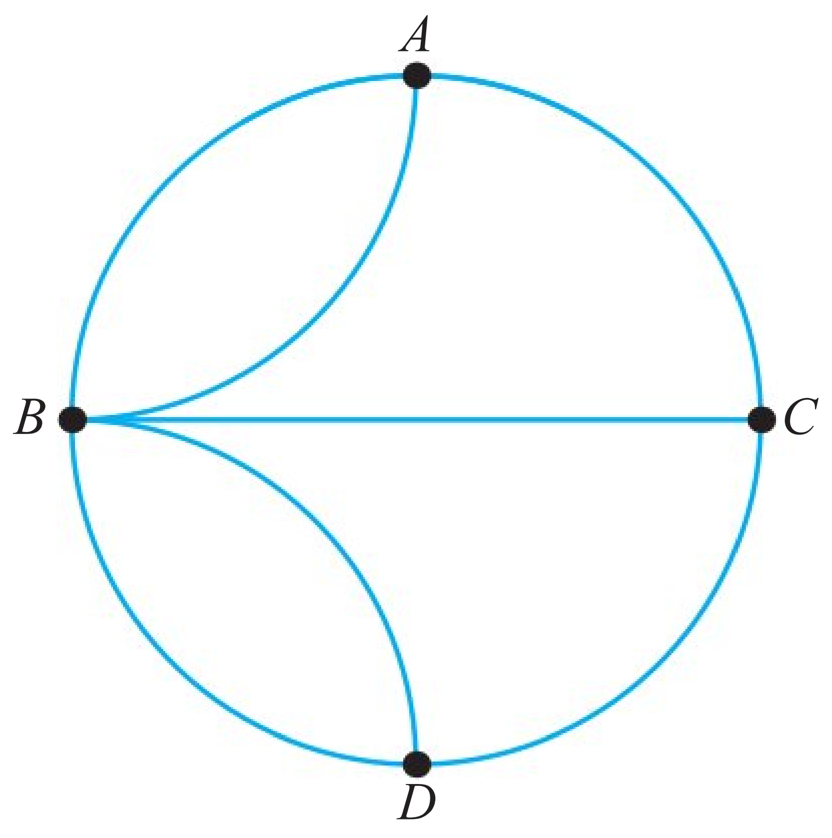
Eulerian trails and circuits

This problem was first studied by Leonhard Euler in 1836.

The following is the Königsberg bridges problem, considered to be the first published problem in graph theory.



Two islands are connected to each other, and to the two river banks, by bridges shown in the diagram. Is it possible to have a walk which crosses each bridge exactly once and returns to the starting point?



The problem is equivalent to trying to find a circuit which includes every edge of the graph shown. It turns out that, in this case, this is not possible.

You already looked at some similar problems in the Starter Activity.

A circuit which includes every edge of the graph exactly once is called an **Eulerian circuit**. A graph which has an Eulerian circuit is called an **Eulerian graph**. So the graph in the example above is not Eulerian. There is a simple way to check whether a graph is Eulerian.

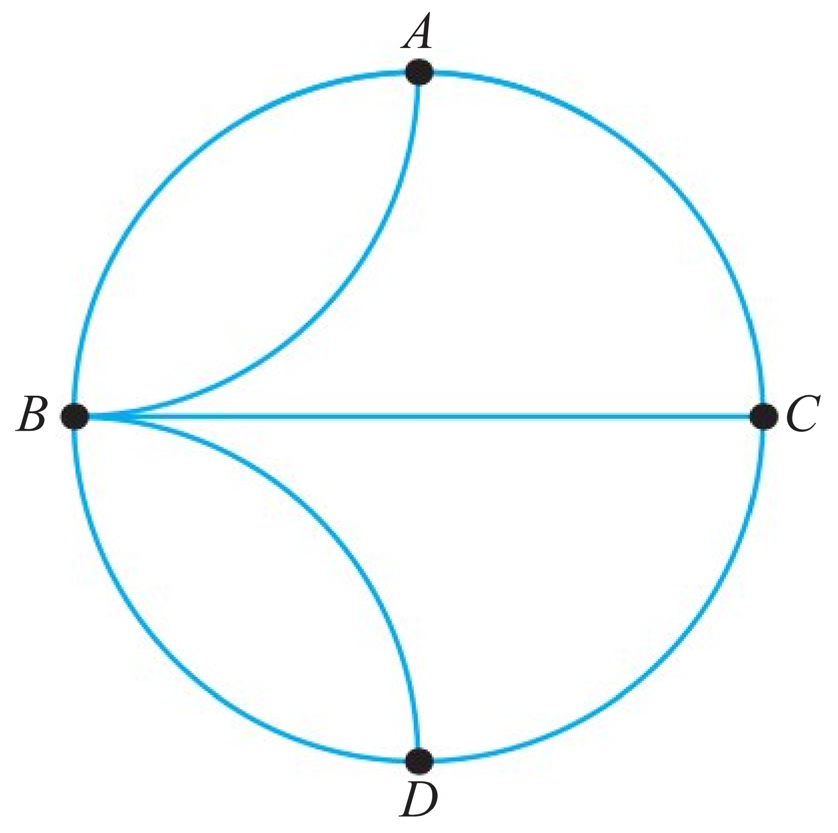
KEY POINT 7.4

In an Eulerian graph every vertex has even degree.

This result can be explained as follows: in an Eulerian circuit, every time we visit a vertex we must go in and out of it, which uses up two edges. This also applies to the starting vertex, as we return to it at the end. So the number of edges at each vertex must be even.

WORKED EXAMPLE 7.9

Show that the graph in the Königsberg bridges problem is not Eulerian.



Determine the degree of each vertex — Degrees of vertices: 3, 5, 3, 3

In an Eulerian graph all vertices have even degree — All vertices have odd degree, so the graph is not Eulerian.

When an Eulerian circuit does not exist, it may still be possible to find a walk which uses each edge exactly once (without returning to the starting point). This is called an **Eulerian trail** and the graph is then called **semi-Eulerian**. The argument about going 'in and out' of each vertex still applies to all vertices except for the start and end ones.

KEY POINT 7.5

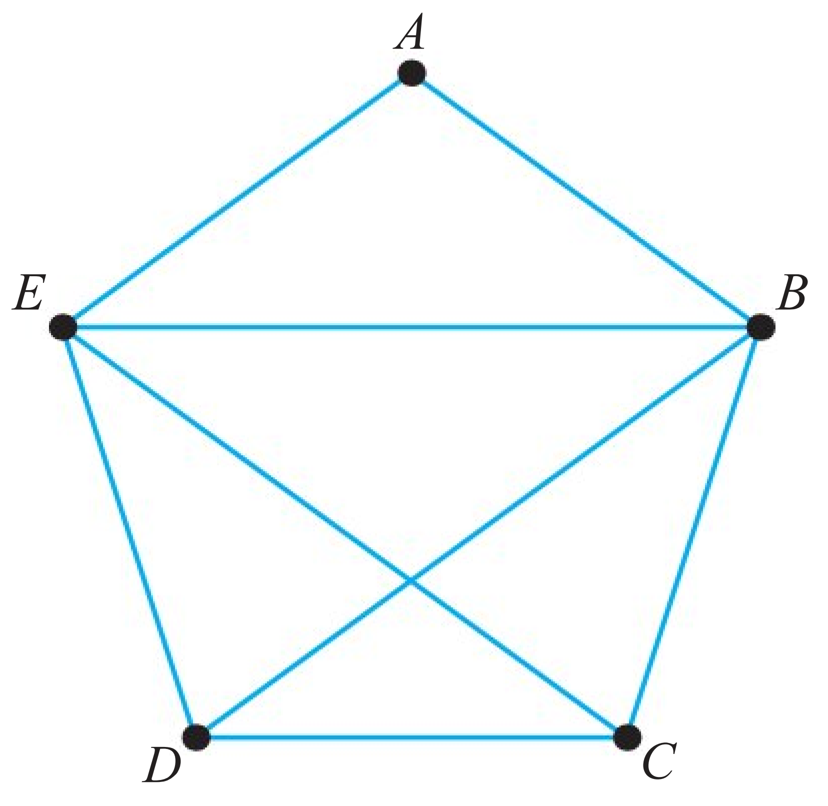
A semi-Eulerian graph has exactly two vertices of odd degree. Every Eulerian trail starts at one of the odd vertices and ends at the other.

Tip

Finding an Eulerian trail or circuit is equivalent to drawing the graph without picking up your pen and without going over any edge more than once. You can use this to check that your trail or circuit is correct.

WORKED EXAMPLE 7.10

Show that the following graph is semi-Eulerian and find an Eulerian trail.



You need the degrees of all the vertices

Degrees of the vertices:

$$A = 2,\ B = 4,\ C = 3,\ D = 3,\ E = 4$$

There are exactly two vertices of odd degree, so the graph is semi-Eulerian.

The Eulerian trail starts and ends at the odd vertices, so you can start at C and end at D

Eulerian trail:

CDECBAEBD

Note that each time you pass through a vertex, you 'use up' two edges. So the trail should visit vertex A once and all the other vertices twice

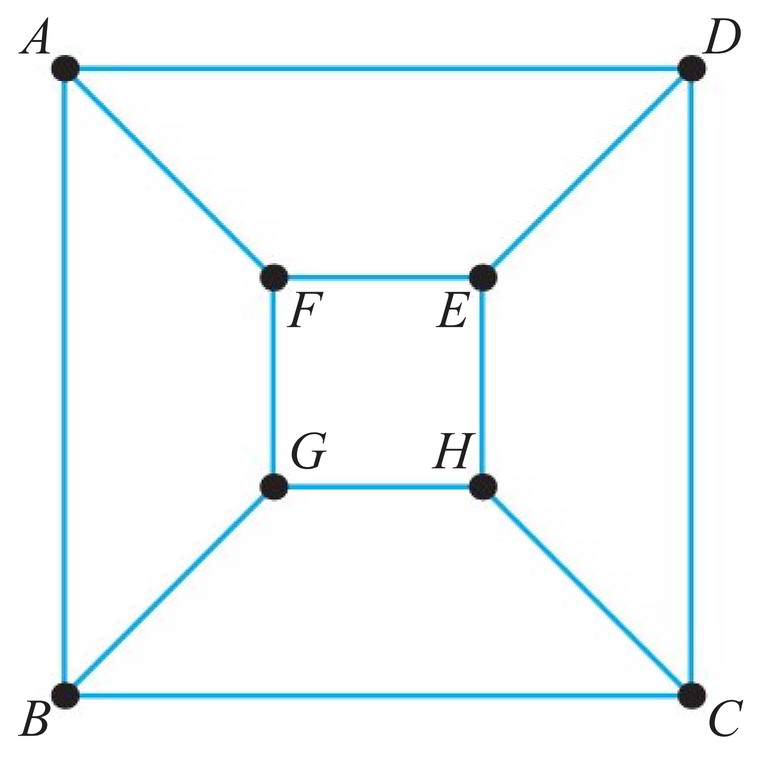
Eulerian circuits and trails will be used when solving the Chinese postman problem in Section 7E.

Hamiltonian paths and cycles

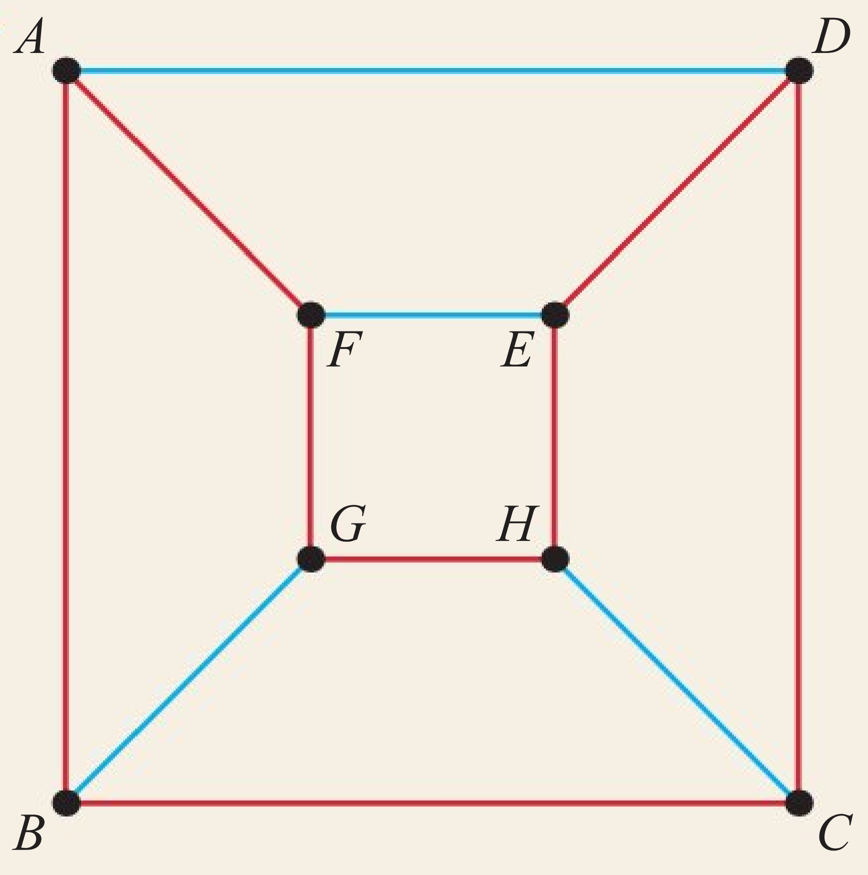
A **Hamiltonian path** in a graph visits each vertex exactly once. A **Hamiltonian cycle** visits each vertex exactly once and returns to the starting vertex. Not every graph has a Hamiltonian cycle; if it does it is called a **Hamiltonian graph**.

WORKED EXAMPLE 7.11

Show that this graph is Hamiltonian.



Try to find a Hamiltonian cycle: can you visit each vertex without repeating any edges?



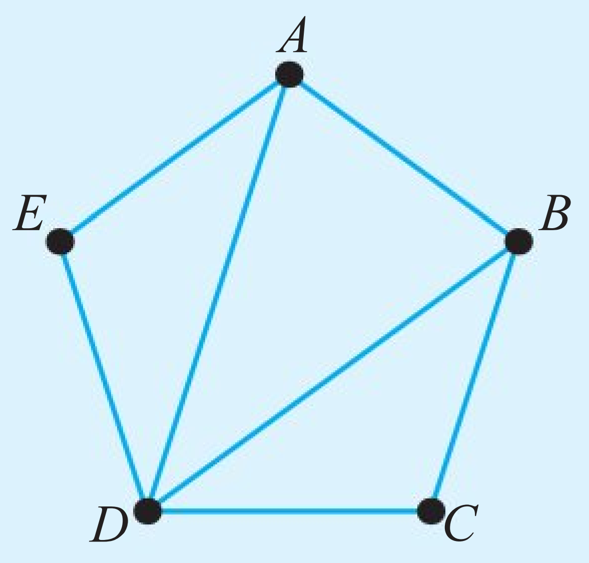
Hamiltonian cycle: *ABCDEHGFA*
Hence, the graph is Hamiltonian.

There is no quick way to identify whether a graph is Hamiltonian. This is what makes some problems with graphs very difficult to solve.

Hamiltonian cycles will be used in solving the travelling salesman problem in Section 7F.

Exercise 7B

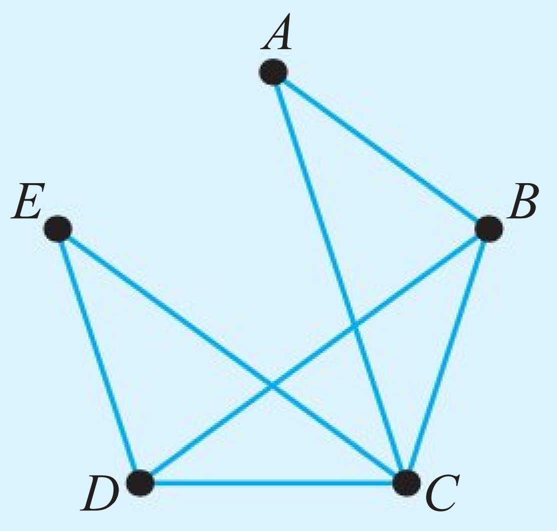
Questions 1 and 2 refer to the graph shown alongside. Use the method demonstrated in Worked Example 7.6.



1 Find all the paths of length 3
 a from A to B
 b from C to E.

2 Find all the cycles of length 4 starting and ending at
 a A
 b C.

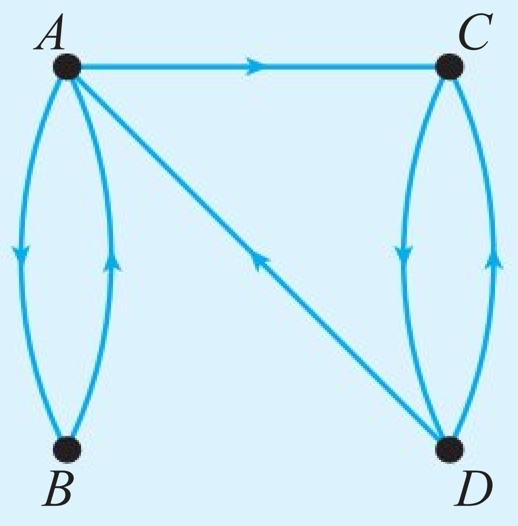
Questions 3 and 4 refer the graph shown alongside. Use the method demonstrated in Worked Example 7.7.



3 Find the number of walks of length 3 between
 a A and C
 b B and C.

4 Find the number of walks of length 4 between
 a A and D
 b C and D.

Questions 5 and 6 refer the graph shown alongside. Use the method demonstrated in Worked Example 7.8.

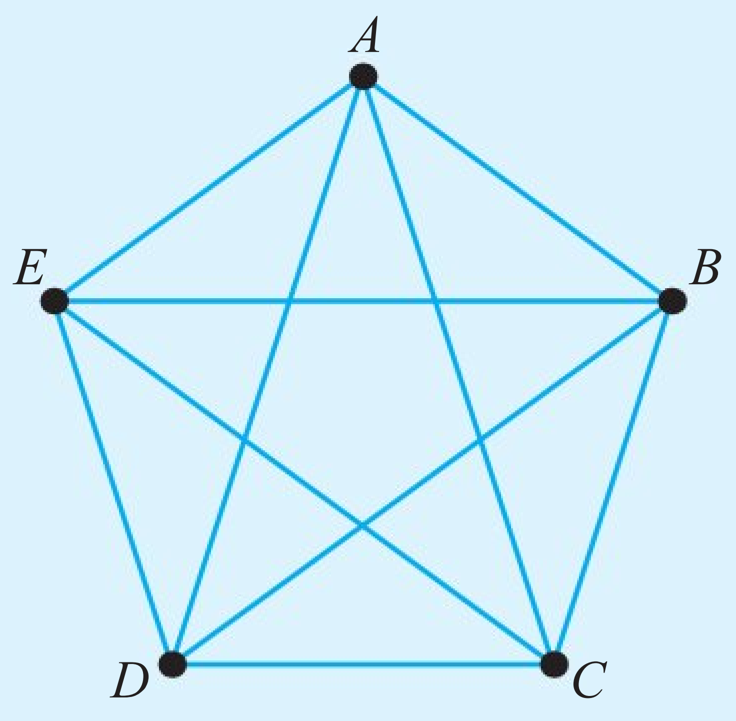


5 Find the number of walks of length 3
 a from A to D
 b from B to D.

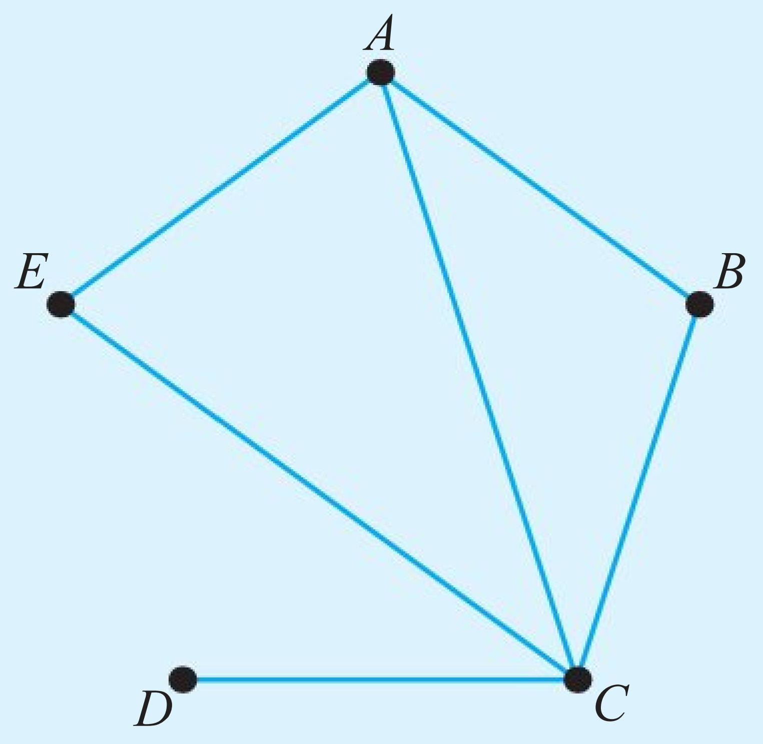
6 Find the number of walks of length 4 between
 a from A to C
 b from C to D.

For questions 7 and 8, use the methods demonstrated in Worked Examples 7.9 and 7.10 to determine whether each graph is Eulerian or semi-Eulerian. If it is, find an Eulerian cycle or an Eulerian trail.

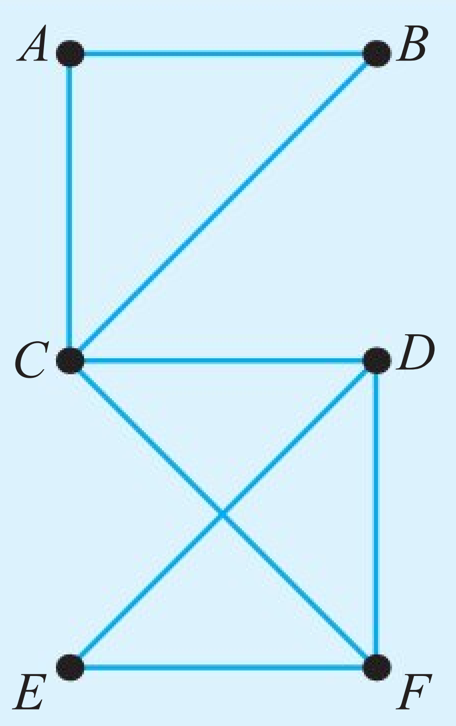
7 a



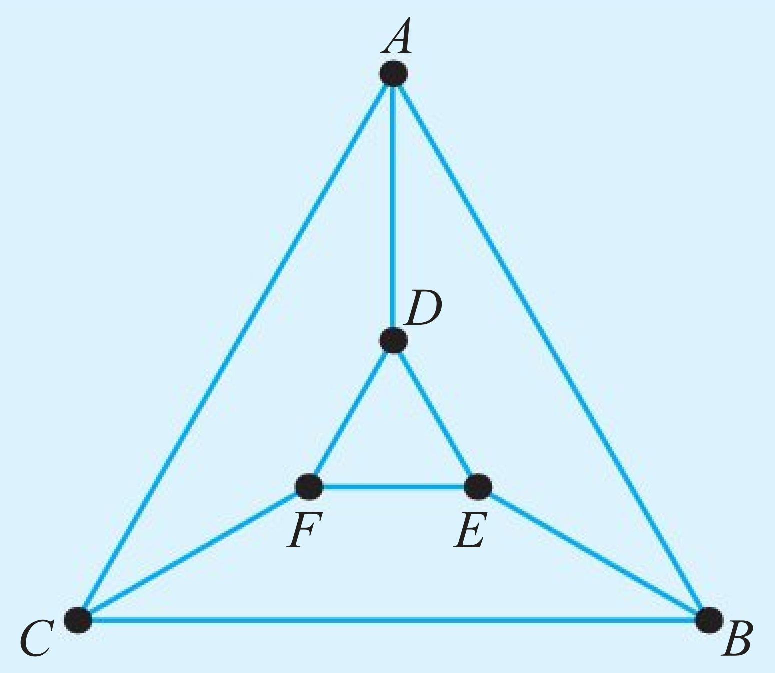
b



8 **a**

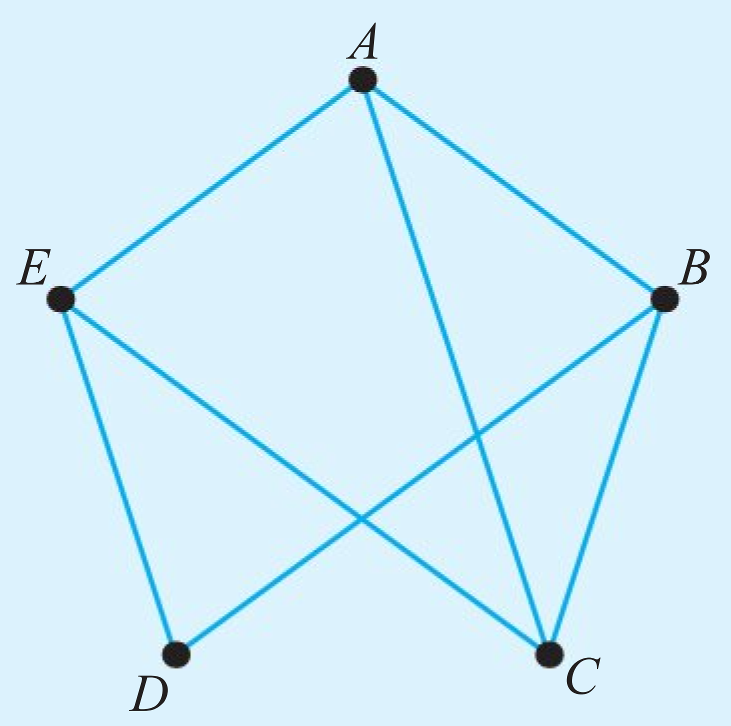


b

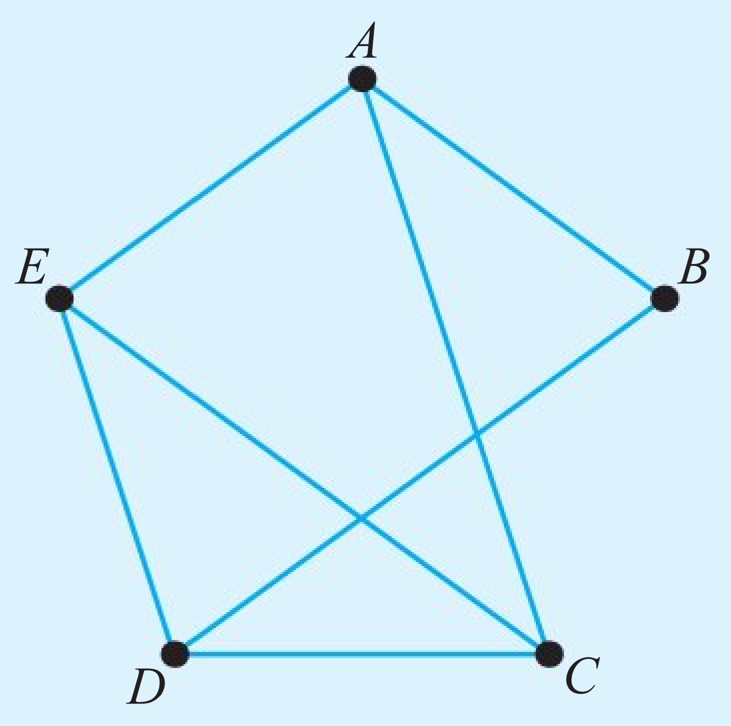


For questions 9 and 10, use the method demonstrated in Worked Example 7.11 to find a Hamiltonian cycle in each of the following graphs.

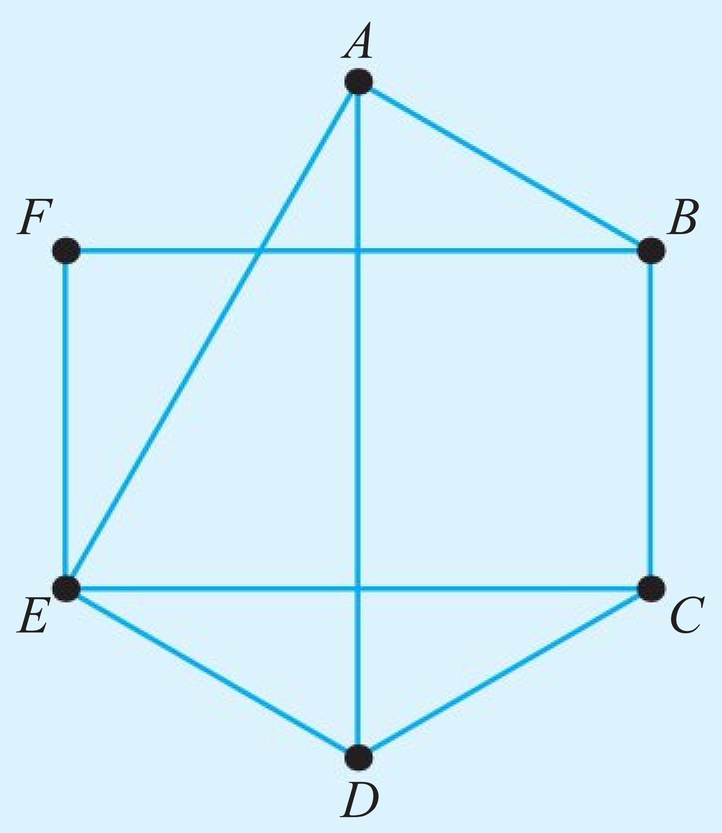
9 **a**



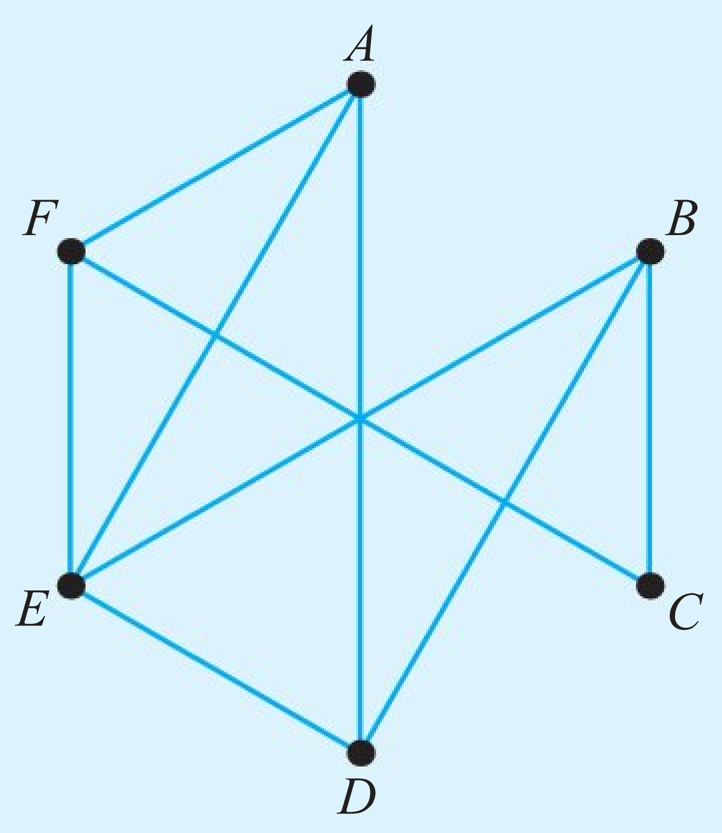
b



10 **a**

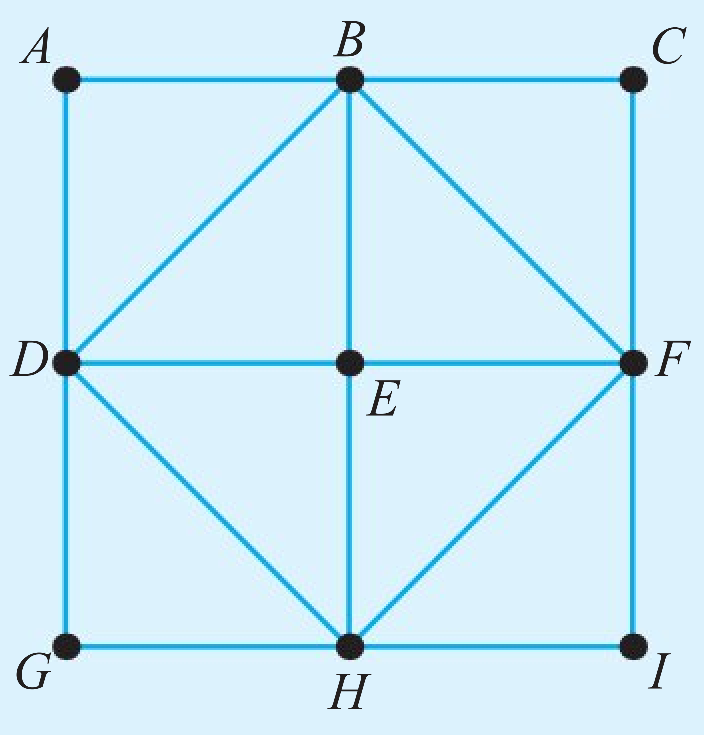


b

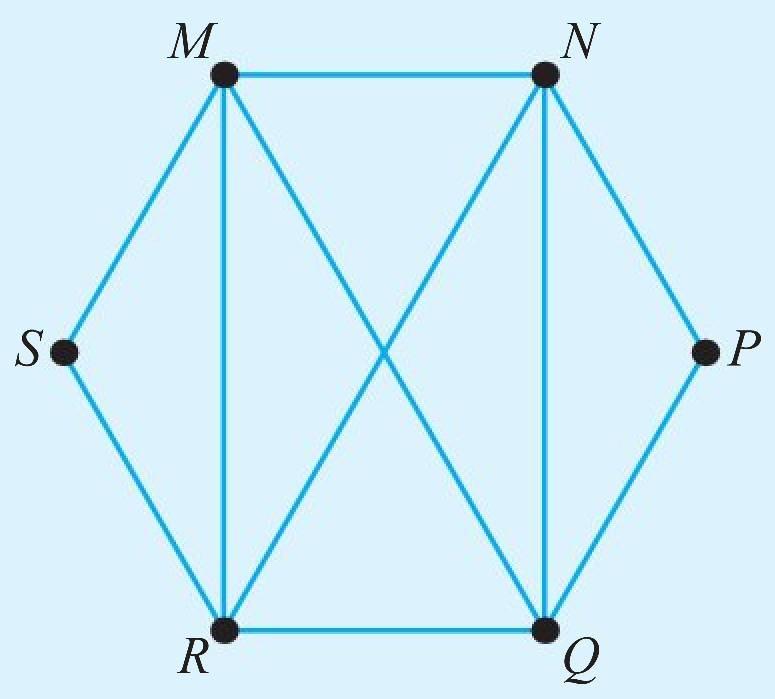


11 Two graphs, G and H, are shown below.

Graph G



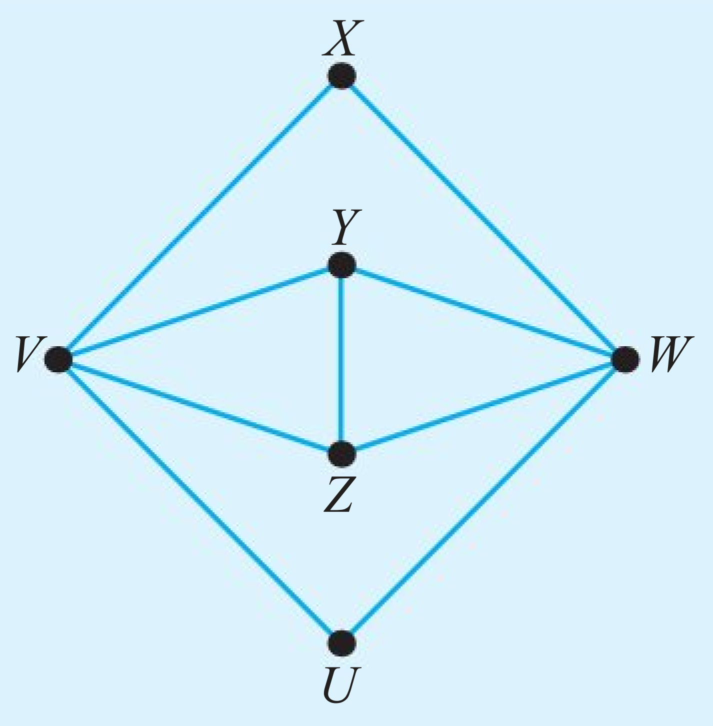
Graph H



a Explain why G is not Eulerian.

b Find an Eulerian circuit in H.

12 Explain why the graph shown alongside is semi-Eulerian, and find an Eulerian trail.



13 **a** Which of these three adjacency matrices represent Eulerian graphs?

$$\boldsymbol{A}=\begin{pmatrix}0&1&0&1&1\\1&0&0&0&1\\0&0&0&0&1\\1&0&0&0&1\\1&1&1&1&0\end{pmatrix},\ \boldsymbol{B}=\begin{pmatrix}0&0&1&1\\0&0&1&1\\1&1&0&0\\1&1&0&0\end{pmatrix},\ \boldsymbol{C}=\begin{pmatrix}0&1&1&0&0&0\\1&0&1&1&1&0\\1&1&0&1&0&1\\0&1&1&0&0&0\\0&1&0&0&0&1\\0&0&1&0&1&0\end{pmatrix}$$

b Draw the graph represented by the matrix B. Label the vertices X, Y, Z and W, so that the first column of the matrix corresponds to the vertex X and so on.

c For the graph represented by the matrix B, find the number of walks of length 5 between X and Z.

14 A graph is represented by the following adjacency matrix. The vertices are labelled A to E.

$$\begin{pmatrix}0&1&1&0&1\\1&0&0&1&1\\1&0&0&1&1\\0&1&1&0&1\\1&1&1&1&0\end{pmatrix}$$

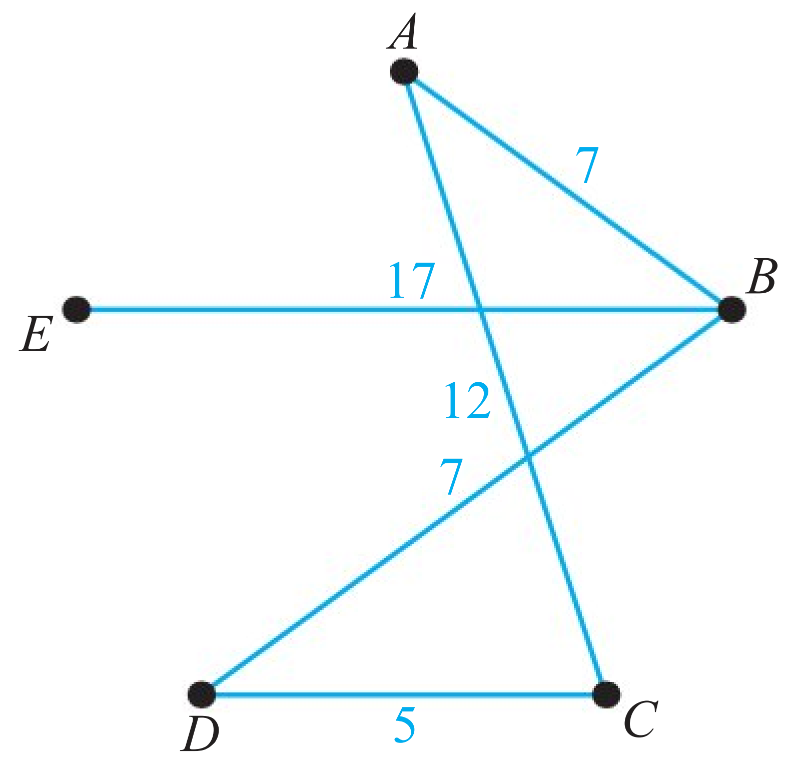
a Show that the graph is Hamiltonian.

b Find the number of walks of length 5 from A to E.

7C Weighted graphs

In a **weighted graph** each edge has a number associated with it. This number is called the **weight**, and can represent the distance, time or cost of travel between the two vertices. Note that this is not the actual length of the edge, and that there is no need to try and draw the graph to scale. Also remember that not all intersections of edges are vertices of the graph.

For example, the graph alongside could represent a road network and the numbers could be times (in minutes) taken to travel between different junctions. So it takes 17 minutes to travel between junctions B and E. To get from B to C to you can go either via A (which takes 19 minutes) or via D (which takes 12 minutes).



When a graph is large, it is not convenient to draw it out. Instead, we can represent it using a **weighted adjacency table**. The numbers in the table represent the weights of the edges. The table shown here represents the above graph.

Weighted graphs can also be directed, and the weight of an edge from *A* to *B* can be different from the weight of an edge from *B* to *A*.

	A	*B*	*C*	*D*	*E*
A	–	7	12	–	–
B	7	–	–	7	17
C	12	–	–	5	–
D	–	7	5	–	–
E	–	17	–	–	–

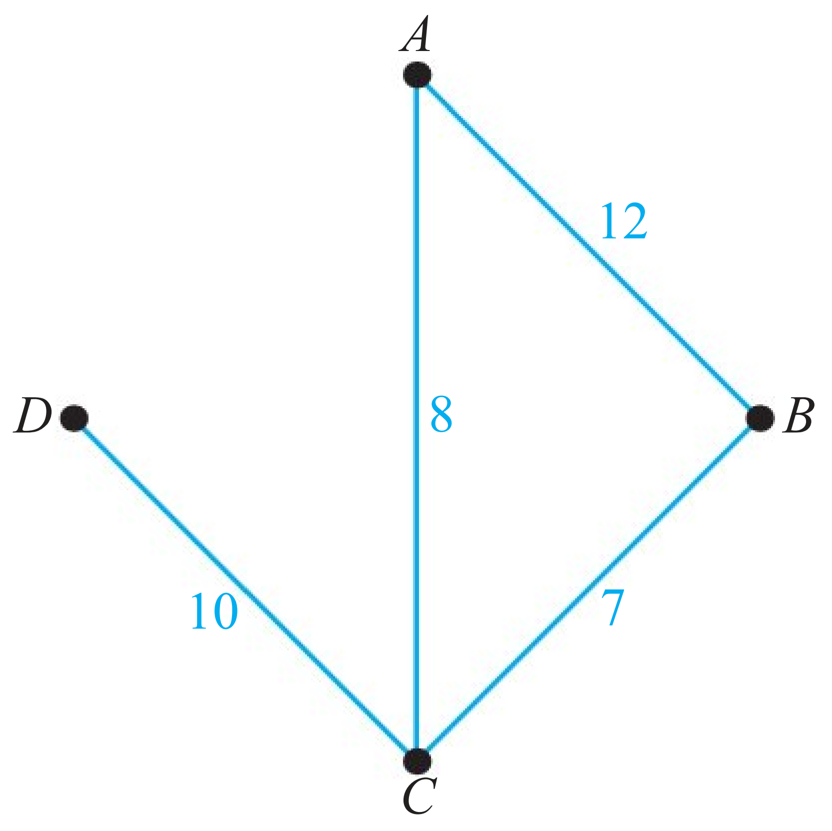
TOOLKIT: Modelling

Think of situations that could be modelled using weighted graphs. For each situation:

- Decide whether a directed or an undirected graph would be more appropriate.
- Discuss the modelling assumptions and any factors that are ignored when representing it as a graph.
- Come up with practical questions that could be answered using your graph. Can you also think of some questions for which your graph would not be the most useful model?

WORKED EXAMPLE 7.12

a Construct a weighted adjacency table for this graph.



b Draw the graph with the following weighted adjacency table.

	A	*B*	*C*	*D*
A	–	4	–	6
B	6	–	5	7
C	–	5	–	3
D	–	–	5	–

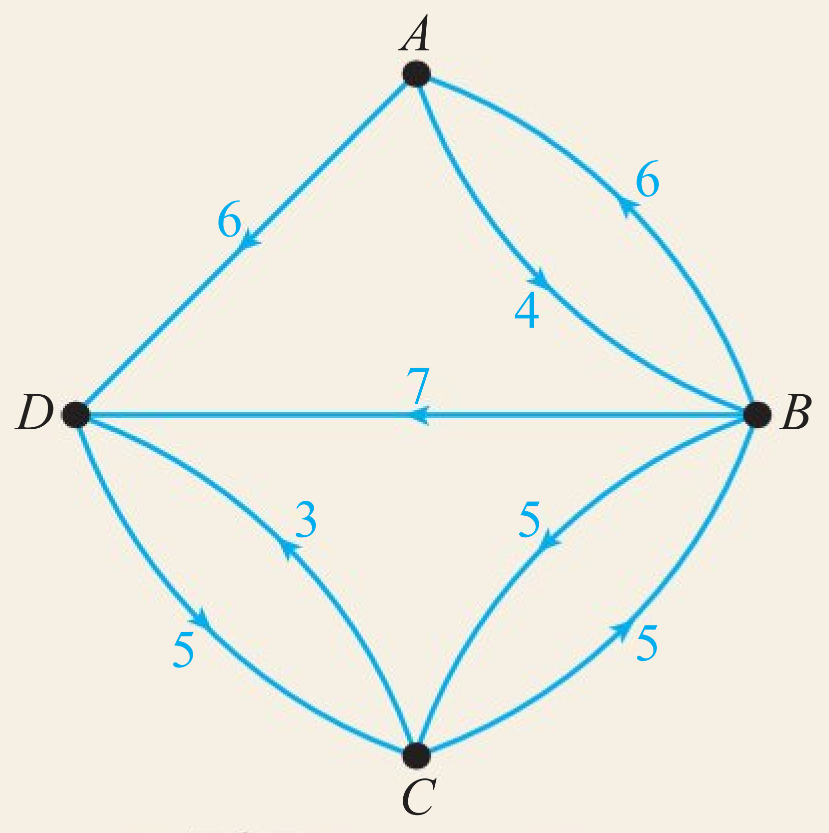
The graph is undirected, so the matrix should be symmetrical

a

	A	*B*	*C*	*D*
A	–	12	8	–
B	12	–	7	–
C	8	7	–	10
D	–	–	10	–

Be careful to get the directions correct. For example, the edge from *A* to *B* has weight 4, but the edge from *B* to *A* has weight 6

b



Transition matrices

Suppose you are moving around a graph so that, whenever you arrive at a vertex, you choose at random one of the possible edges to take next. This is called a **random** walk on the graph. You can represent this situation by assigning to each edge a weight equal to the probability of selecting it; this will be the reciprocal of the degree of the starting vertex. The resulting weighted graph will be directed. The **transition matrix** shows the probabilities of moving from one vertex to another.

Tip

Unlike an adjacency matrix, a transition matrix has the 'from' vertices listed along the top.

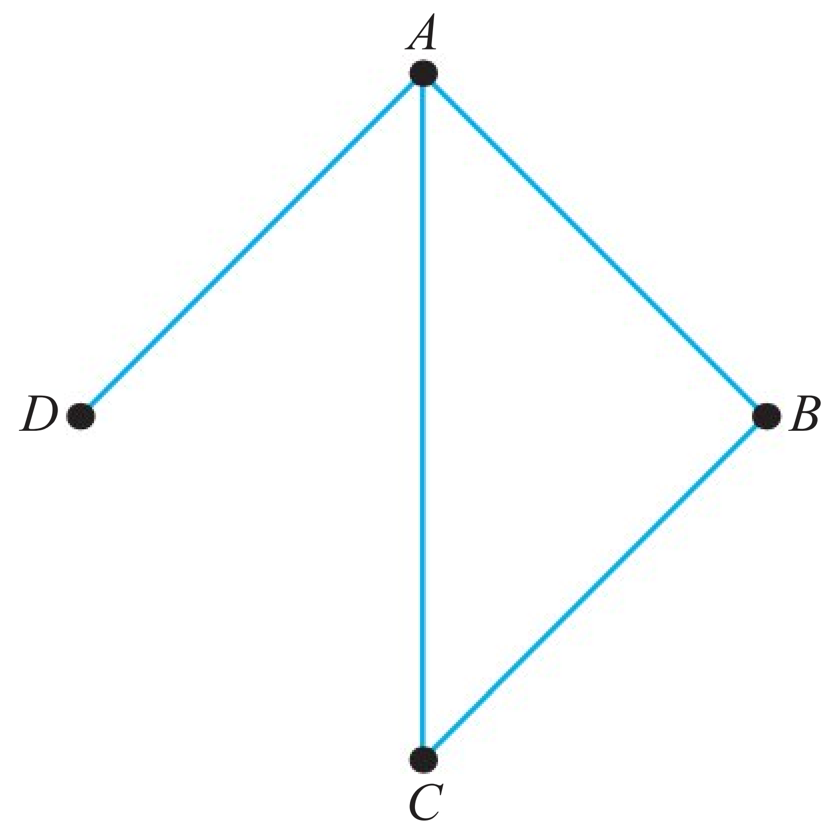
KEY POINT 7.6

In a transition matrix:

- the probability of moving from vertex *A* to vertex *B* is given by the entry in column *A* and row *B*
- the entries in each column must add up to 1.

WORKED EXAMPLE 7.13

Write down the transition matrix for the following graph.



All diagonal entries are zero (no vertex is connected to itself)

From A, there is one edge to each of the other vertices, so the remaining three entries in column A are all $\frac{1}{3}$

The non-zero entries in columns B and C are $\frac{1}{2}$, as those vertices have degree 2

The only edge from D is DA, so the first entry in the D column is 1

$$\begin{array}{cccc} A & B & C & D \\ \end{array}$$
$$\begin{pmatrix} 0 & 1/2 & 1/2 & 1 \\ 1/3 & 0 & 1/2 & 0 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \end{pmatrix}$$

You will learn in Section 8D how transition matrices can be used in a variety of other situations, and how the long-term behaviour is determined by the eigenvalues and eigenvectors of the matrix.

Raising a transition matrix to a power gives the probabilities of being at various vertices after a certain number of moves. For example, for the graph from Worked Example 7.13,

$$\mathbf{M}^3 = \begin{pmatrix} 1/6 & 11/24 & 11/24 & 2/3 \\ 11/36 & 1/6 & 7/24 & 1/6 \\ 11/36 & 7/24 & 1/6 & 1/6 \\ 2/9 & 1/12 & 1/12 & 0 \end{pmatrix}$$

If you start at vertex A, the probability of being at B after three moves is $\frac{11}{36}$ (the number highlighted in the matrix). If the power is very large, the probabilities are the proportion of time spent at each vertex if you continue to move around the graph for a long time. You treat the time taken to move between vertices as negligible.

TOOLKIT: Problem Solving

You could use a tree diagram to find the probability of getting from A to B after three moves. Reflect whether the tree diagram or the matrix method would be more efficient for a larger number of moves.

WORKED EXAMPLE 7.14

This is the transition matrix for the graph from the previous worked example.

$$\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 1 \\ 1/3 & 0 & 1/2 & 0 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

If a random walk is performed on this graph, find the percentage of time spent at vertex B. Ignore the transition time between vertices.

Look at a high power of the matrix – it should be high enough that the numbers don't change to three decimal places when you go up another power

$$\mathbf{M}^{40} = \begin{pmatrix} 0.375 & 0.375 & 0.375 & 0.375 \\ 0.250 & 0.250 & 0.250 & 0.250 \\ 0.250 & 0.250 & 0.250 & 0.250 \\ 0.125 & 0.125 & 0.125 & 0.125 \end{pmatrix}$$

Look at the row corresponding to vertex B – all the entries are 0.250

The random walk will spend 25% of the time at vertex B.

TOOLKIT: Problem Solving

In the above example, raising the transition matrix to a high power resulted in all four columns being identical. This means that the long-term proportion of time spent on each vertex does not depend on the starting point. Will this always be the case?

Investigate large powers of transition matrices of connected and unconnected graphs. For directed graphs, investigate the difference between graphs which are strongly connected and those which are not.

The PageRank algorithm

Google's PageRank algorithm ranks the pages in an internet search based on the number of links to them from other pages, but also taking into account how well linked those pages are. So a page receives a high ranking if it has links from many other pages, but also if it has links from a few high-ranking pages.

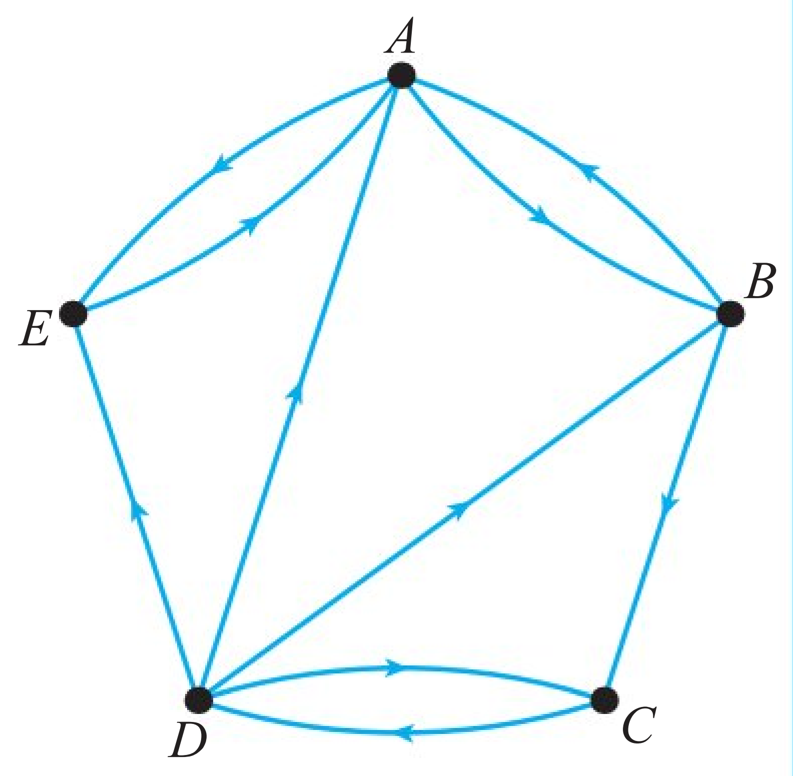
A simplified version of the algorithm assigns each page a ranking according to the probability that a person randomly clicking on links would end up on that page. If links between pages are represented by a directed graph, then the transition matrix can be used to determine the proportion that a random walk spends at each vertex.

The PageRank algorithm was developed by Larry Page and Sergey Brin in 1996 and has since formed the basis of all Google's search algorithms. More sophisticated versions of the algorithm take into account whether any of the sites are from a particularly authoritative source, as well as the probability of the user abandoning the search.

WORKED EXAMPLE 7.15

The graph shows the links between five web pages.

Use a transition matrix to determine which page should come top in a search.



Write down the transition matrix for the graph

The transition matrix is:

$$\mathbf{M} = \begin{pmatrix} 0 & 0.5 & 0 & 0.25 & 1 \\ 0.5 & 0 & 0 & 0.25 & 0 \\ 0 & 0.5 & 0 & 0.25 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0 & 0.25 & 0 \end{pmatrix}$$

Raise the matrix to a large power

In $\mathbf{M}^n$, each column is

$$\begin{pmatrix} 0.333 \\ 0.2 \\ 0.133 \\ 0.133 \\ 0.2 \end{pmatrix}$$

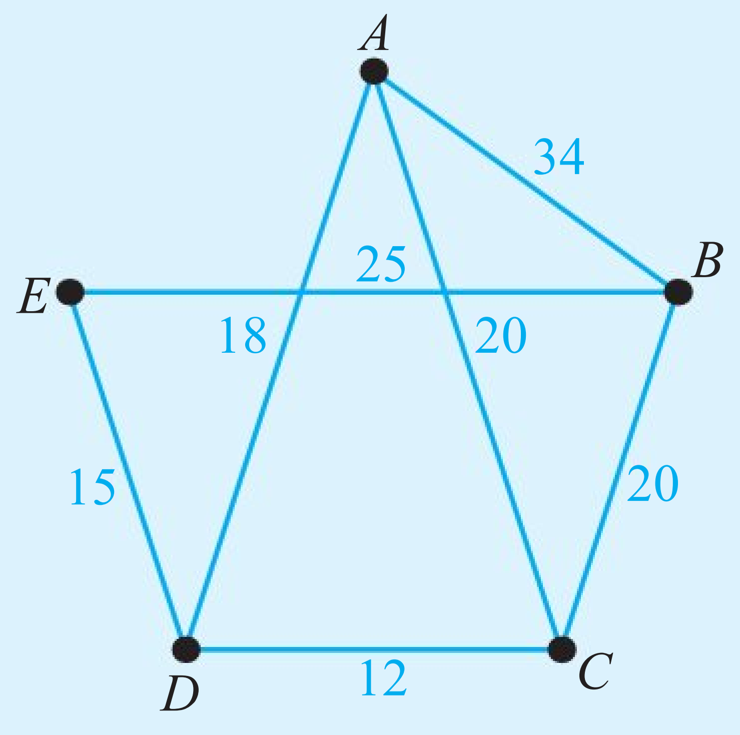
The vertex with the largest entry should come at the top of the search

Page A should come at the top of the search.

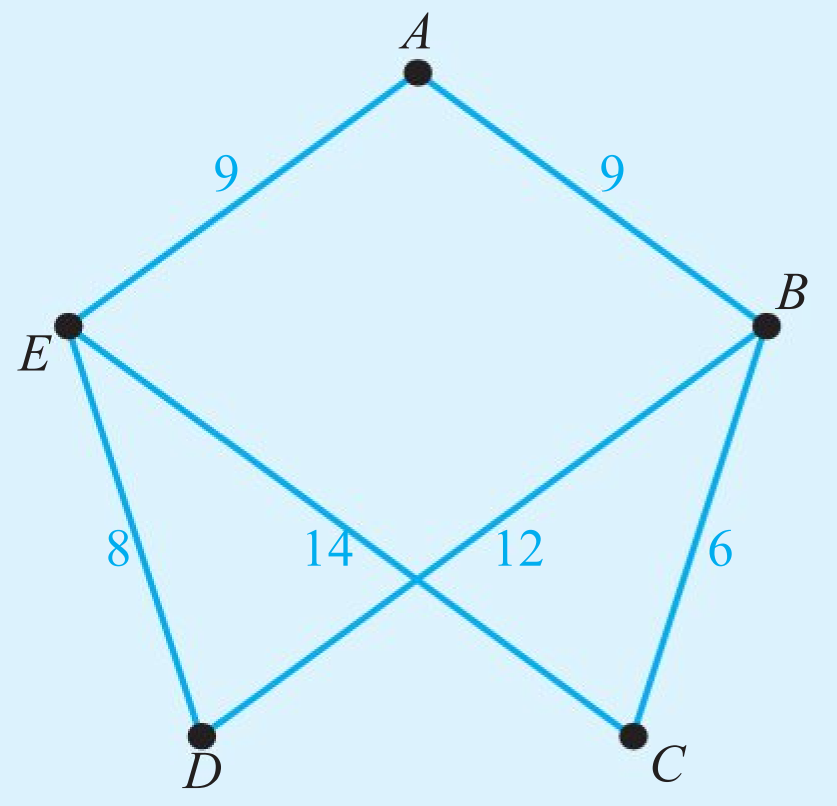
Exercise 7C

For questions 1 and 2, use the method demonstrated in Worked Example 7.12a to construct the adjacency table for each graph.

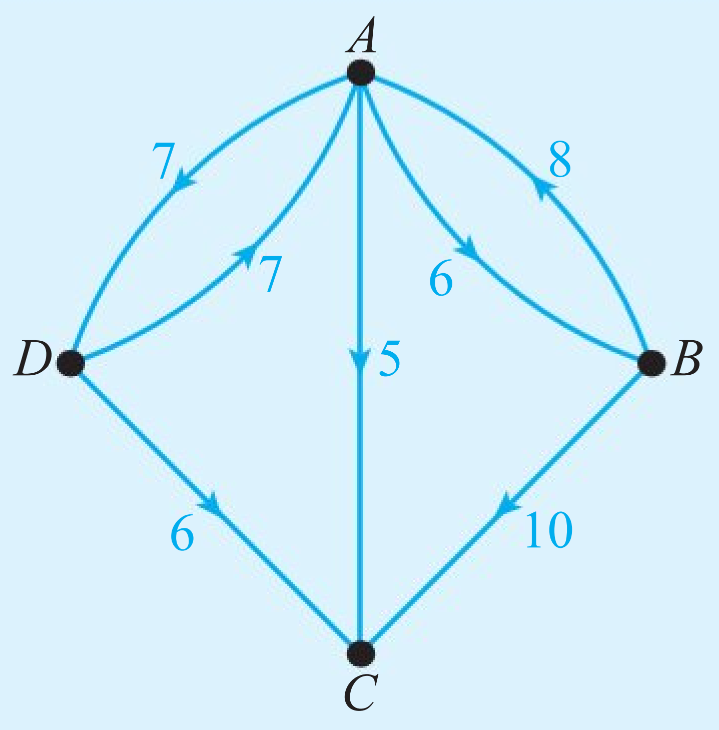
1 **a**



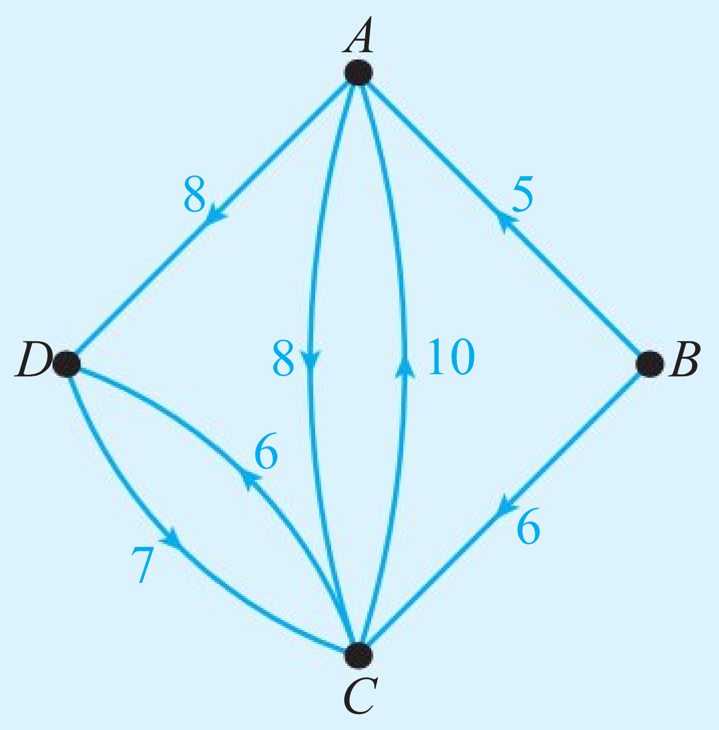
b



2 a



b



For questions 3 and 4, use the method demonstrated in Worked Example 7.12b to draw a weighted graph with the given adjacency table.

3 a

	A	**B**	**C**	**D**
A	–	–	–	7
B	6	–	7	–
C	–	8	–	5
D	–	8	–	–

b

	A	**B**	**C**	**D**
A	–	12	–	–
B	17	–	15	–
C	–	15	–	–
D	25	–	–	–

4 a

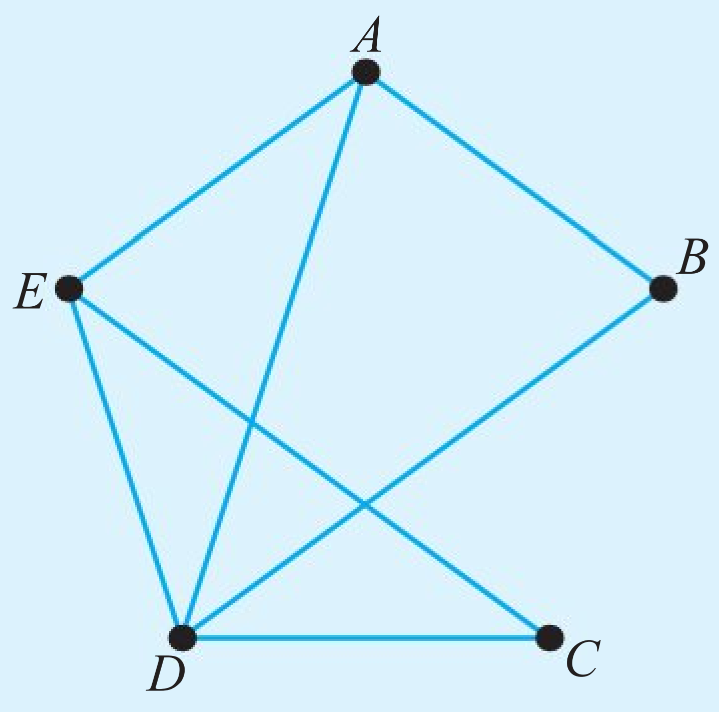
	A	**B**	**C**	**D**
A	–	13	8	10
B	13	–	12	–
C	8	12	–	12
D	10	–	12	–

b

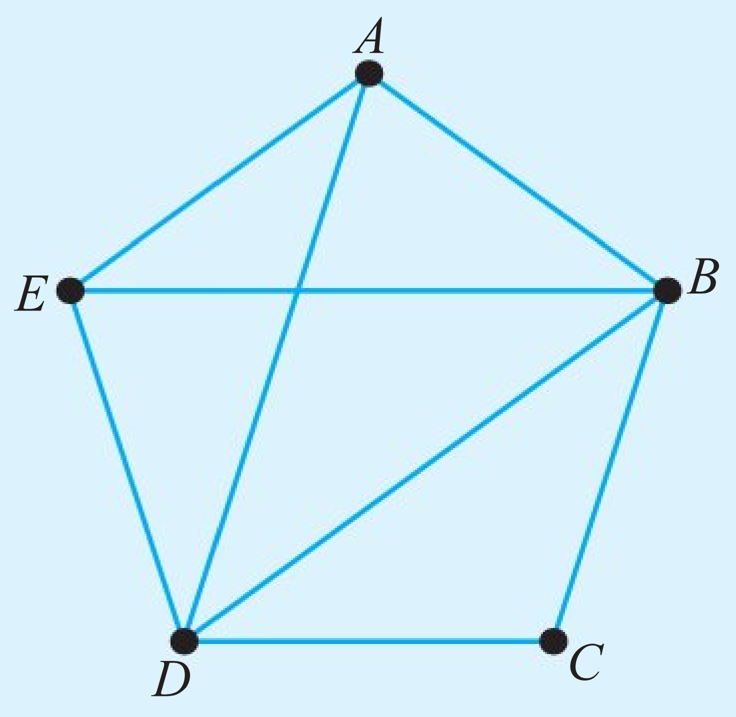
	A	**B**	**C**	**D**
A	–	8	12	9
B	8	–	10	18
C	12	10	–	–
D	9	18	–	–

For questions 5 and 6, use the method demonstrated in Worked Example 7.13 to construct the transition matrix for each graph.

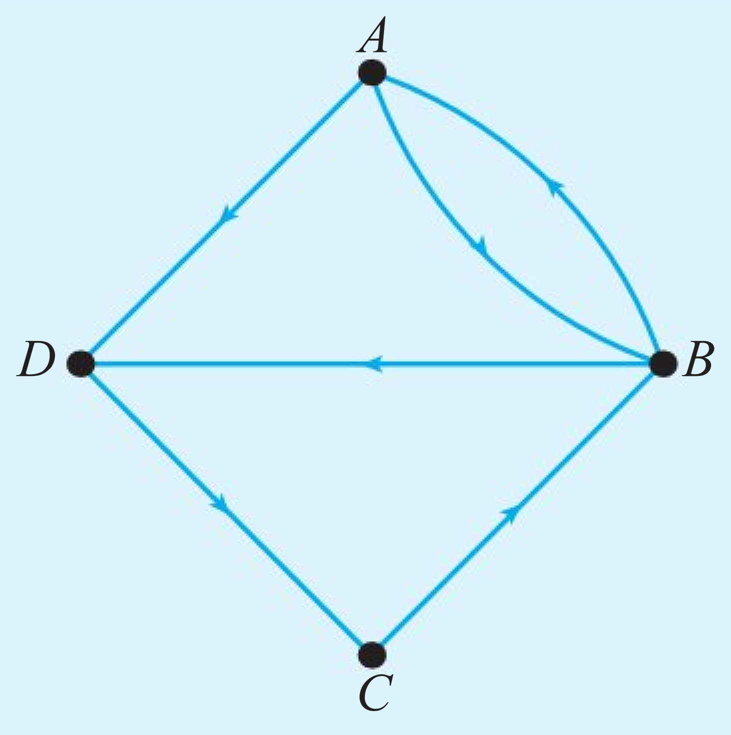
5 a



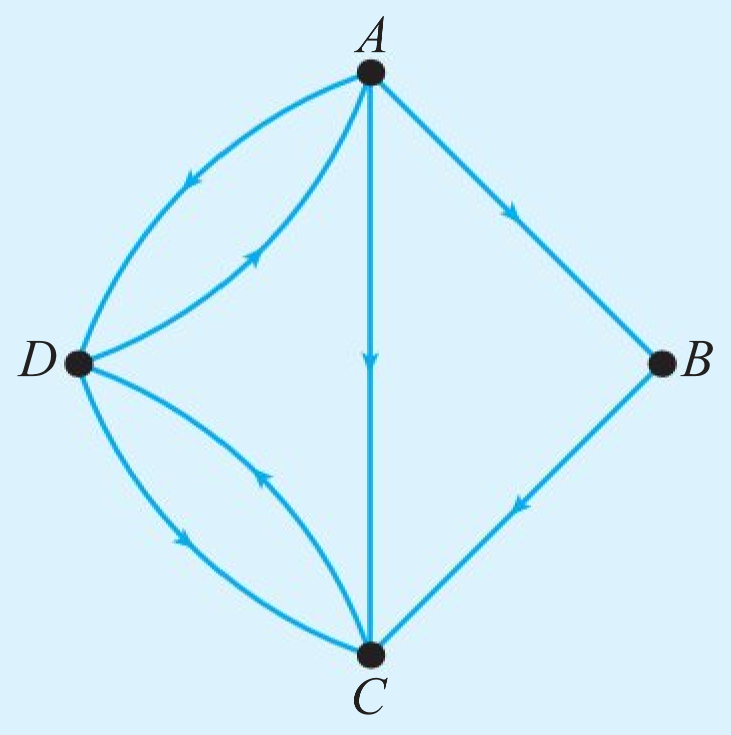
b



6 **a**



b



Questions 7 and 8 refer to the graphs from questions 5 and 6. Use the method demonstrated in Worked Example 7.14 to find the proportion of the time that a random walk would spend at the given vertex.

7 **a** Question 5a, vertex B

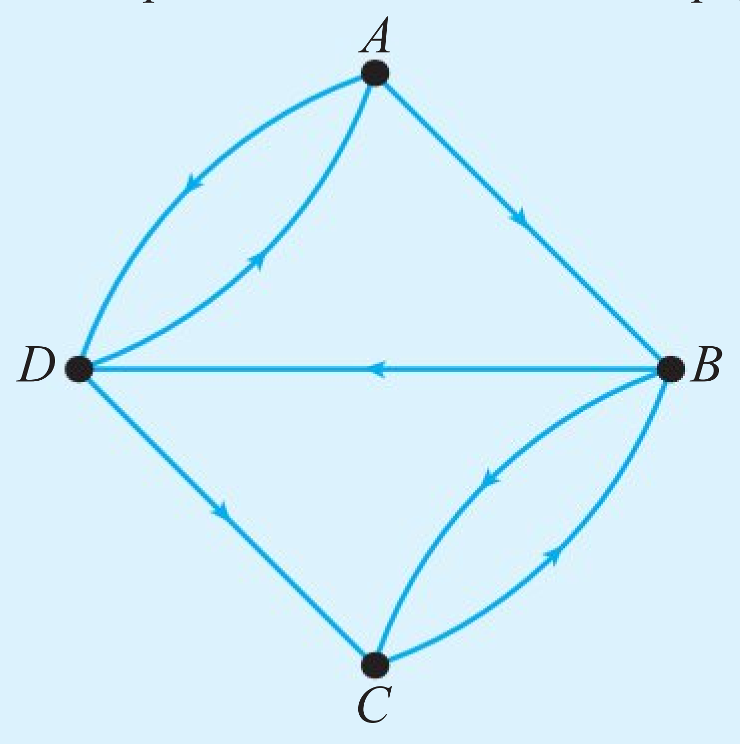
b Question 5b, vertex C

8 **a** Question 6a, vertex A

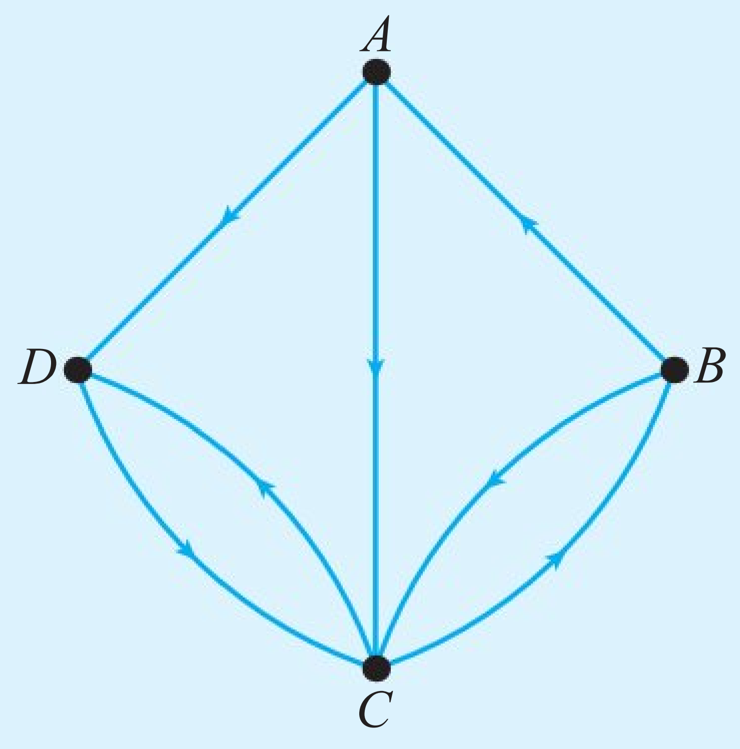
b Question 6b, vertex D

The graphs in questions 9 and 10 show the links between several web pages. Use the method demonstrated in Worked Example 7.15 to determine which page should come top in a search.

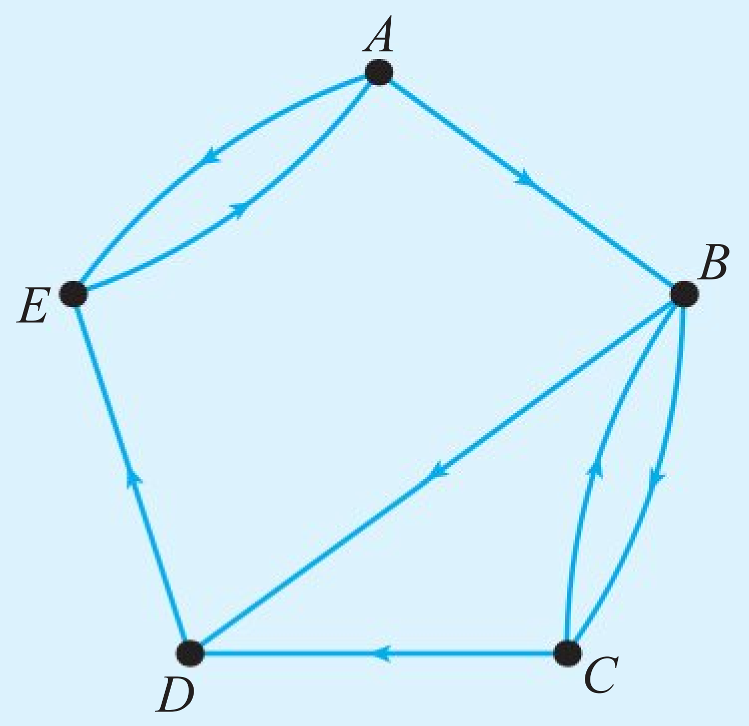
9 **a**



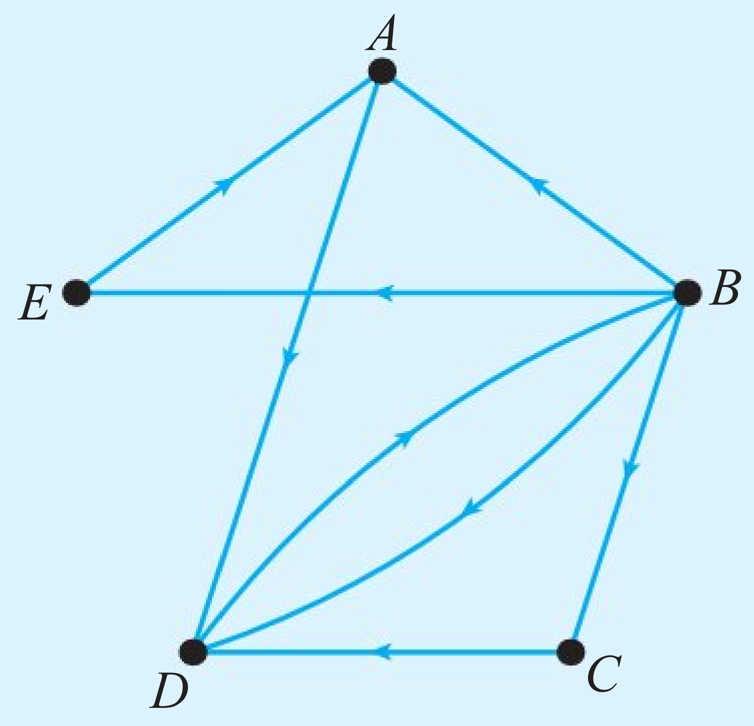
b



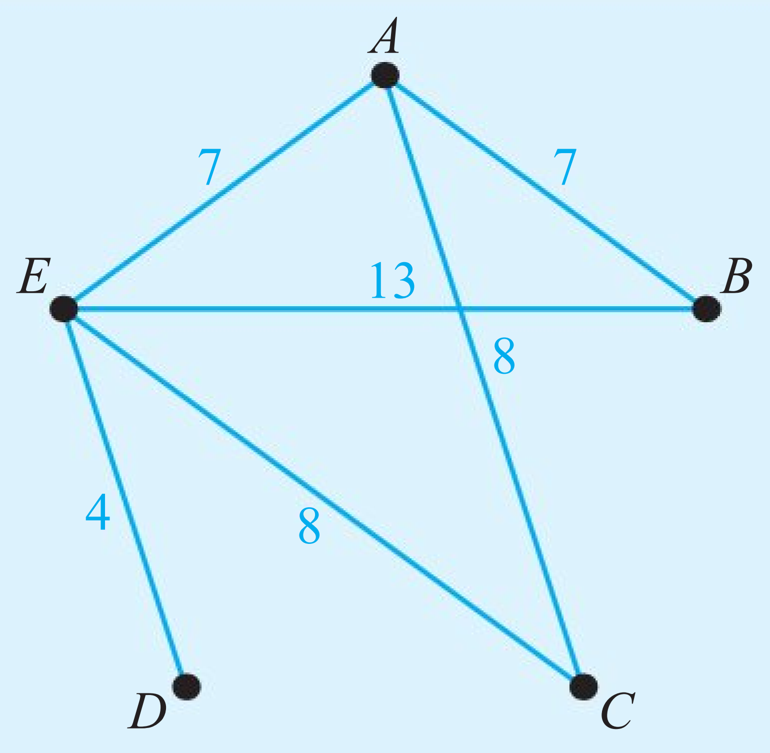
10 **a**



b



11 The figure shows a graph representing direct flight distances, in thousands of kilometres, between five cities.



a Find the shortest total flight distance between cities B and C.

The weighted adjacency table shows the cost of flights, in hundreds of dollars, between the same five cities.

	A	**B**	**C**	**D**	**E**
A	–	0.9	2.7	–	0.8
B	0.9	–	–	–	1.2
C	2.7	–	–	–	2.3
D	–	–	–	–	1.2
E	0.8	1.2	2.3	1.2	–

b Find the cheapest route between cities B and C.

12 Six locations in a park are marked A to F. There are direct paths between some of them. The time, in minutes, taken to walk along those direct paths is given in the following table.

	A	**B**	**C**	**D**	**E**	**F**
A	–	3	–	–	5	–
B	3	–	5	5	–	–
C	–	5	–	–	–	4
D	–	5	–	–	4	3
E	5	–	–	4	–	–
F	–	–	4	3	–	–

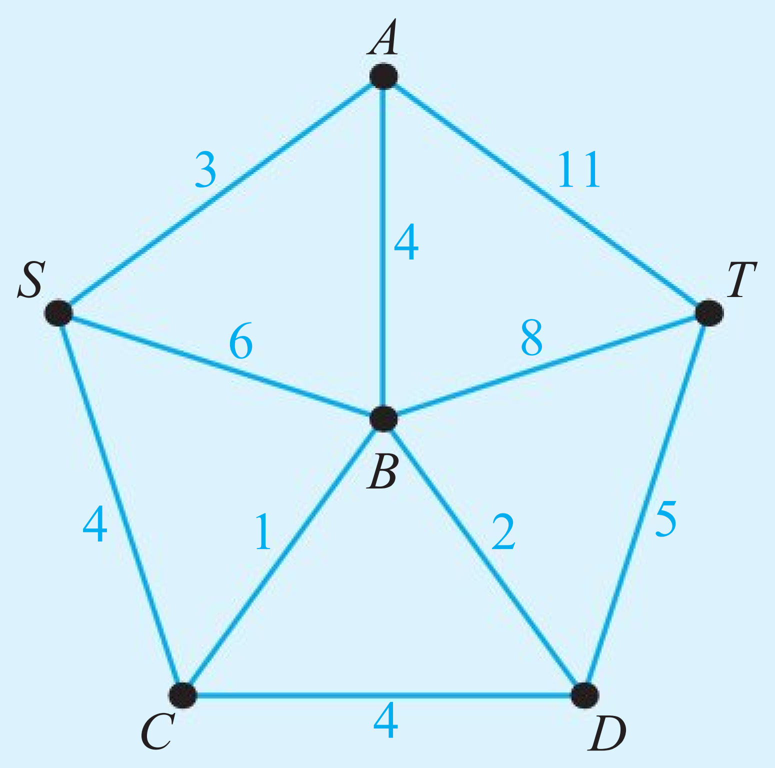
a Draw a graph to represent the information in the table.

b What is the quickest way to walk from A to D? How long does it take?

13 The graph shows the lengths, in kilometres, of roads between six villages.

a Find the length of the shortest route from S to T.

b The road between B and D is closed. What is the new shortest route from S to T? State its length.



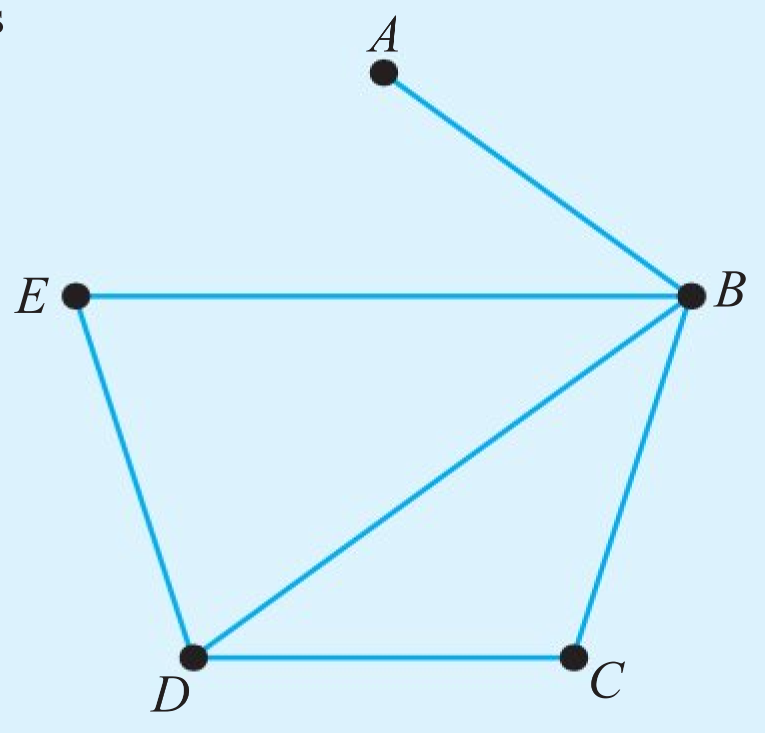
14 Some of the five ports on an island are connected by roads. The connections are shown in the graph alongside.

A tourist arrives at port E. Every day (including the first), he selects one of the available roads at random and travels to another port.

a Construct a transition matrix for the graph.

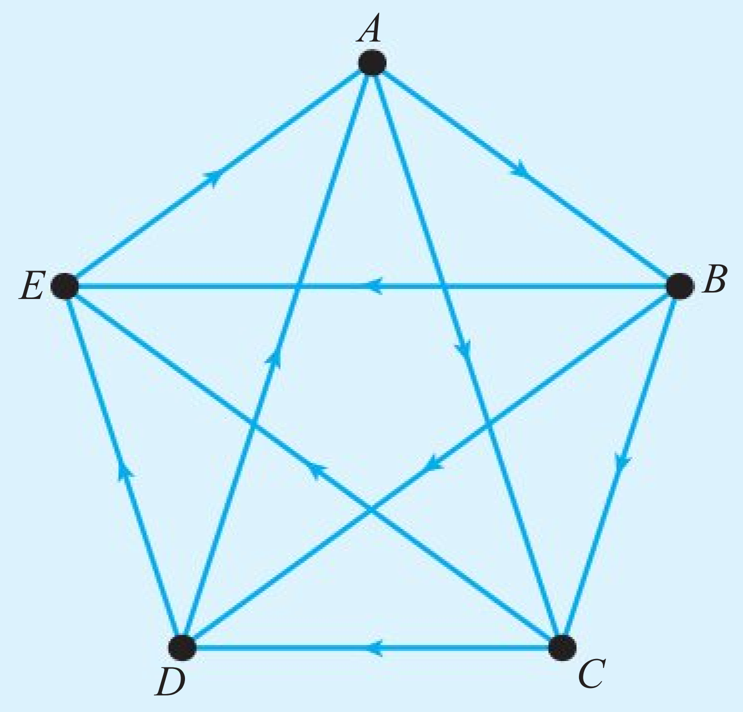
b Find the probability that the tourist travels to port C on the 7th day.

c If the tourist continues travelling around the island for a long time, which port will he visit most often?



15 Links between five websites are represented as a directed graph. For example, website A contains links to websites B and C.

Rank the pages in order of importance, justifying your answer.



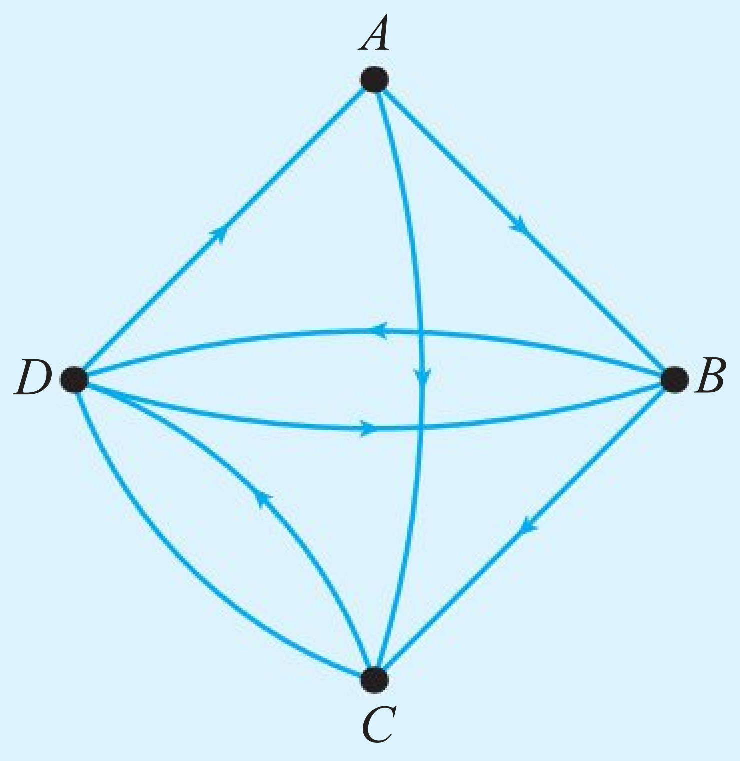
16 The graph shows links between four websites.

a Which website would come top in an internet search?

In an alternative model of an internet search, at each stage the user stops the search with probability 0.1.

b By adding a fifth site corresponding to the end of the search, write down the transition matrix to model this new situation.

c Does the new model predict a different order of importance of the four original websites?



7D Minimum spanning tree algorithms

The table shows the distances (in km) between six villages. Roads are to be built between some of the towns so that it is possible to get from any town to any other town. What is the minimum length of road required?

	A	***B***	***C***	***D***	***E***	***F***
A	–	5	7	–	8	8
B	5	–	6	–	5	–
C	7	6	–	4	4	3
D	–	–	4	–	5	2
E	8	5	4	5	–	–
F	8	–	3	2	–	–

Clearly, if town A is connected to town B, and town B is connected to town C, then we don't need a direct road between A and C. In the language of graph theory, the resulting graph will have no cycles – it will be a tree. The problem is to find the tree of minimum total length (weight) which includes every vertex of the graph. This is called the **minimum spanning tree**.

In this section, you will learn about two different algorithms for finding the minimum spanning tree. Kruskal's algorithm adds edges to the graph, starting with the shortest, until the graph is connected. Prim's algorithm starts with the vertex and then adds the closest possible vertex at each stage.

Tip

The minimum spanning tree is not necessarily unique – there may be more than one tree with the same weight. You are only required to find one answer unless explicitly told otherwise.

Kruskal's algorithm

KEY POINT 7.7

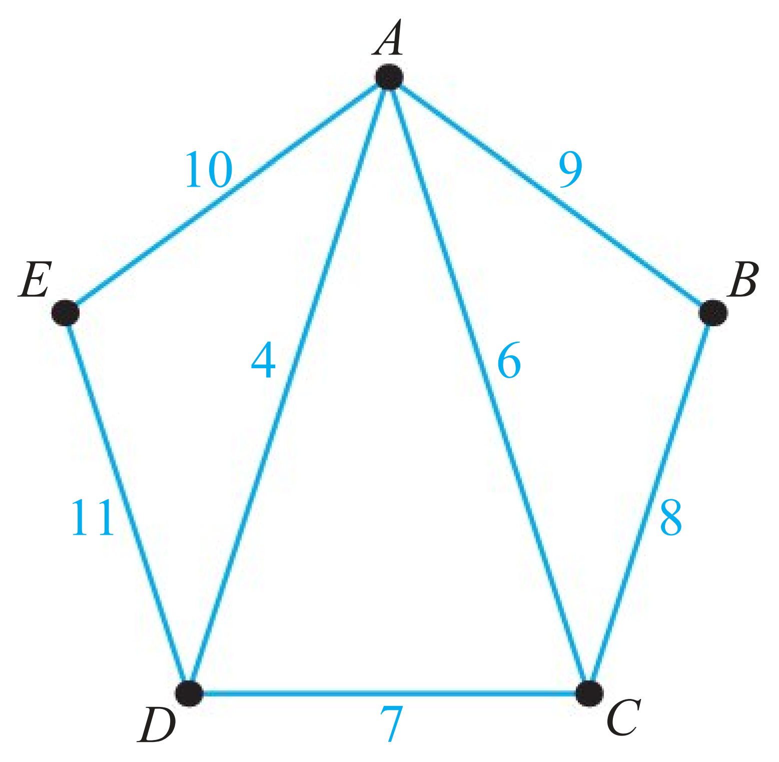
Kruskal's algorithm builds a minimum spanning tree by adding edges one at a time.

- Start by adding the shortest edge.
- At each stage, add the shortest remaining edge as long as it does not create a cycle.
- Keep adding edges until all of the vertices have been connected.

WORKED EXAMPLE 7.16

Use Kruskal's algorithm to find a minimum spanning tree for this graph.

List the edges in the order in which you add them and state the weight of your tree.

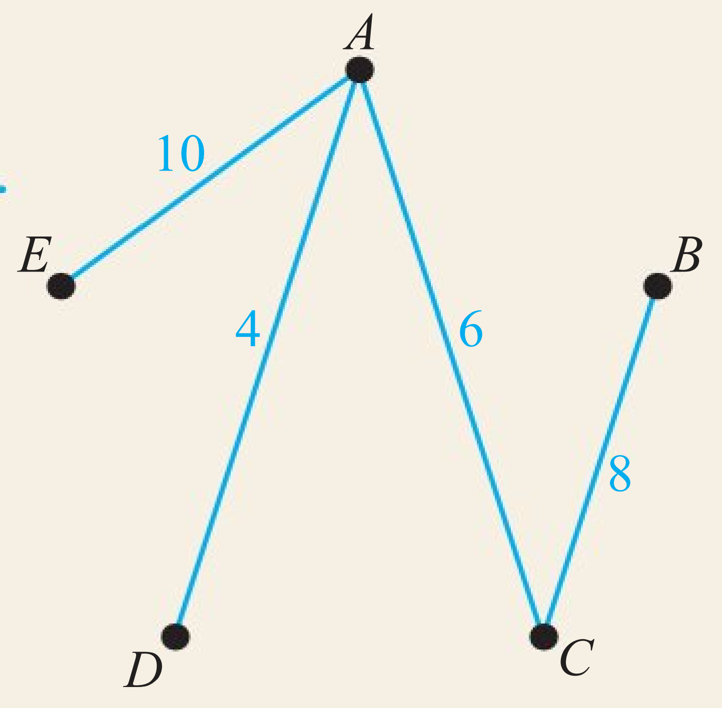


Start with the shortest edge, in this case AD

Add edges in order of increasing length, skipping those that would create a cycle

AD	(4)
AC	(6)
CD	skip
BC	(8)
AB	skip
AE	(10)

Draw the graph as you go along, so you can check that there are no cycles and know when all the vertices have been connected



Add up the lengths of all the edges to find the weight of the tree

The weight of the tree is

$$4+6+8+10=28$$

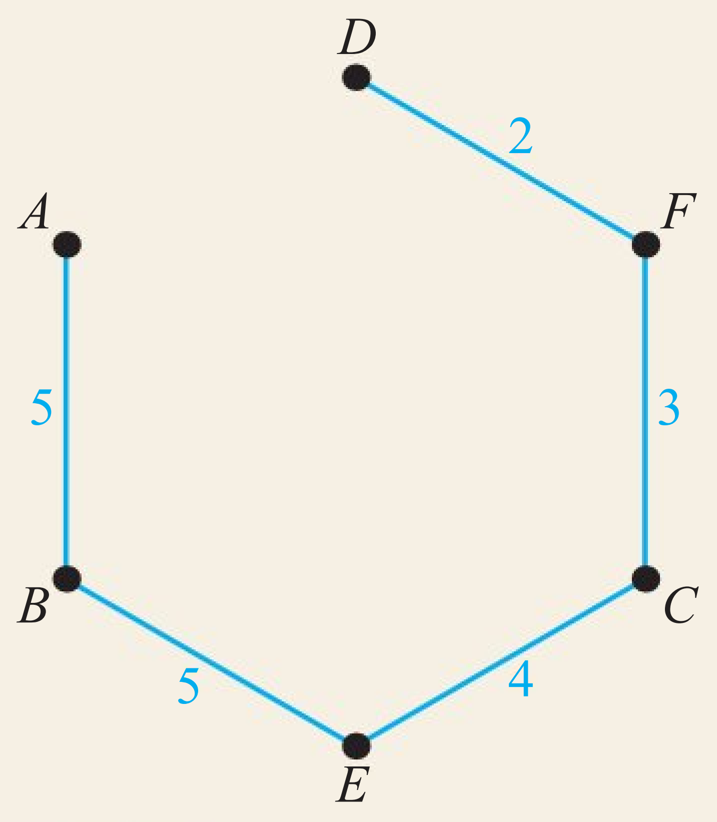
You can also use Kruskal's algorithm when the graph is given by a weighted adjacency table. In this case, it is even more important to draw the graph as you go along, as it is difficult to check for cycles from the table.

WORKED EXAMPLE 7.17

Use Kruskal's algorithm to find a minimum spanning tree for the graph given by this weighted adjacency table.

	A	**B**	**C**	**D**	**E**	**F**
A	–	5	7	–	8	8
B	5	–	6	–	5	–
C	7	6	–	4	4	3
D	–	–	4	–	5	2
E	8	5	4	5	–	–
F	8	–	3	2	–	–

Strat with the shortest edge, which is *DF*	*DF* (2)
	CF (3)
Draw the graph as you go along to make sure that you do not create any cycles	
CD and *CE* both have length 4. *CD* would create a cycle, so add *CE*	*CD* skip
	CE (4)
There are three edges of length 5	*ED* skip
ED would create a cycle (*CEDF*)	*AB* (5)
You can add both *AB* and *EB* without creating a cycle	*EB* (5)
All six vertices have been connected, so the tree is complete	



Prim's algorithm

Kruskal's algorithm is quick and easy to implement on small graphs. However, on a large graph, it is difficult to check whether you are creating a long cycle. Also note that, for an algorithm to be implemented on a computer, you would need an additional algorithm to check for cycles. This can be very time consuming, so for large graphs an alternative algorithm is used.

Tip

Kruskal's and Prim's algorithms will give the same minimum spanning tree. To make it clear which algorithm you have used, you must give the order in which you have added the edges.

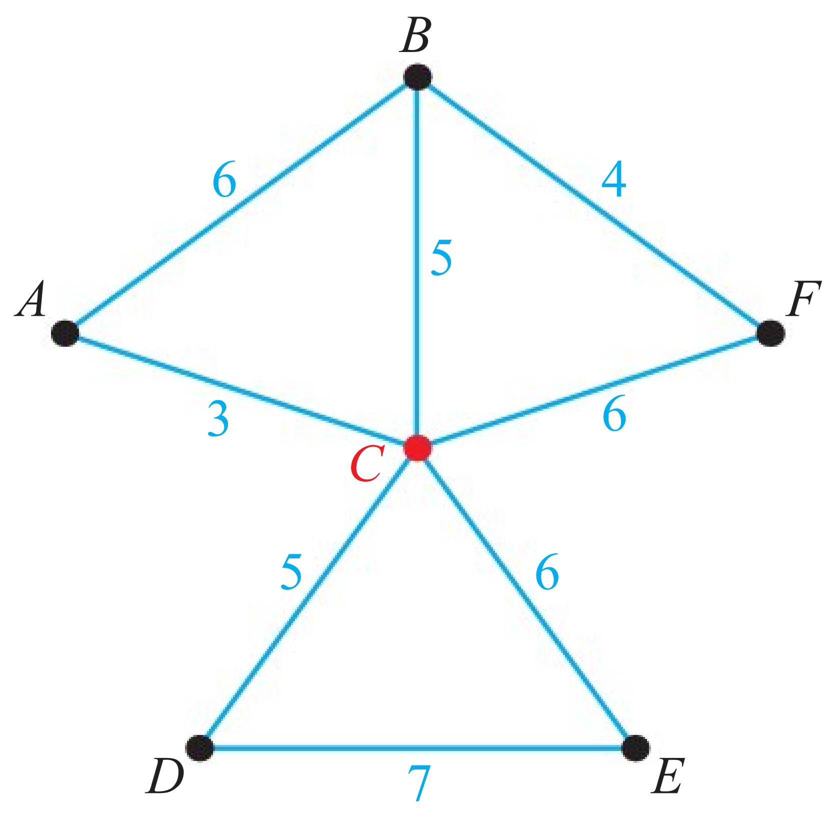
KEY POINT 7.8

Prim's algorithm builds up the tree by adding vertices one at a time.

- Specify a starting vertex.
- At each stage, find the vertex (which has not yet been added) that has the shortest possible distance to any of the vertices that have already been added. Add that vertex and the corresponding edge.
- Stop when all the vertices have been connected.

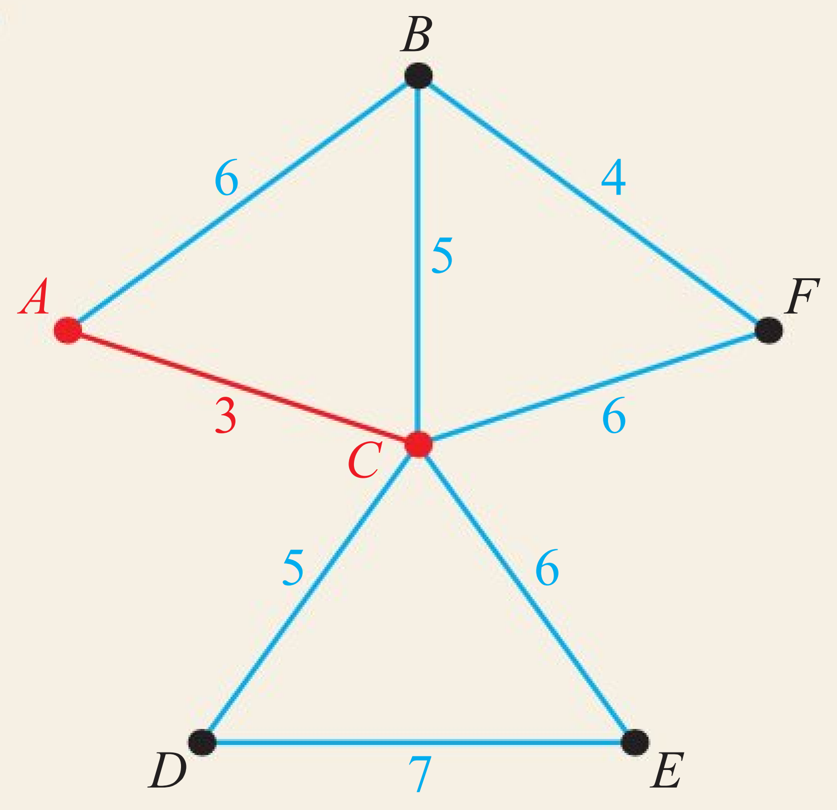
WORKED EXAMPLE 7.18

Use Prim's algorithm starting with vertex *C* to find the minimum spanning tree for this graph. List the edges in order in which they have been added.

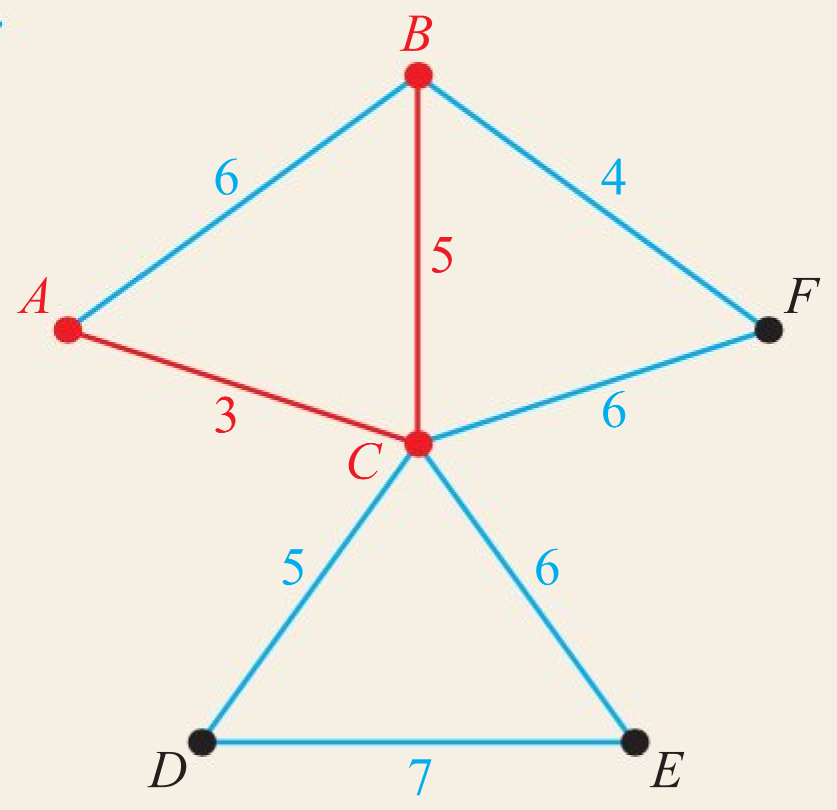


Note: in the diagrams on the right, the vertices and edges that have been added to the graph are marked in red

Find the vertex closest to C – this is A. Add it, together with the edge CA



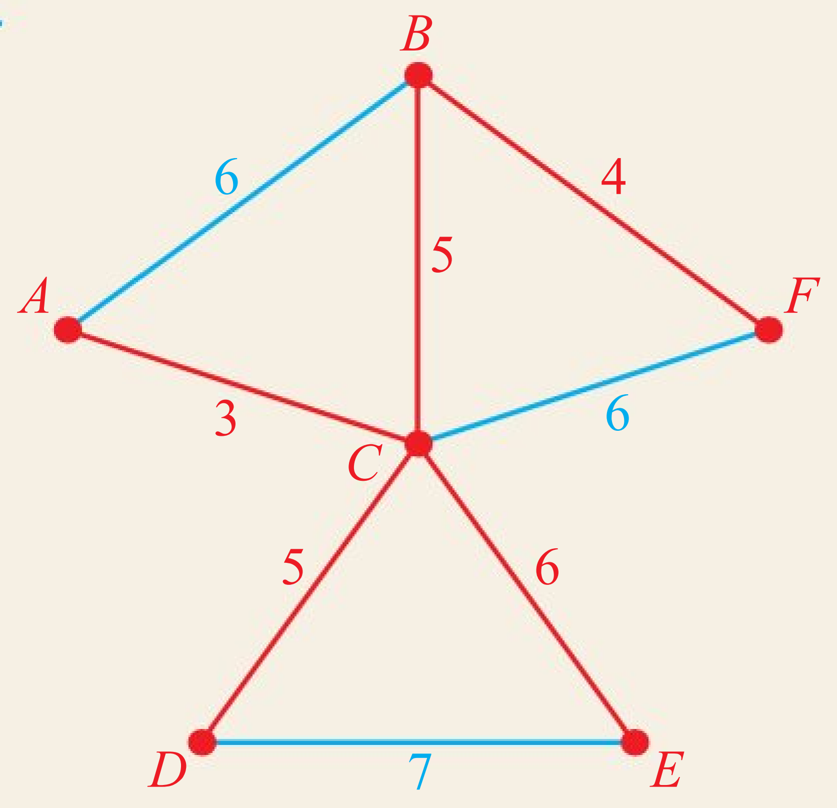
Next look for the vertex closest to *either A or C*. This is vertex B, so we add the edge BC



Then look for the vertex closest to *any of A*, B, or C. This is F, so add the edge BF

The vertex closest to *any of A*, B, C or F is D, so add the edge CD

Finally, the only remaining vertex is E and the shortest edge leading to it is CE



The edges have been added in the following order:
AC, CB, EF, CD, CE

To implement an algorithm on a computer, the graph needs to be stored in the form of its adjacency table. The next example illustrates the implementation of Prim's algorithm in a table. Once a vertex has been connected, we circle its column label and cross out its row. We also draw a box around the connecting edge length. By crossing out a row, we know that the corresponding vertex has already been connected and we do not need to look at any other edges leading to it.

Tip

You do not have to draw a different table for each stage; your method can be seen from the boxed numbers. You may be asked to write down the order in which the edges have been added.

WORKED EXAMPLE 7.19

Graph G has the following weighted adjacency table.

	A	**B**	**C**	**D**	**E**	**F**
A	–	6	3	–	–	5
B	6	–	3	–	3	5
C	3	3	–	–	6	4
D	–	–	–	–	5	2
E	–	3	6	5	–	–
F	5	5	4	2	–	–

Use Prim's algorithm starting at vertex A to find the minimum spanning tree for G. Draw your tree and state its length.

The first connected vertex is A. Circle its column label and cross out its row. Box the shortest edge length in A's column

	$\textcircled{A}$	**B**	**C**	**D**	**E**	**F**
~~**A**~~	~~–~~	~~6~~	~~3~~	~~–~~	~~–~~	~~5~~
B	6	–	3	–	3	5
C	$\boxed{3}$	3	–	–	6	4
D	–	–	–	–	5	2
E	–	3	6	5	–	–
F	5	5	4	2	–	–

The next connected vertex is C. Circle the column label and cross out the row

Look at all the numbers in A and C columns that haven't been crossed out; box the smallest

	$\textcircled{A}$	**B**	$\textcircled{C}$	**D**	**E**	**F**
~~**A**~~	~~–~~	~~6~~	~~3~~	~~–~~	~~–~~	~~5~~
B	6	–	$\boxed{3}$	–	3	5
~~**C**~~	~~$\boxed{3}$~~	~~3~~			~~6~~	~~4~~
D	–	–	–	–	5	2
E	–	3	6	5	–	–
F	5	5	4	2	–	–

Vertex B has now been connected, so we circle its column label and cross out its row

The smallest uncrossed number in columns A, B and C is 3, corresponding to vertex E

	$\textcircled{A}$	$\textcircled{B}$	$\textcircled{C}$	**D**	**E**	**F**
~~**A**~~	~~–~~	~~6~~	~~3~~	~~–~~	~~–~~	~~5~~
~~**B**~~	~~6~~	~~–~~	~~$\boxed{3}$~~	~~–~~	~~3~~	~~5~~
~~**C**~~	~~$\boxed{3}$~~	~~3~~			~~6~~	~~4~~
D	–	–	–	–	5	2
E	–	$\boxed{3}$	6	5	–	–
F	5	5	4	2	–	–

Vertex *E* has now been connected

We look for the smallest uncrossed number from columns *A*, *B*, *C*, *E*. It is in row *F*

	$\textcircled{A}$	$\textcircled{B}$	$\textcircled{C}$	D	$\textcircled{E}$	F
~~A~~	~~–~~	~~6~~	~~3~~	~~–~~	~~–~~	~~5~~
~~B~~	~~6~~	~~–~~	~~$\boxed{3}$~~	~~–~~	~~3~~	~~5~~
~~C~~	~~$\boxed{3}$~~	~~3~~	~~–~~	~~–~~	~~6~~	~~4~~
D	–	–	–	–	5	2
~~E~~	~~–~~	~~$\boxed{3}$~~	~~6~~	~~5~~	~~–~~	~~–~~
F	5	5	$\boxed{4}$	2	–	–

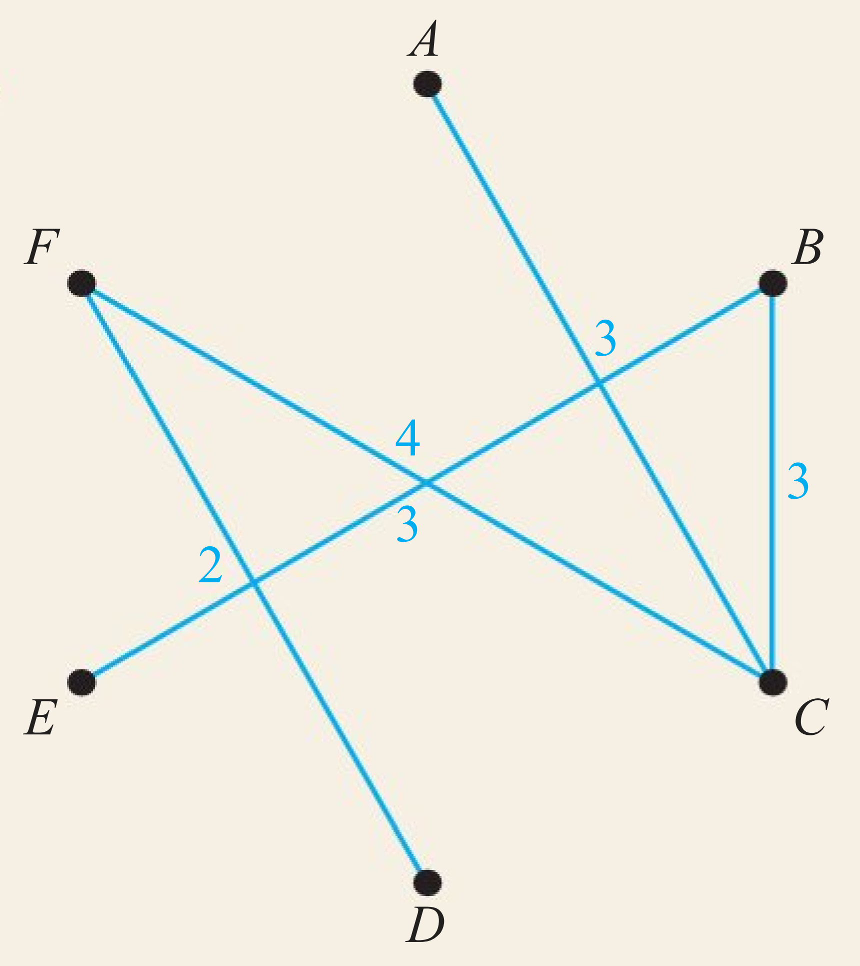
Vertex *F* has now been connected

The smallest uncrossed number is 2, and it corresponds to vertex *D* (the only vertex that has not been connected yet)

	$\textcircled{A}$	$\textcircled{B}$	$\textcircled{C}$	D	$\textcircled{E}$	$\textcircled{F}$
~~A~~	~~–~~	~~6~~	~~3~~	~~–~~	~~–~~	~~5~~
~~B~~	~~6~~	~~–~~	~~$\boxed{3}$~~	~~–~~	~~3~~	~~5~~
~~C~~	~~$\boxed{3}$~~	~~3~~	~~–~~	~~–~~	~~6~~	~~4~~
D	–	–	–	–	5	$\boxed{2}$
~~E~~	~~–~~	~~$\boxed{3}$~~	~~6~~	~~5~~	~~–~~	~~–~~
~~F~~	~~5~~	~~5~~	~~$\boxed{4}$~~	~~2~~	~~–~~	~~–~~

All vertices have been connected, so the tree is complete

You can use the boxed numbers to draw the tree and find its weight



Weight = 15

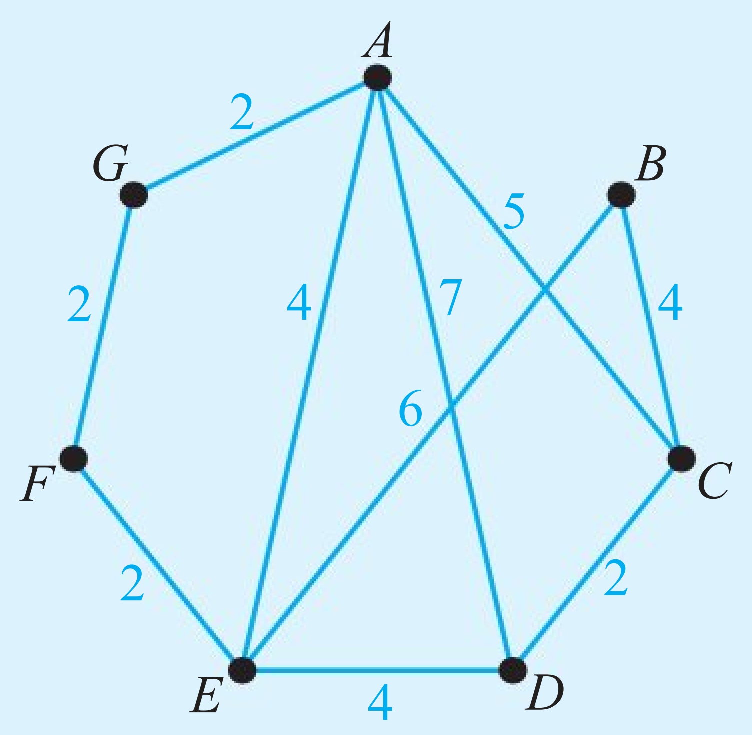
You are the Researcher

Both Kruskal's and Prim's are examples of so-called **greedy algorithms**, where at each stage we pick the best possible option (in this case, the shortest possible edges). This strategy does not work for other types of problems on graphs, for example finding the shortest path between two vertices. One method to solve this type of problem is called Djikstra's algorithm.

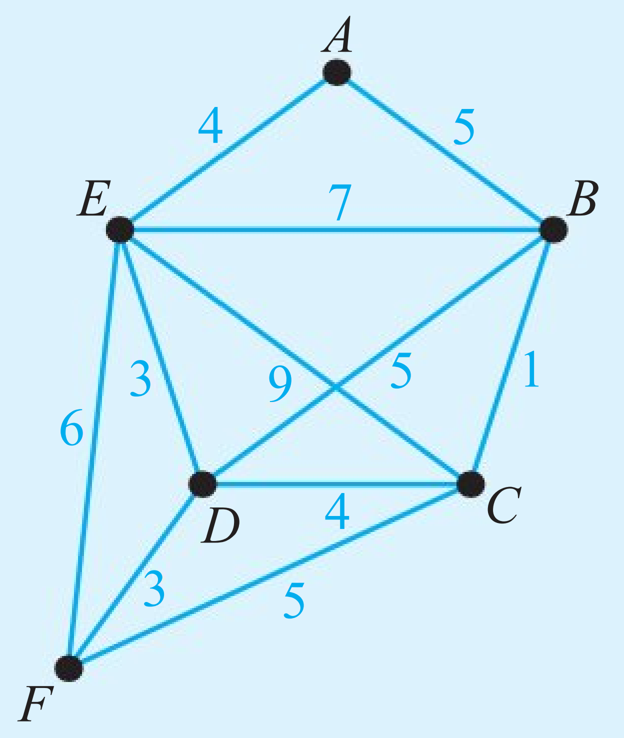
Exercise 7D

For questions 1 to 3, use Kruskal's algorithm, illustrated in Worked Example 7.16, to find the minimum spanning tree for these graphs. Draw your tree and state its weight.

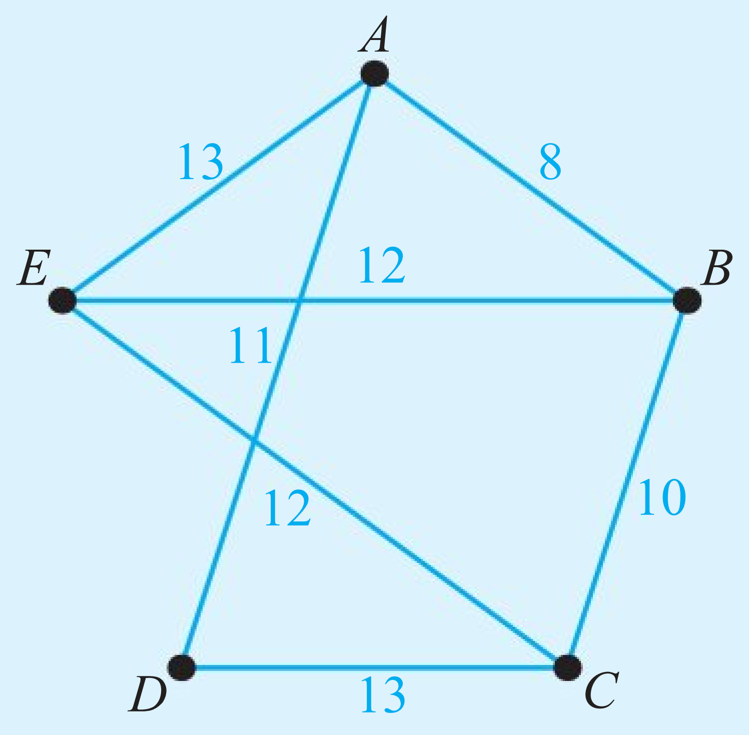
1 a



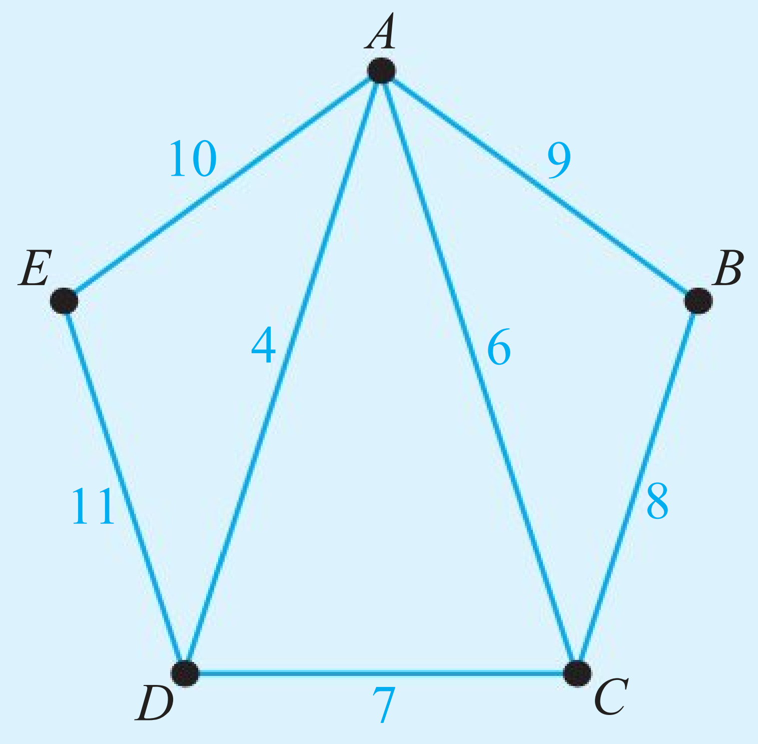
b



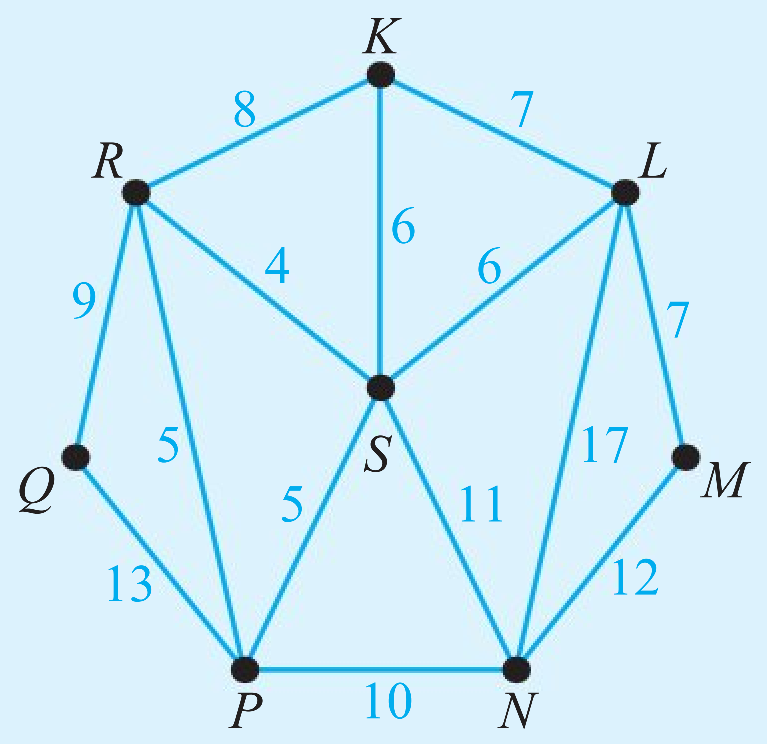
2 a



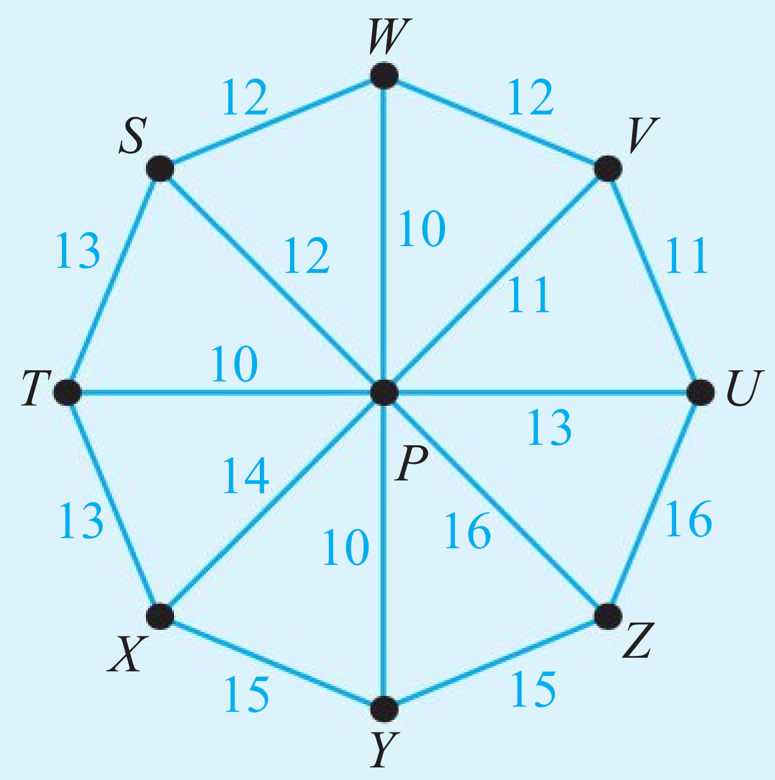
b



3 a



b



For questions 4 to 6, use Kruskal's algorithm, illustrated in Worked Example 7.17, to find the minimum spanning tree for the graphs given by these adjacency tables. Draw your tree and state its weight.

4 a

	A	**B**	**C**	**D**	**E**
A	–	10	8	7	10
B	10	–	5	4	9
C	8	5	–	7	10
D	7	4	7	–	8
E	10	9	10	8	–

b

	A	**B**	**C**	**D**	**E**
A	–	12	16	11	16
B	12	–	13	14	19
C	16	13	–	15	18
D	11	14	15	–	18
E	16	19	18	18	–

5 a

	A	**B**	**C**	**D**	**E**	**F**	**G**
A	–	–	30	–	–	50	45
B	–	–	70	35	40	–	–
C	30	70	–	50	–	–	20
D	–	35	50	–	10	–	15
E	–	40	–	10	–	15	–
F	50	–	–	–	15	–	10
G	45	–	20	15	–	10	–

b

	A	**B**	**C**	**D**	**E**	**F**	**G**
A	–	25	35	40	–	–	–
B	25	–	50	35	40	–	–
C	35	50	–	45	45	–	40
D	40	35	45	–	20	35	30
E	–	40	45	20	–	15	25
F	–	–	–	35	15	–	–
G	–	–	40	30	25	–	–

6 a

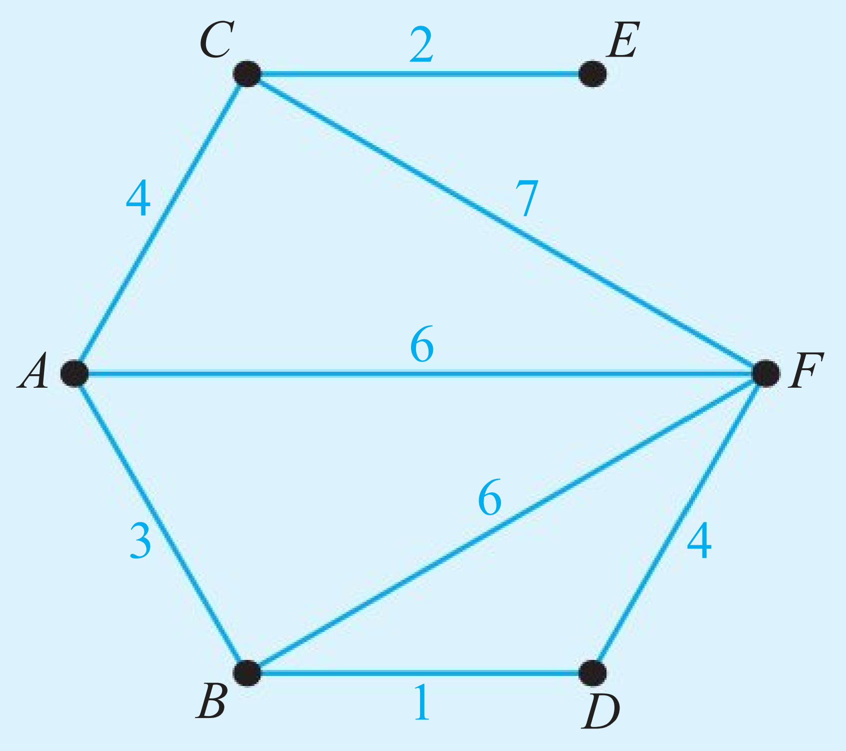
	A	**B**	**C**	**D**	**E**	**F**
A	–	4	–	6	–	10
B	4	–	5	–	5	–
C	–	5	–	6	–	4
D	6	–	6	–	7	–
E	–	5	–	7	–	2
F	10	–	4	–	2	–

b

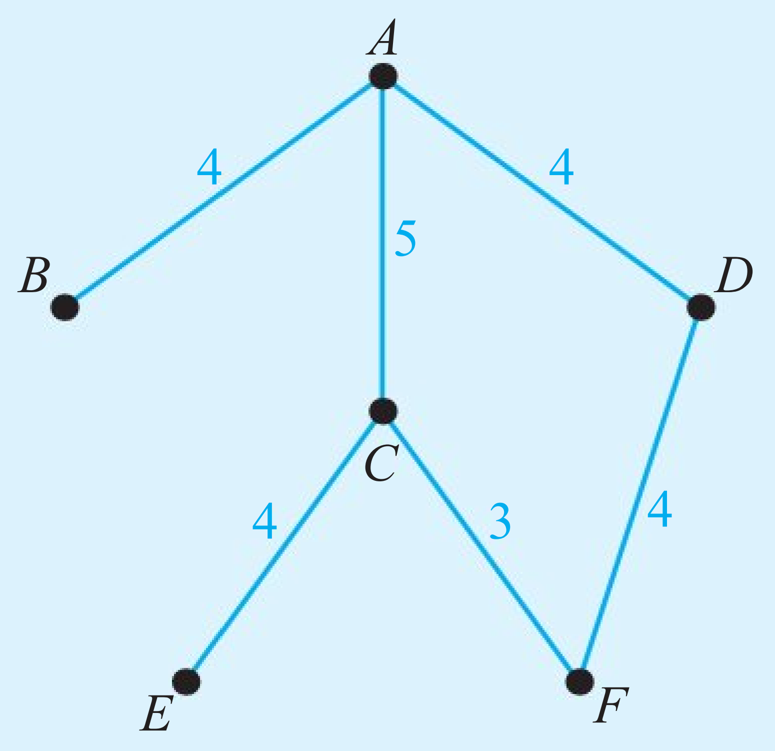
	A	**B**	**C**	**D**	**E**	**F**
A	–	5	7	–	8	8
B	5	–	6	–	5	–
C	7	6	–	4	4	3
D	–	–	4	–	5	2
E	8	5	4	5	–	–
F	8	–	3	2	–	–

For questions 7 to 9, use Prim's algorithm starting at vertex *A*, as illustrated in Worked Example 7.18, to find the minimum spanning tree for each graph. State the edges in the order that you add them.

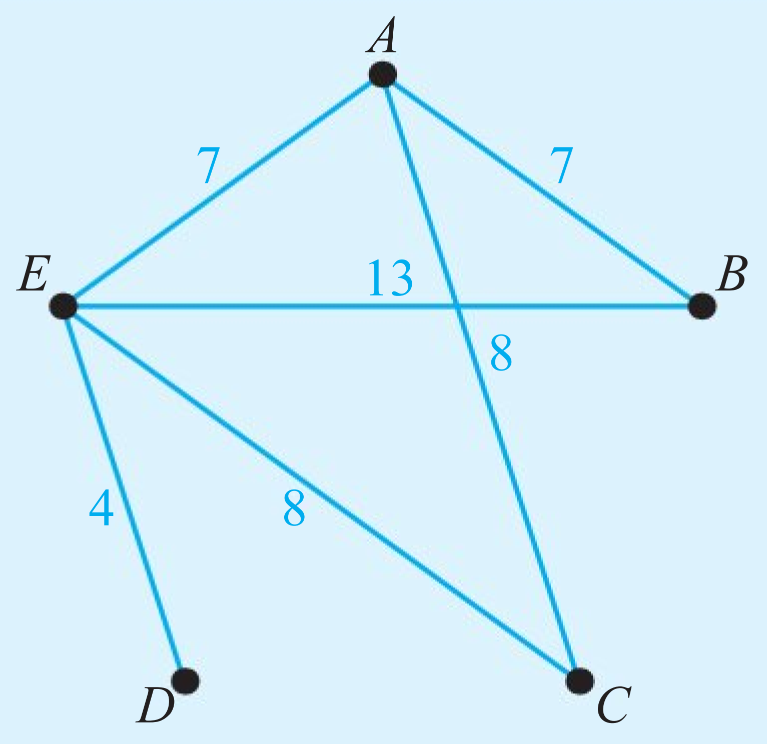
7 a



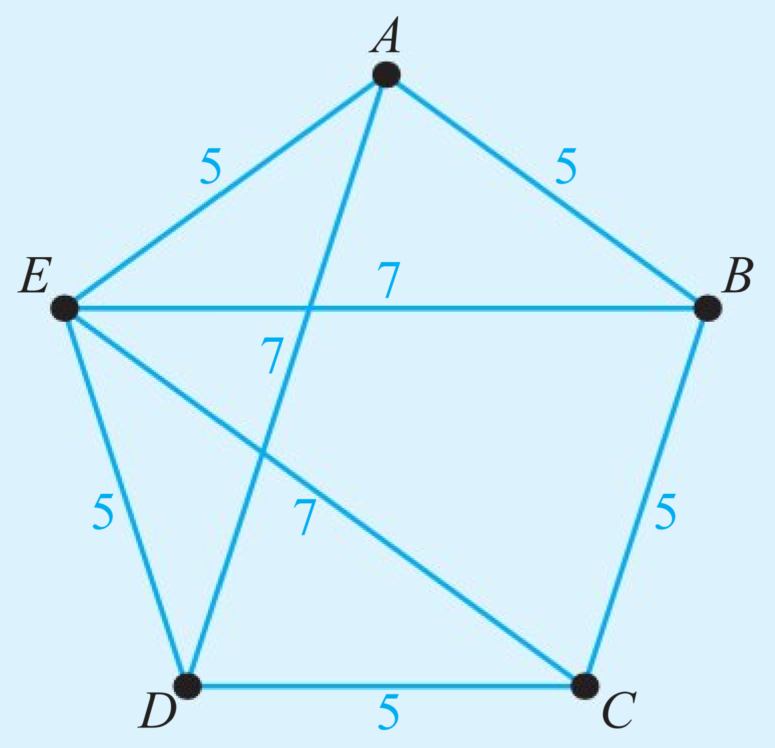
b



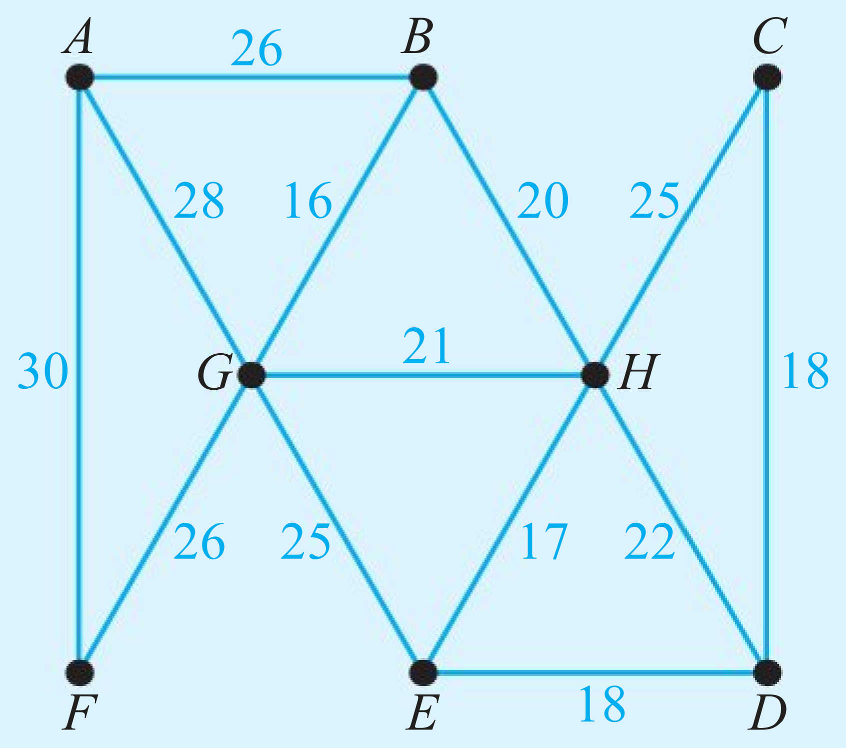
8 **a**



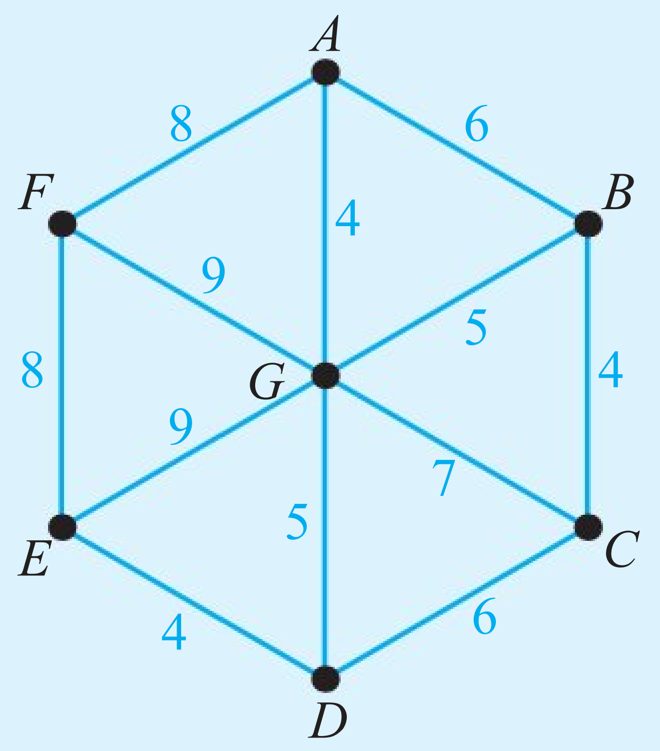
b



9 **a**



b



For questions 10 to 12, use the matrix method for Prim's algorithm, illustrated in Worked Example 7.19, to find the minimum spanning tree for the graphs given by adjacency tables in questions 4 to 6. State the order in which you added the edges. The starting vertex is given in each question.

10 **a** Question 4a, start at A

b Question 4b, start at A

11 **a** Question 5a, start at E

b Question 5b, start at E

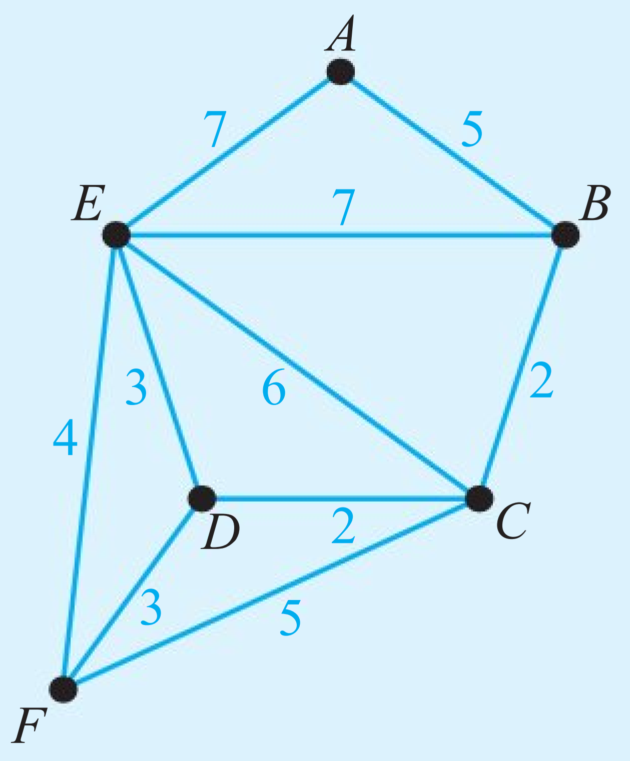
12 **a** Question 6a, start at C

b Question 6b, start at C

13 Six villages are to be connected by new roads. The graph shows potential roads and their lengths in kilometres.

a Use Prim's algorithm, starting at vertex A, to find the minimum spanning tree for the graph. Draw your tree.

b What is the minimum length of road required to ensure that each village is connected to all the others?

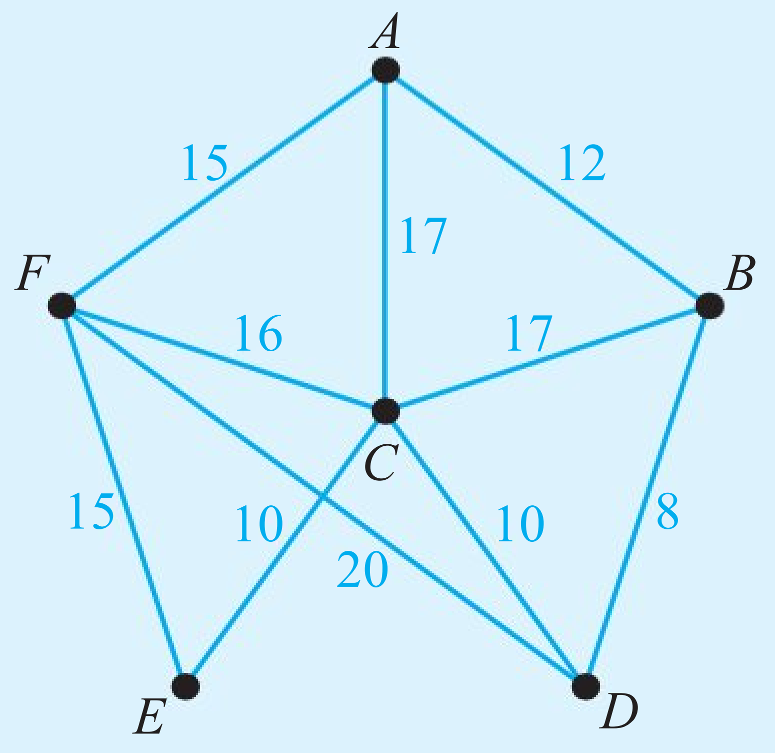


14 The graph alongside shows travel time, in minutes, for bus routes between different locations in a town. Some of the routes are to be closed, but it must still be possible to get from any location to any other.

a Use Kruskal's algorithm to determine which routes should remain open.

It is required that the route *CF* remains open.

b Describe how you could adapt Kruskal's algorithm to determine which other routes should remain open.



15 **a** Explain briefly the differences between Kruskal's and Prim's algorithms for finding the minimum spanning tree of a graph.

b Use Prim's algorithm to find the minimum spanning tree of the graph given by the weighted adjacency table shown. Perform the algorithm on the matrix and use *A* as the starting vertex. Draw your tree.

	A	**B**	**C**	**D**	**E**	**F**
A	–	12	8	14	9	10
B	12	–	10	7	11	15
C	8	10	–	12	14	11
D	14	7	12	–	13	10
E	9	11	14	13	–	12
F	10	15	11	10	12	–

The numbers in the table represent distances, in metres, between different workstations in an office. New wiring needs to be laid down to supply electricity to each workstation. An electricity supply is located next to workstation *A*.

c State the minimum length of wiring required.

d Explain the significance, in this context, of your minimum spanning tree being

i connected

ii a tree.

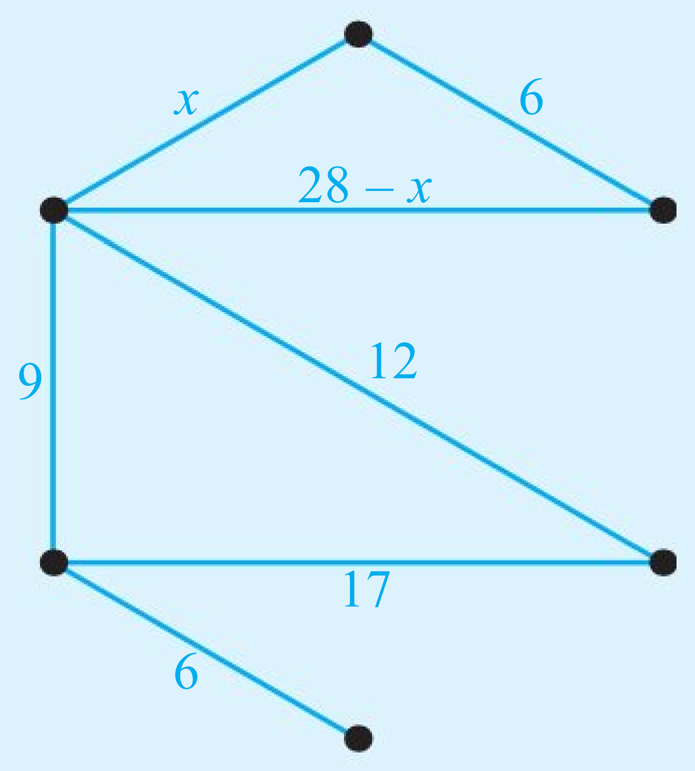
16 A simple connected graph has 5 vertices and 7 edges of lengths 10, 11, 13, 15, 18, 20 and 22. Find the

a minimum possible

b maximum possible

weight of the minimum spanning tree.

17 The minimum spanning tree of the graph shown in the diagram has weight less than 40. Find the range of possible values of *x* given that all the edges have positive weight.



7E The Chinese postman problem

In the final two sections of this chapter you will study two of the most famous problems of graph theory: The Chinese postman problem and the travelling salesman problem.

The Chinese postman problem is to find the shortest path around the graph which uses each edge at least once and returns to the starting point. Think about a postman who needs to walk along every street. This is sometimes also called the route inspection problem.

You know from Section 7B that a graph is Eulerian if all of its vertices have even degrees.

If the graph is Eulerian, any Eulerian cycle is a solution to the problem; it uses each edge exactly once so the length of the required path is the sum of the lengths of all the edges.

If the graph is not Eulerian, then some edges need to be repeated. This is equivalent to adding extra edges ('doubling up' some of the existing ones) to make the graph Eulerian. The method for doing this depends on the number of odd degree vertices in the graph.

Graphs with two odd vertices

KEY POINT 7.9

Chinese postman algorithm for a graph with two odd vertices:

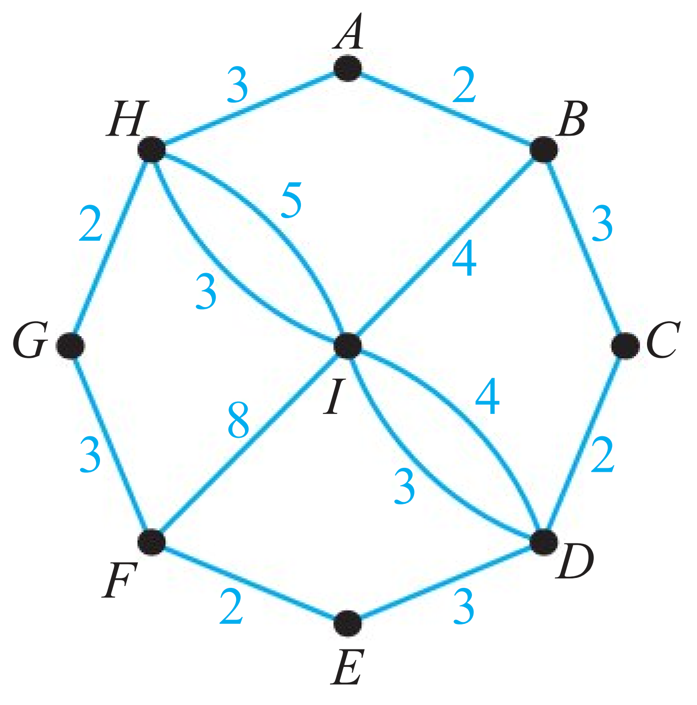
- Identify the two vertices of odd degree.
- Find the shortest path between those two vertices.
- The edges in this shortest path need to be used twice, and all the other edges are used only once.
- The length of the route is the total weight of all the edges plus the length of the edges that are used twice (the shortest path from the previous step).

WORKED EXAMPLE 7.20

a Find the length of the shortest Chinese postman route for the graph on the right.

b State which edges need to be used twice.

c Find one such route starting and finishing at A.



Check the degrees of all the vertices — **a** There are two odd vertices: B and F

Find the shortest path from E to F — Shortest path from B to F: B-I-F (length = 7)

The length is the weight of all the edges plus 7 — Total length = 42 + 7 = 49

The edges in the shortest path from E to F need to be repeated — **b** The edges BI and IF are used twice.

Find one possible route by inspection, remembering that BI and IF can be used twice. The easiest way to achieve this, if possible, is to include $BIFIB$ at some point — **c** A possible route: $ABIFIBCDIDEFGHIHA$

Graphs with four odd vertices

You need to split the four vertices into two pairs and find which particular pairing option produces the smallest possible extra distance.

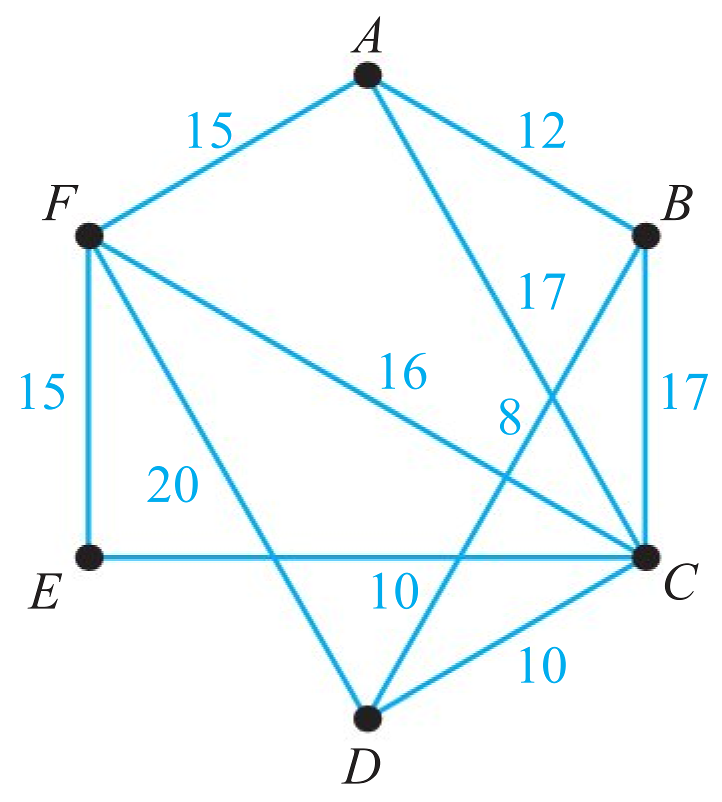
KEY POINT 7.10

Chinese postman algorithm for a graph with four odd vertices:

- Write down all possible pairings between the four odd vertices (there will be three of them).
- For each pairing possibility, find the total of the shortest distances between the two vertices in a pair.
- Select the pairing which gives the smallest total. The corresponding edges need to be repeated.

WORKED EXAMPLE 7.21

Solve Chinese postman problem for the graph below. State which edges need to be repeated, and the length of the shortest route.



Identify the vertices of odd degree — Vertices of odd degree:

$A(3),\ B(3),\ C(5),\ D(3)$

Consider all possible pairings. For each pairing, find the shortest path between the vertices —

AB and CD: $12(AB)+10(CD)=22$

AC and BD: $17(AC)+8(BD)=25$

AD and BC: $10(ABD)+17(BC)=27$

Select the pairing with the smallest total — Edges AB and CD need to be repeated.

The sum of all the weights in the original graph is 140 — The length of the shortest route

$140+22=162$

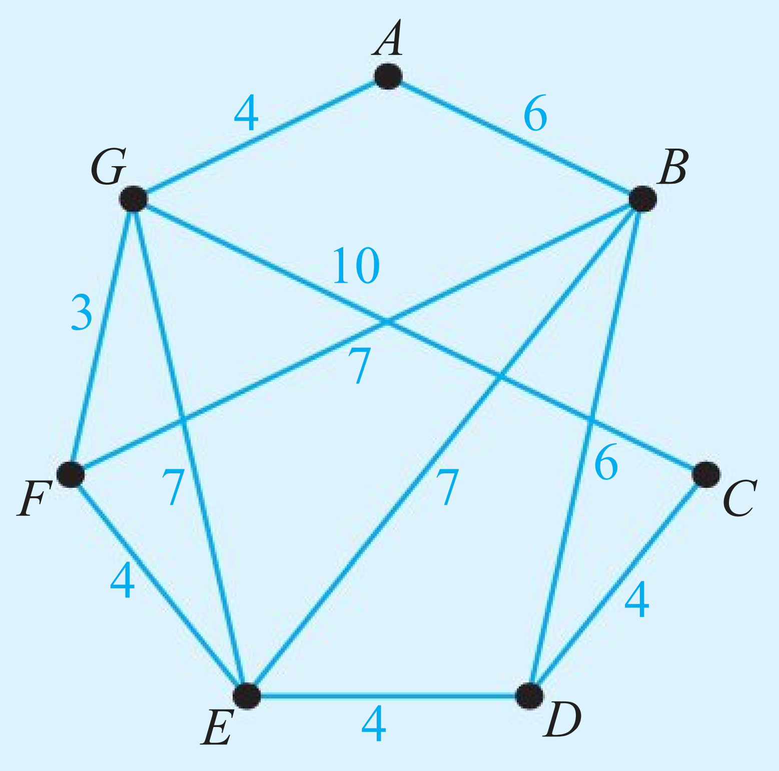
You are the Researcher

You may be wondering why we have not considered graphs with one or three odd vertices. Find out more about the handshaking lemma, the important result mentioned earlier that restricts the possible number of odd vertices in a graph.

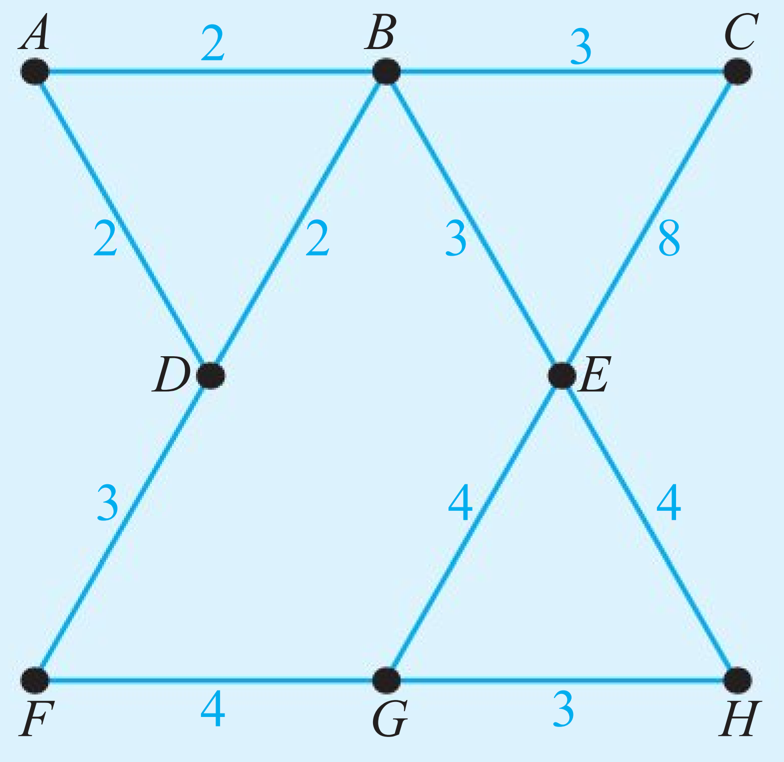
Exercise 7E

For questions 1 and 2, use the method demonstrated in Worked Example 7.20 to solve the Chinese postman problem for each graph. State which edges need to be repeated and the length of the shortest route.

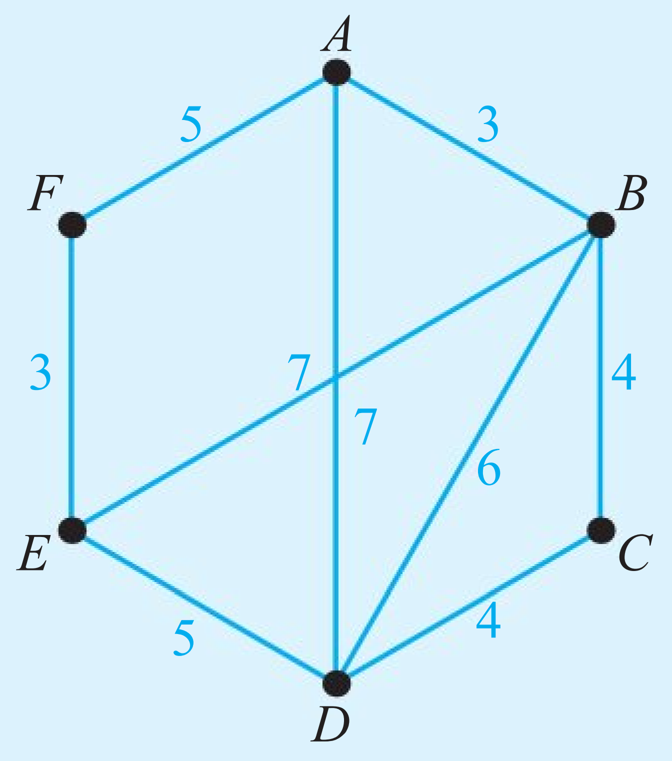
1 **a**



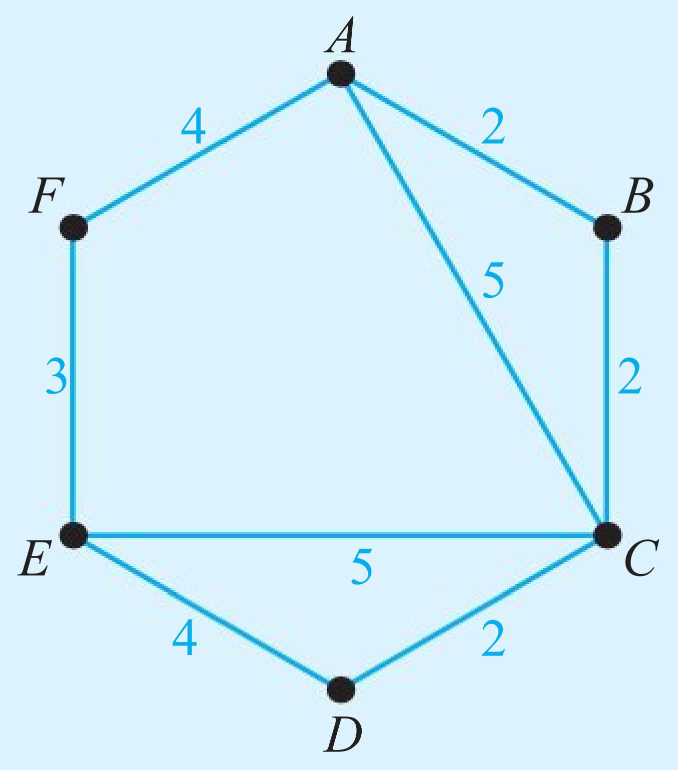
b



2 **a**

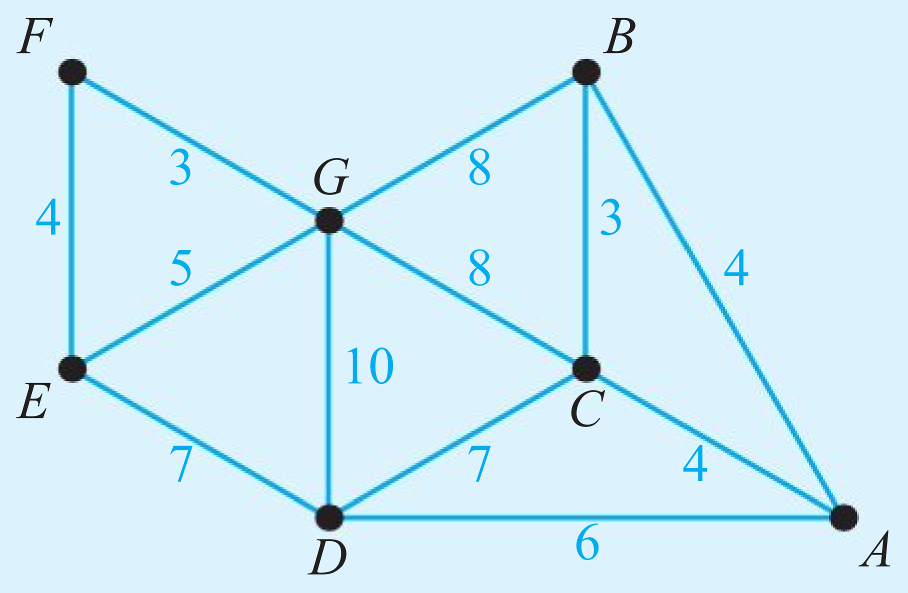


b

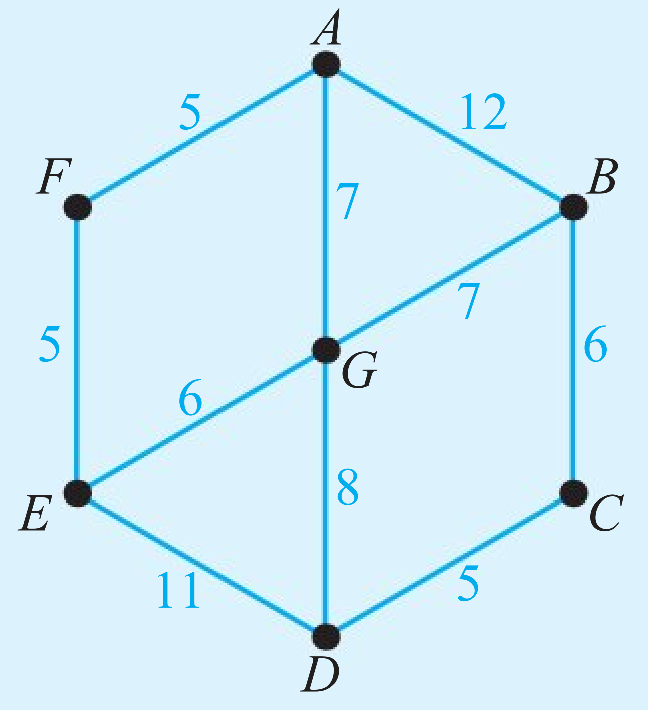


For questions 3 and 4, use the method demonstrated in Worked Example 7.21 to solve the Chinese postman problem for each graph. State which edges need to be repeated and the length of the shortest route.

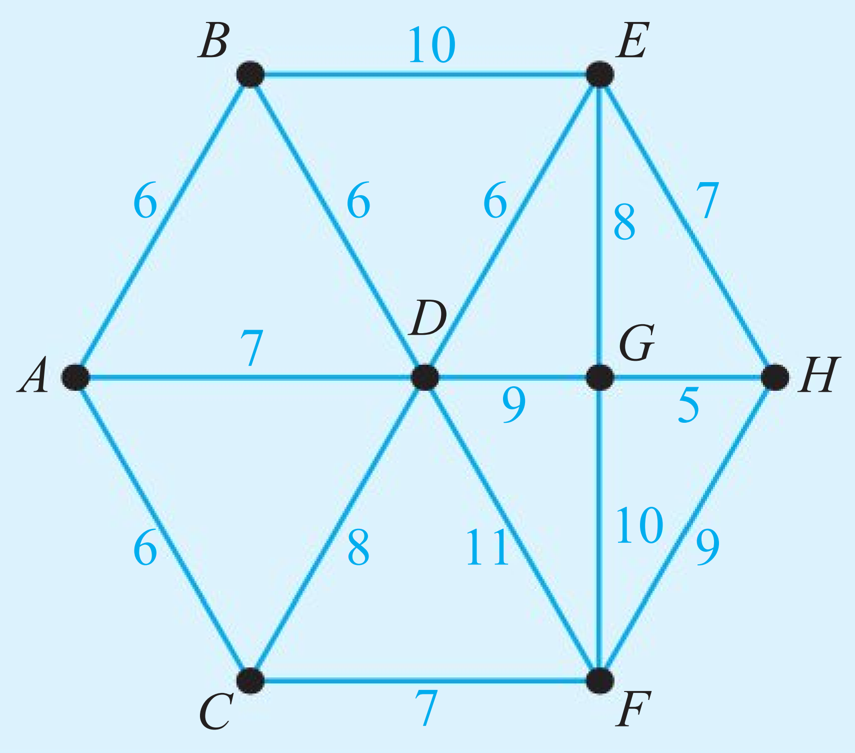
3 **a**



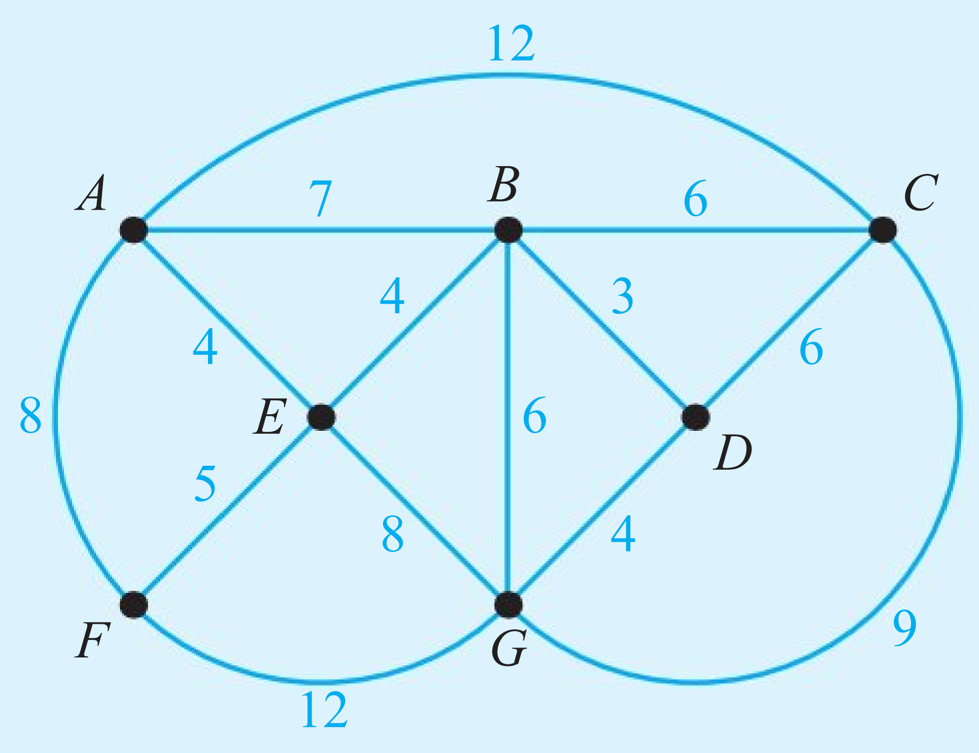
b



4 **a**



b



5 The graph shows a network of roads connecting seven villages.

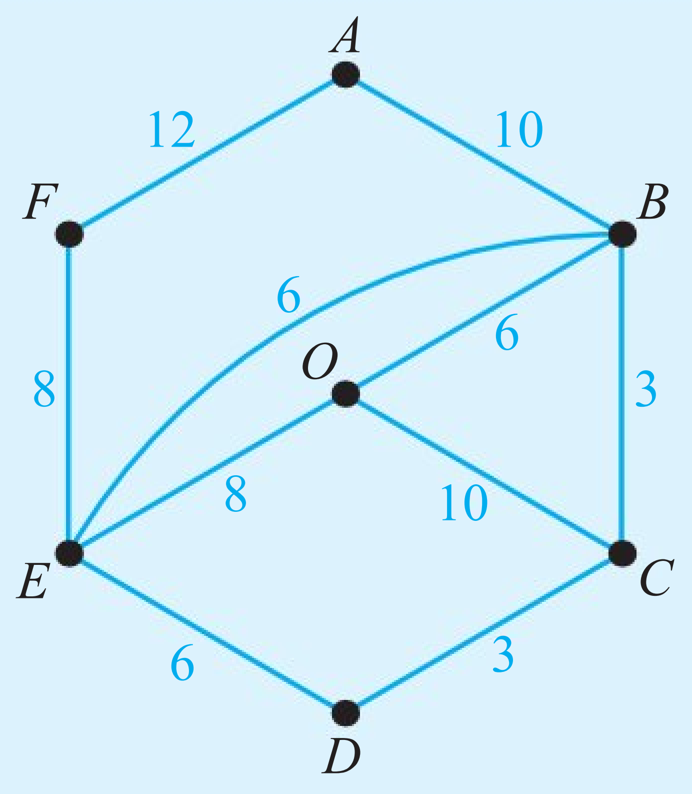
a Write down the two vertices of odd degree.

b State the length of the shortest path between the two vertices of odd degree.

A snow plough is located at village *C*. It needs to clear all the roads and return to *C*.

c Find the length of the shortest route the snowplough can take.

d Which roads does the snowplough need to use twice?



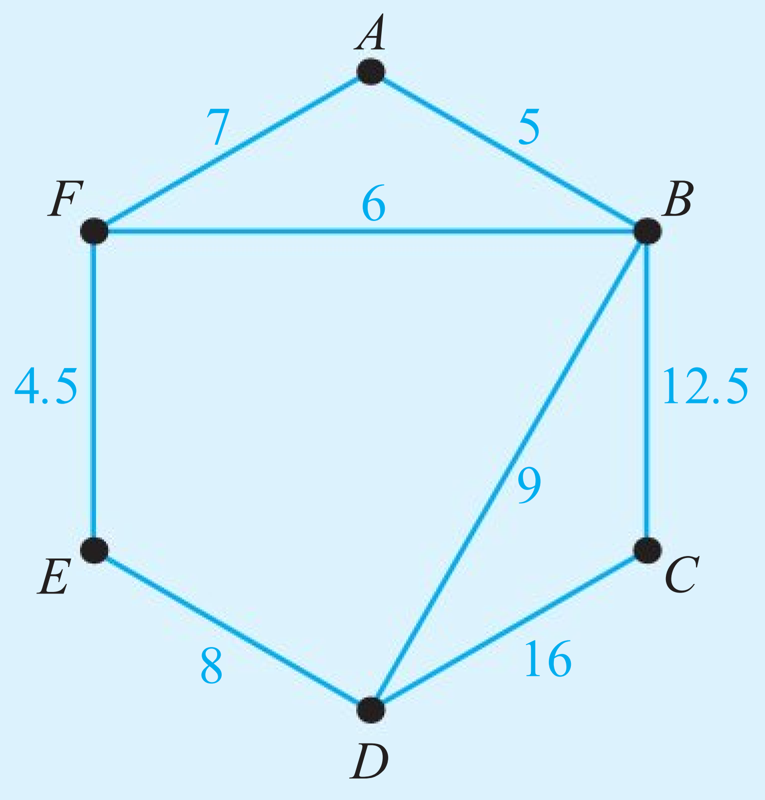
6 A post office is based at the point *A*. The graph shows the length of streets, in hundreds of metres, in a village. A postman needs to walk along each street at least once in order to deliver letters.

a The postman wishes to walk along each street *exactly once*, and can choose the start and end points (which do not need to be the same). State at which points he should start and end, and the length of the shortest possible route he can take.

b Explain why it is not possible for the postman to start at *A*, walk along each street exactly once, and return to *A*.

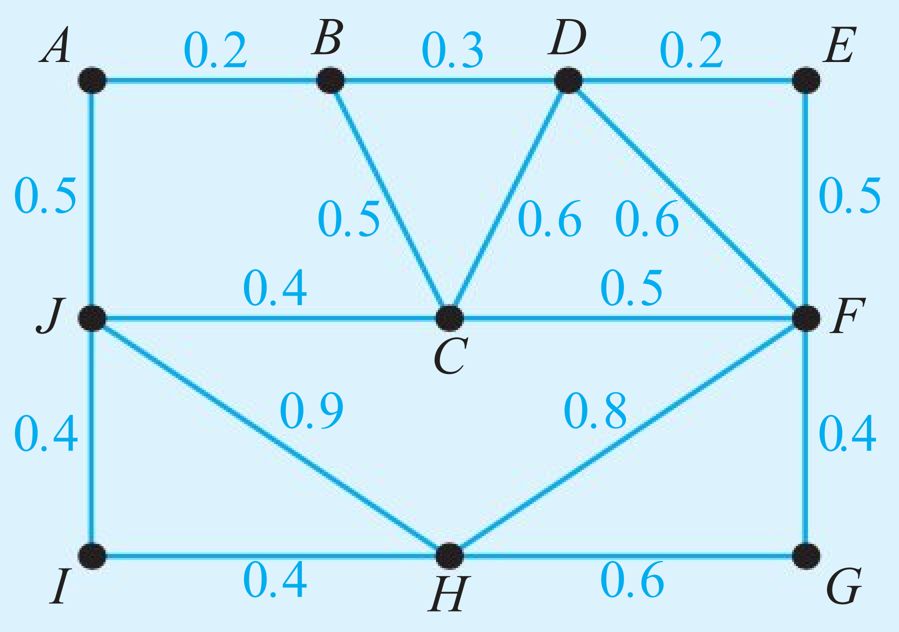
c Find the length of the shortest route which uses each street *at least once*, starting and finishing at the post office.

d In the route from part **c**, which streets are used more than once?

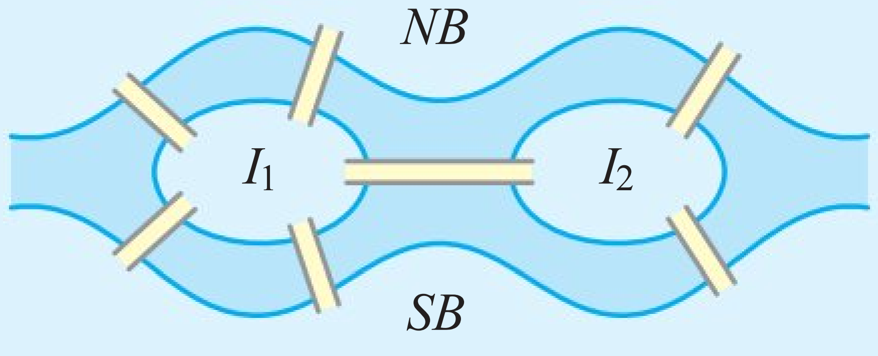


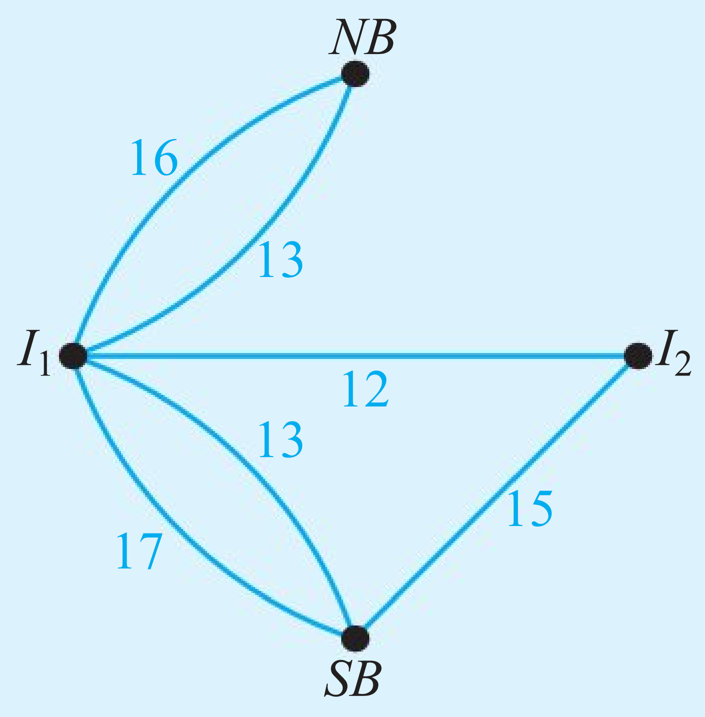
7 The graph represents the network of roads in a small town, with the weights of the edges representing the lengths of the roads, in kilometres. A salesman wants to visit all the houses in the town, so he must walk along each road at least once. He wants to start and finish at his shop, which is located at junction *C*.

Find the shortest possible route for the salesman and state its length.



8 The diagram shows the town of Königsberg with its two islands and seven bridges. The bridge between the north bank (*NB*) and the second island (I_2) is closed. In the corresponding graph, the weights of the edges represent the time, in minutes, to walk between different places.



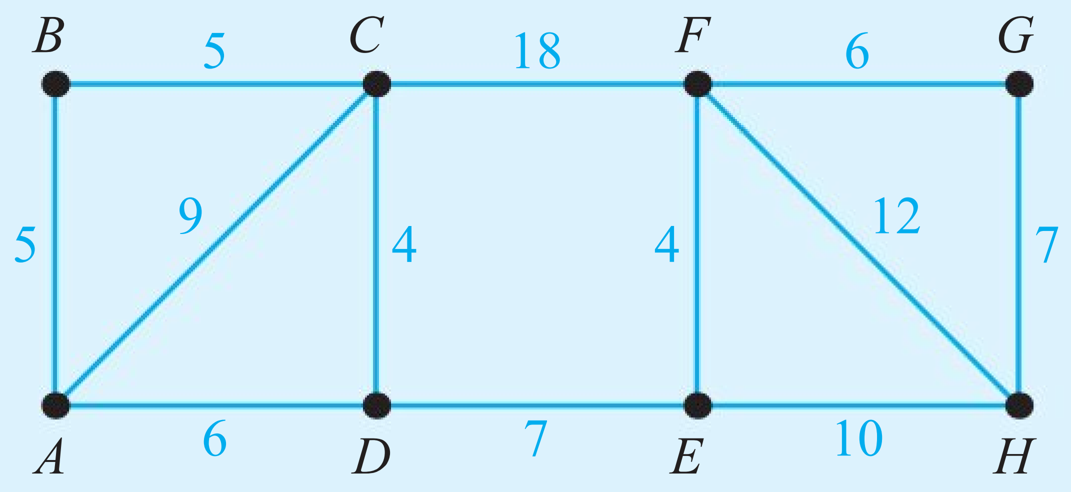


a A tour around the town starts at the north bank, crosses every bridge at least once (except for the closed one), and returns to the north bank. Find the shortest possible time taken by such a tour. You must make your method clear.

b The bridge between *NB* and I_2 reopens, and it takes 16 minutes to walk between the two places. Find the shortest possible time for a route which crosses every bridge at least once, starting and finishing at the north bank.

c State one factor you have ignored in your calculation.

9 The graph shows a network of paths in a park. The weights of the edges represent the number of minutes it takes to walk between the junctions.



A park ranger needs to inspect all the paths, starting and finishing at her hut, which is located at junction *A*. She wants to find the route that takes the shortest possible time.

a The ranger will need to walk along some of the paths twice. Determine which ones.

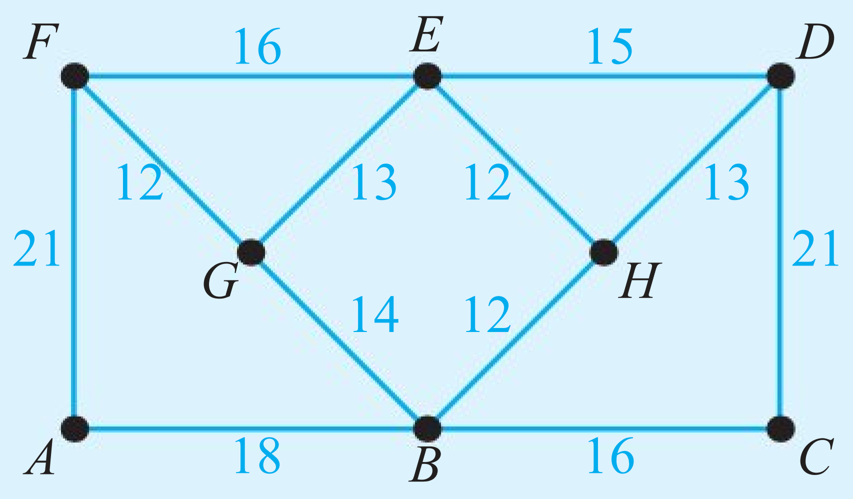
b Find the total time for the shortest possible route.

A tourist wants to walk along all paths in the park, starting at the hut at *A*, but finishing at the café which is located at *H*.

c Find the shortest time that the tourist can take.

d Find one possible route for the tourist.

10 The graph shows a network of paths between different sites within a school. The total length of all the paths is 183 m.



a Find the length of the shortest possible route around the school which starts and finishes at A and uses each path at least once.

b State which paths need to be used twice.

c Find the length of the shortest possible route which starts at D and finishes at F.

d A new path of length 16 m is built between sites G and H. Find the new length of the shortest route which starts and finishes at A and uses every path at least once.

11 Graph K is represented by the following weighted adjacency table.

	A	B	C	D	E	F
A	–	21	18	–	–	–
B	21	–	8	14	42	–
C	18	8	–	21	4	16
D	–	14	21	–	6	4
E	–	42	4	6	–	11
F	–	–	16	4	11	–

a Explain why K is not Eulerian.

b Find the shortest path between vertices C and F.

c Find the shortest route, starting and finishing at D, which uses each edge of graph K at least once. Make your method clear and state the length of your route.

d It is required to find the shortest route which uses each edge of graph K at least once, but does not need to return to the starting vertex.

i Find the least possible length of such a route.

ii What starting and finishing vertices should be used to achieve this shortest route?

7F The travelling salesman problem

The travelling salesman problem is to find the shortest route around a graph which visits each vertex at least once and returns to the starting point. Think about a salesman who wants to visit every town in a region.

You met complete graphs in Section A and Hamiltonian cycles in Section B.

The classical travelling salesman problem

For the simplest version of the travelling salesman problem, assume that the graph is complete. Assume, further, that the shortest route between two vertices is the direct route (we say that the graph satisfies the 'triangle inequality'). Such a graph has several Hamiltonian cycles (which visit each vertex *exactly* once), so the problem becomes finding the shortest one.

One way to solve the classical travelling salesman problem is to list all Hamiltonian cycles and find their weights. This is guaranteed to find the shortest cycle, but is inefficient, or may not be feasible, for large graphs. For example, a complete graph with five vertices has 12 different Hamiltonian cycles, but a complete graph with ten vertices has 181 440.

It turns out that this problem is a lot more difficult than it might at first appear. In fact, there is no known algorithm which guarantees finding the shortest Hamiltonian cycle. The best approach is to find **upper and lower bounds** for the problem. These are two numbers such that we can guarantee that the shortest possible Hamiltonian cycle has a length that is somewhere between them.

Nearest neighbour algorithm for determining an upper bound

Any Hamiltonian cycle will provide an upper bound (UB) – the shortest cycle cannot be any longer. The following algorithm finds a good upper bound by selecting the shortest edge at each stage and then returning to the starting point.

KEY POINT 7.11

Nearest neighbour algorithm for an upper bound:

- Pick a starting vertex.
- Go to the closest vertex not yet visited.
- Repeat until all the vertices have been used.
- Add the final edge to return to the starting vertex.

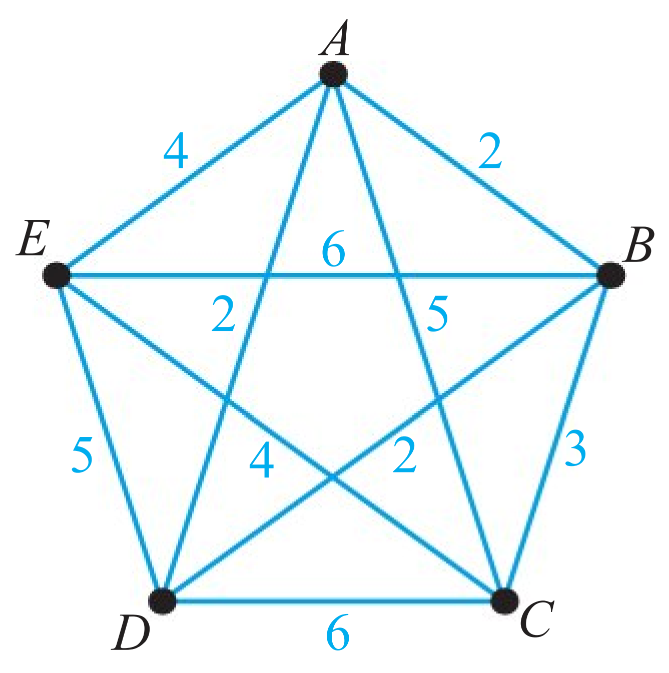
You want the upper bound to be as small as possible. Starting at a different vertex may give a better upper bound.

WORKED EXAMPLE 7.22

Consider this graph.

a Use the nearest neighbour algorithm starting at B to find an upper bound for the travelling salesman problem.

b What does your answer to part **a** tell you about the shortest path that visits each vertex and returns to the starting point?



The vertex closest to B is D

a Edges to include:

$BD(2)$

The vertex closest to D and different from B is A

$DA(2)$

Continue until you have visited all the vertices

$AE(4)$

$EC(3)$

Finally, return to B directly from C

$CB(3)$

The length of the cycle $BDAECB$ gives an upper bound

$\therefore\ UB = 15$

The shortest cycle can't be longer than the upper bound

b It has length $\leqslant 15$.

Deleted vertex algorithm for determining a lower bound

A lower bound (*LB*) is a length such that the shortest Hamiltonian cycle is definitely at least that long.

KEY POINT 7.12

Deleted vertex algorithm for a lower bound:

- Remove one vertex and all associated edges.
- Find the length of the minimum spanning tree for the remaining graph.
- Add the two shortest edges that were originally connected to the removed vertex.

You want the lower bound to be as large as possible. Removing a different vertex may give a better lower bound.

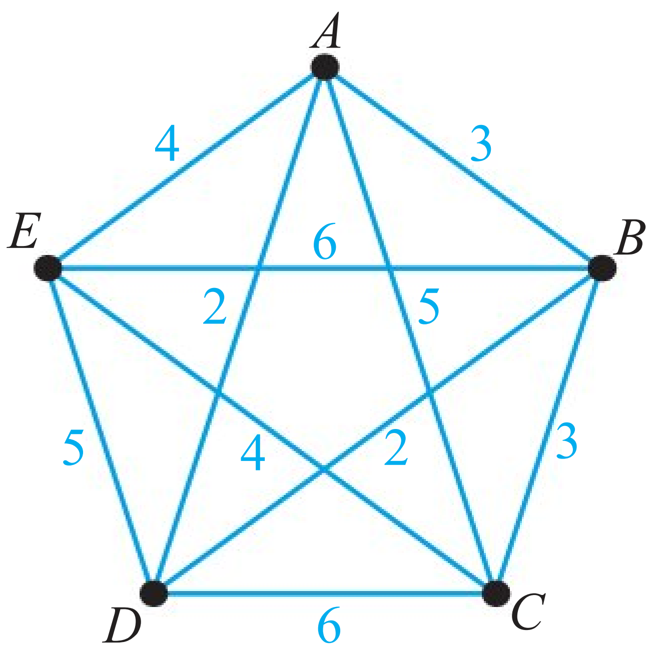
WORKED EXAMPLE 7.23

Consider this graph.

a By deleting vertex A, find a lower bound for the travelling salesman problem.

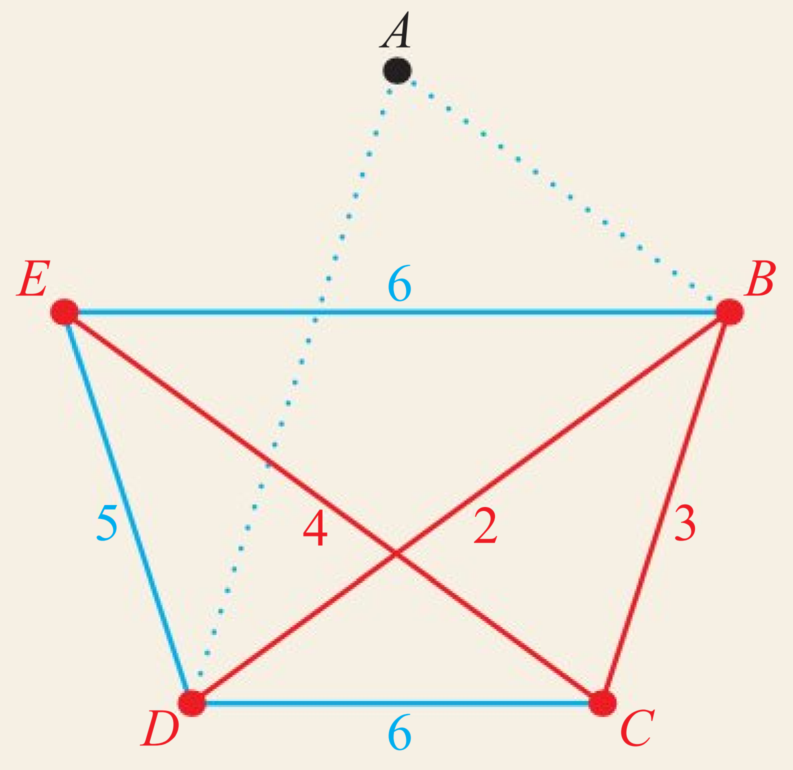
b By deleting vertex C, find a different lower bound.

c Which of the two lower bounds is better?



Use Kruskal's algorithm to find the minimum spanning tree for the graph with vertices B, C, D and E: add edges starting with the shortest

a Remove A:



Minimum spanning tree:
BD (2), BC (3), CE (4)

Tip

Notice that these five edges in part **a** do not form a cycle. This will often happen when finding a lower bound.

Add the two shortest edges from A

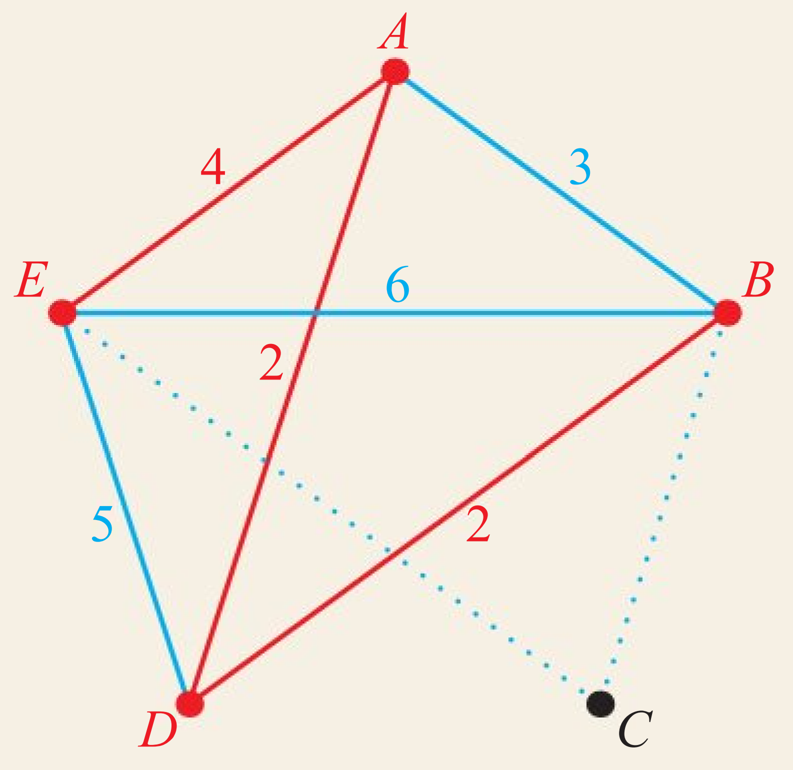
Shortest edges from A:
$AD(2)$, $AB(3)$

The total length gives a lower bound

$$LB = (2+3+4)+(2+3) = 14$$

Now find the minimum spanning tree for the graph with vertices A, B, D E

b Remove C:



Minimum spanning tree:
$AD(2)$, $BD(2)$, $AE(4)$

Add the two shortest edges from C

Shortest edges from C:
$BC(3)$, $BE(4)$

$$LB = (2+2+4)+(3+4) = 15$$

You want the lower bound to be as large as possible

c The lower bound of 15 is better.

In Worked Examples 7.22 and 7.23 you found that both upper and lower bounds are 15. This means that the lower bound is achievable, so the shortest Hamiltonian cycle has length 15. One possible such cycle is $BDAECB$, found in Worked Example 7.22.

KEY POINT 7.13

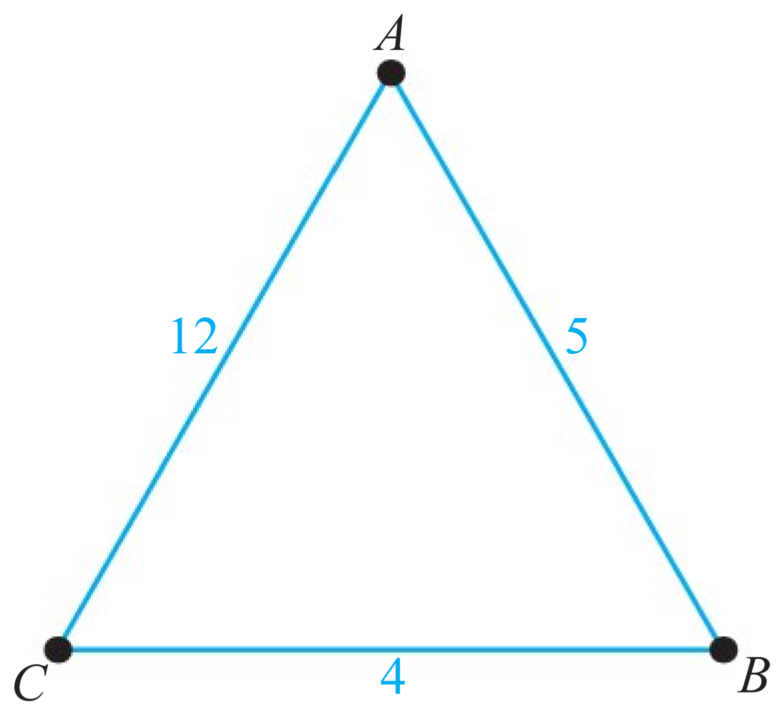
- The solution to the travelling salesman problem lies between the lower and the upper bounds.
- If $LB = UB$, then this is also the length of a shortest cycle.
- If you have found a cycle of the same length as the LB, then this is a shortest cycle.

You are the Researcher

There is no efficient algorithm for solving the travelling salesman problem. We say that the problem is 'NP-hard'. Investigate what this means and find examples of some other problems that fall into this category.

Practical problems

In a practical situation, the graph may not be complete, which means that it may not have a Hamiltonian cycle at all. Furthermore, a graph may not satisfy the triangle inequality, meaning that the shortest route between two vertices is not always the direct one.



In this graph, the shortest route between A and C is ABC (9), not AC (12)

In both of these cases, some vertices may need to be visited more than once. You can convert any practical problem into the classical problem by drawing a new graph which is complete and shows the shortest distance between any two vertices.

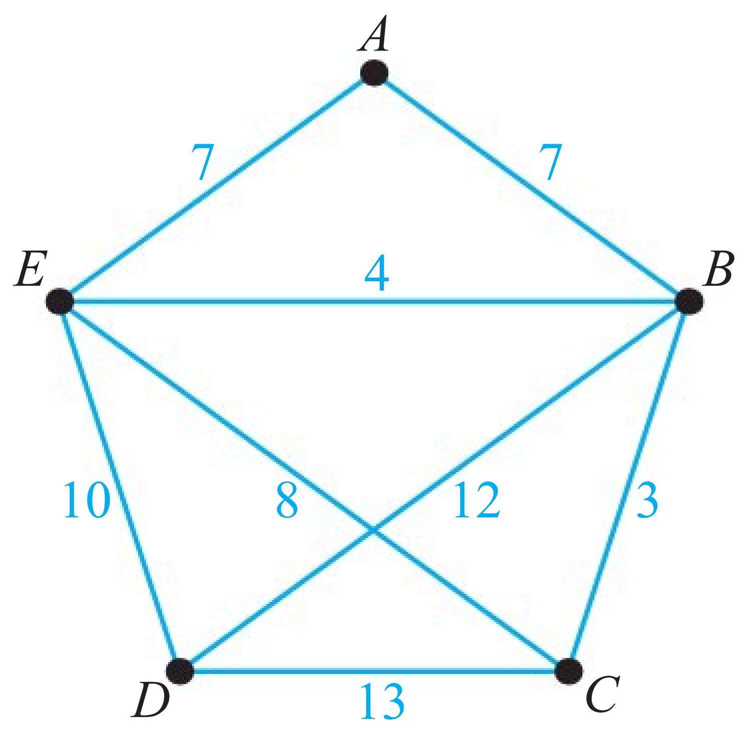
WORKED EXAMPLE 7.24

Look at this graph.

a Construct a table showing the shortest distances between the vertices.

b Use the nearest neighbour algorithm starting at B to find an upper bound for the travelling salesman problem on the new graph.

c Find the corresponding route in the original graph and state its length. Which vertices are visited twice?



The graph is not complete – it is missing edges AC and AD

a The shortest distance between A and C is 10 (ABC).
The shortest distance between A and D is 17 (AED).

The shortest distance between C and E is not the direct edge

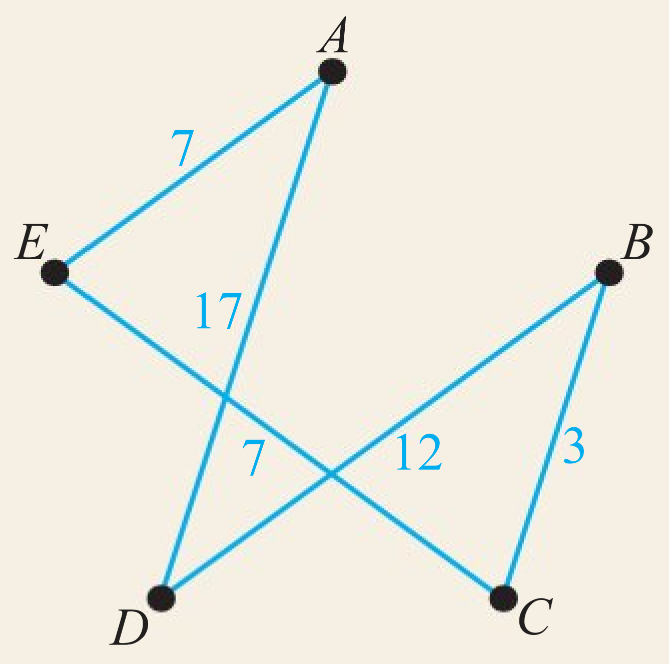
The shortest distance between C and E is 7 (CBE).

The other shortest distances are given by the edges

	A	B	C	D	E
A	–	7	10	17	7
B	7	–	3	12	4
C	10	3	–	13	7
D	17	12	13	–	10
E	7	4	7	10	–

You can perform the nearest neighbour algorithm on the table. Draw the graph as you go along to ensure that you have visited each vertex exactly once

b BC (3)
CE (7)
EA (7)
AD (17)
DB (12)
Hence, UB = 46



The above route uses the edge AD, which did not exist in the original graph, which needs to use A-E-D instead

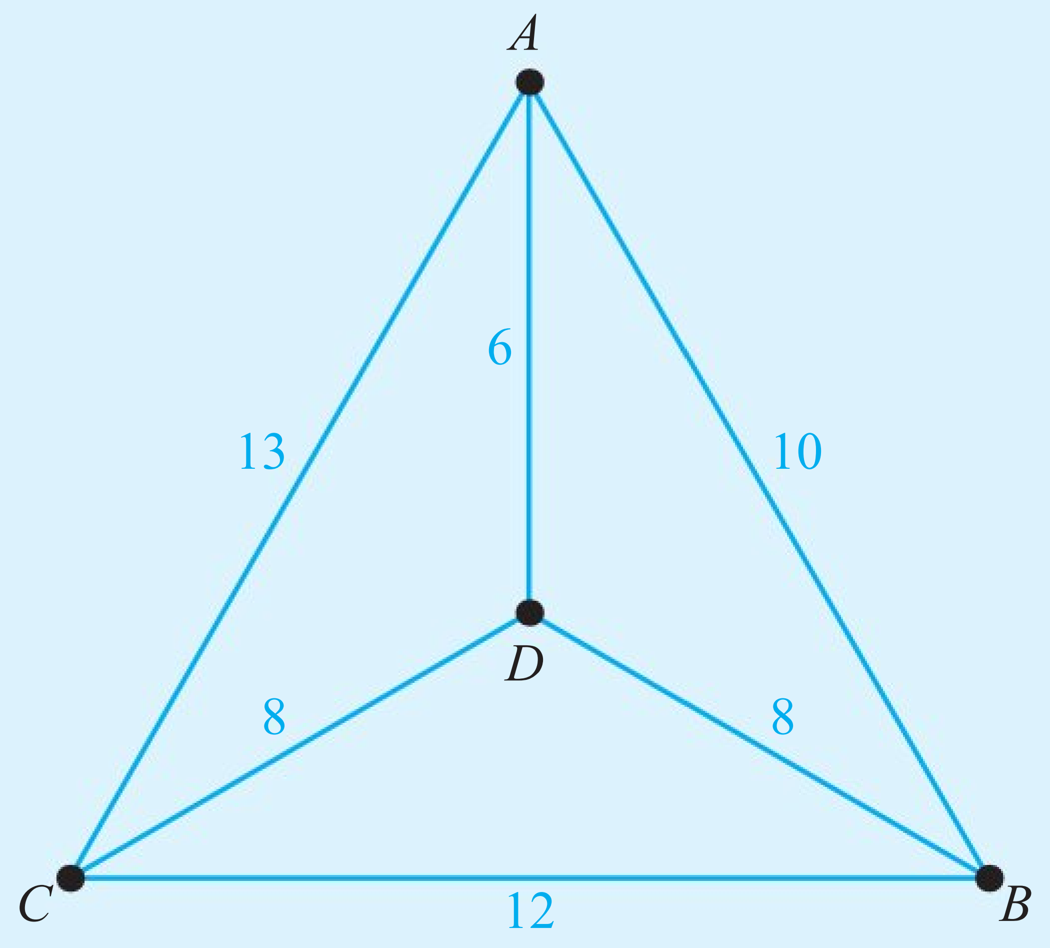
The distance of 7 between C and E was achieved by C-B-E

c The corresponding route, of length 46, in the original graph is:
B-C-B-E-A-E-D-B
Vertices B and E are visited twice.

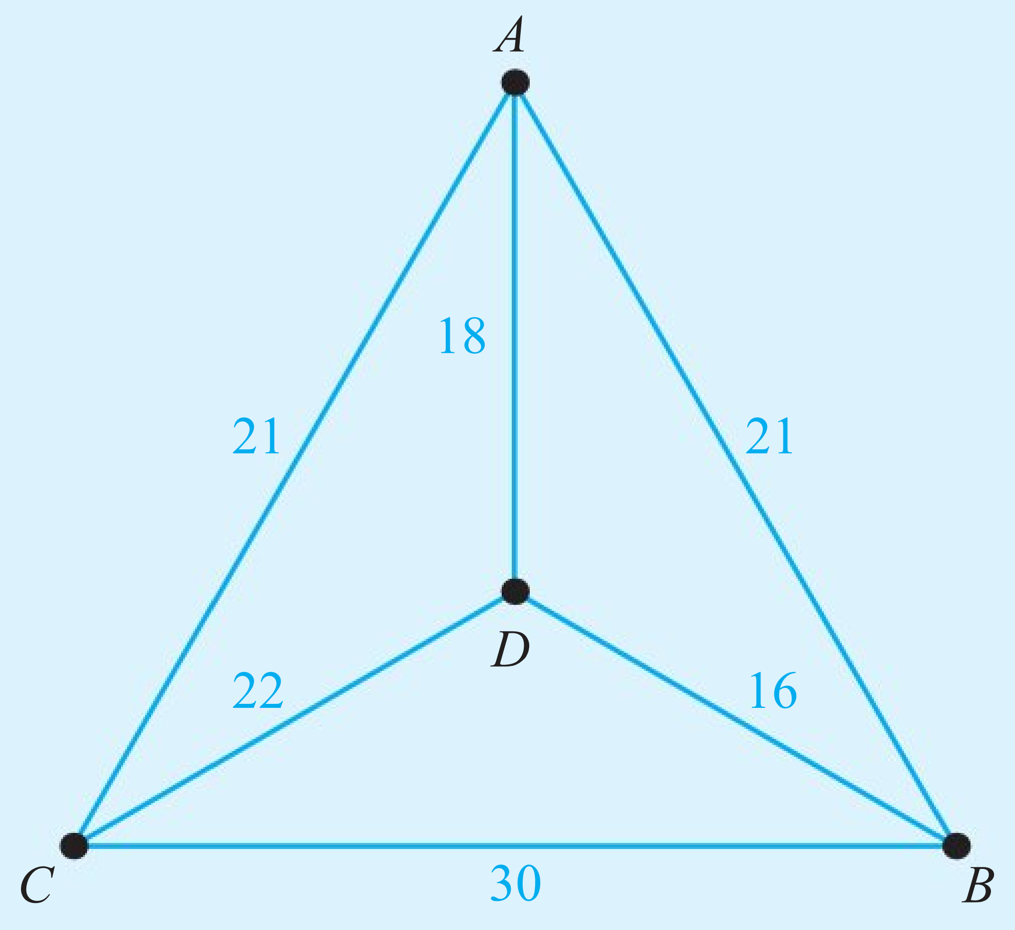
Exercise 7F

For questions 1 to 3, use the nearest neighbour algorithm starting at A, as illustrated in Worked Example 7.22, to find an upper bound for the travelling salesman problem for each graph.

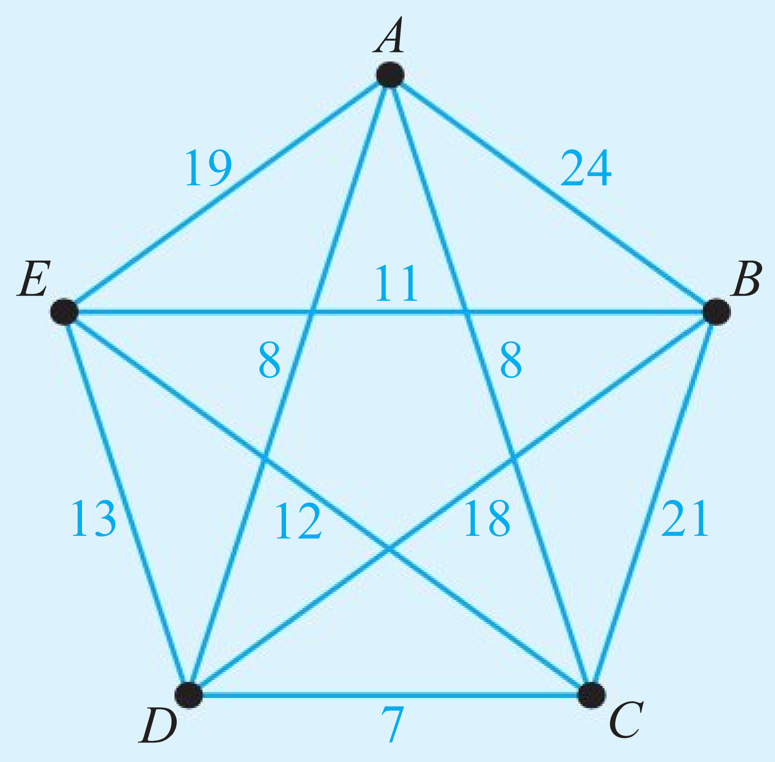
1 **a**



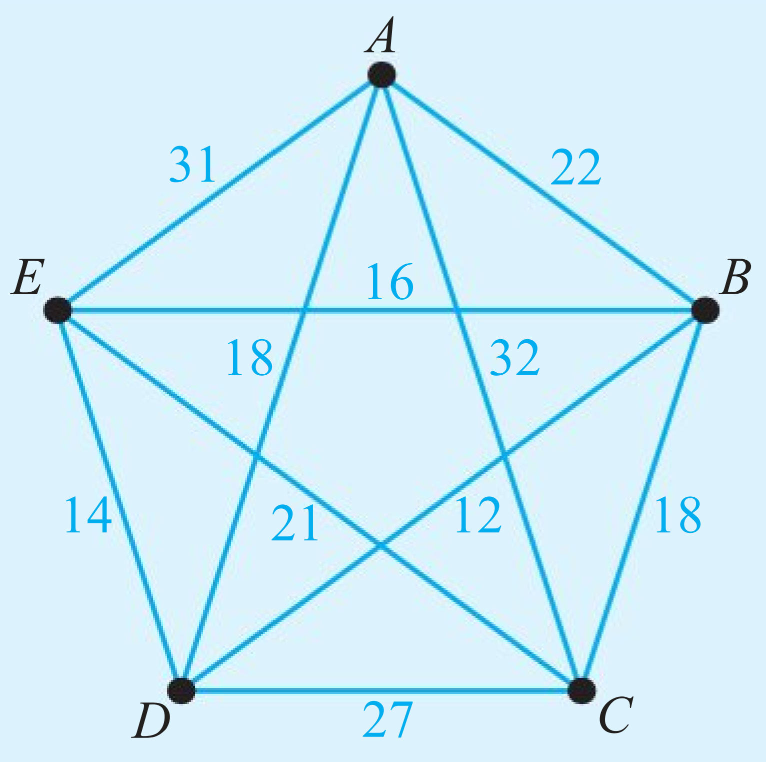
b



2 **a**



b



3 **a**

	A	**B**	**C**	**D**	**E**
A	–	32	28	26	19
B	32	–	14	21	26
C	28	14	–	18	18
D	26	21	18	–	22
E	19	26	18	22	–

	A	**B**	**C**	**D**	**E**
A	–	22	21	17	22
B	22	–	22	23	31
C	21	22	–	18	31
D	17	23	18	–	26
E	22	31	31	26	–

For questions 4 to 6, use the deleted vertex algorithm, as illustrated in Worked Example 7.23, to find a lower bound for the travelling salesman problem for each of the graphs from questions 1 to 3. The vertex to be removed is given in each question.

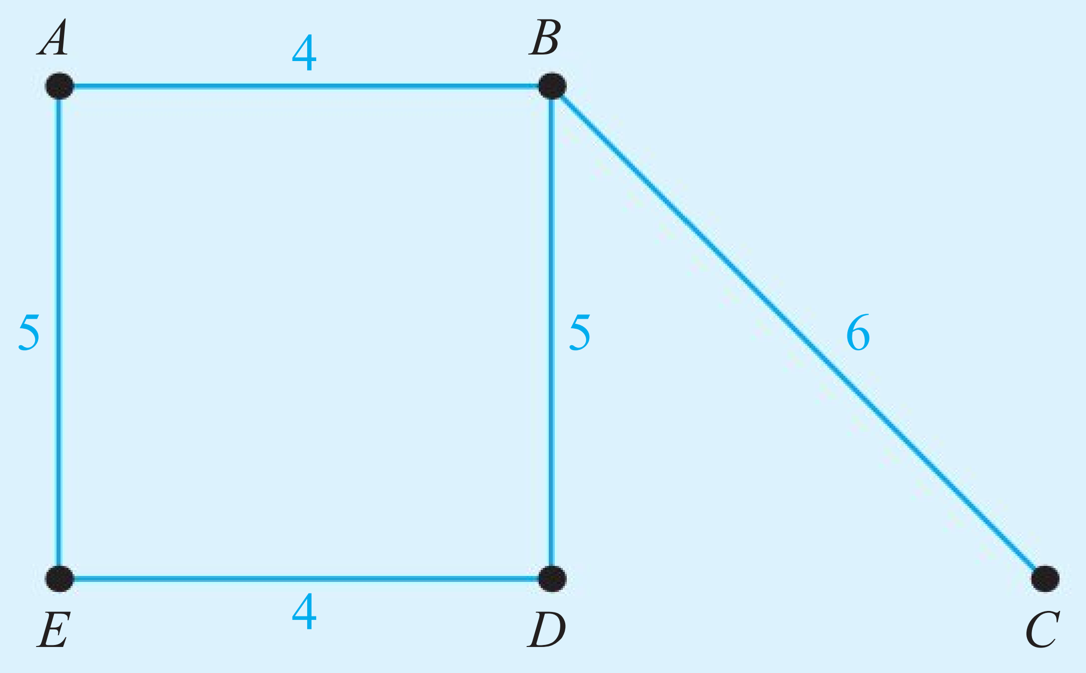
4 **a** Question 1a, vertex A
b Question 1b, vertex A

5 **a** Question 2a, vertex B
b Question 2b, vertex B

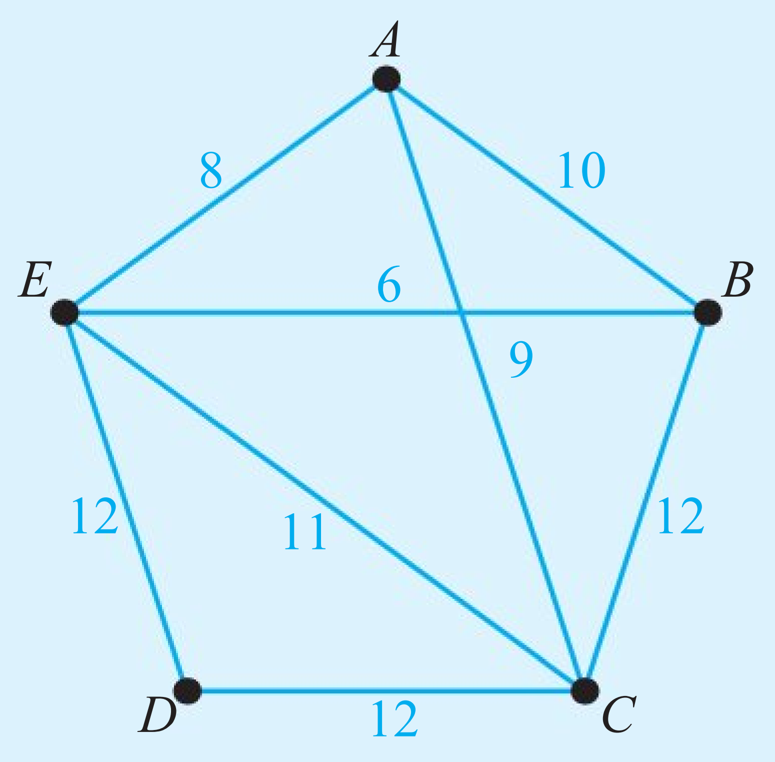
6 **a** Question 3a, vertex E
b Question 3b, vertex E

For questions 7 to 9, use the method demonstrated in Worked Example 7.24a to construct the table of shortest distances for each graph.

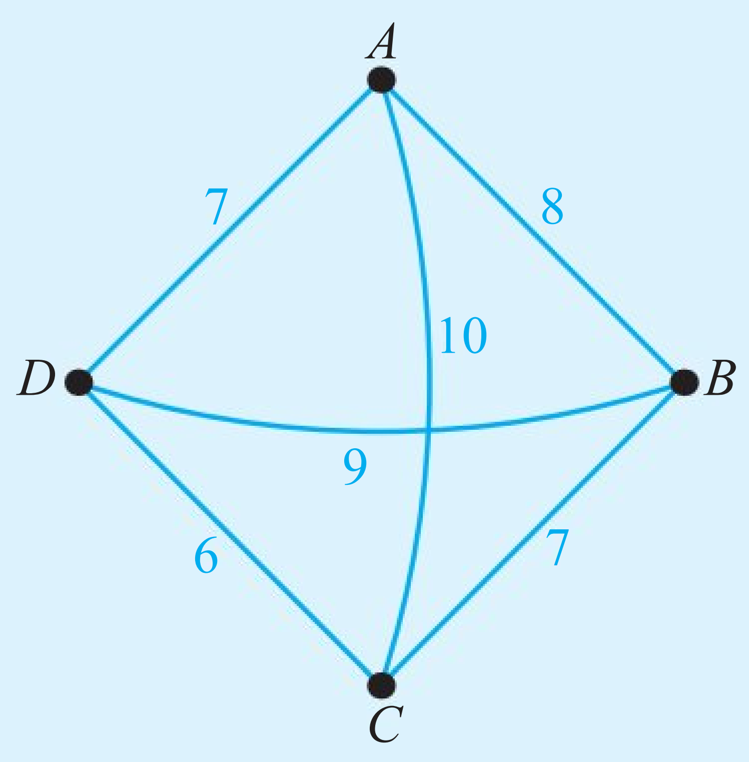
7 **a**



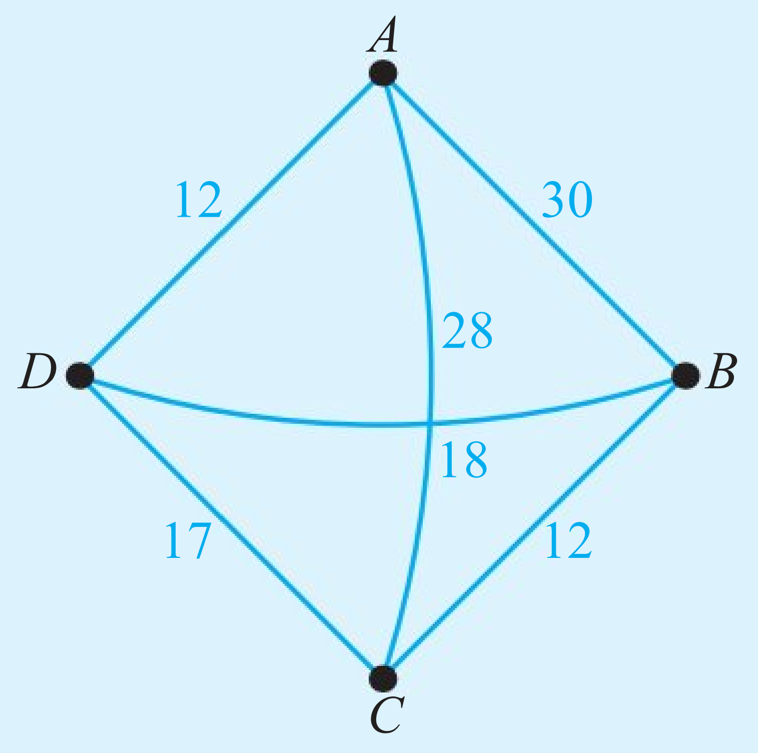
b



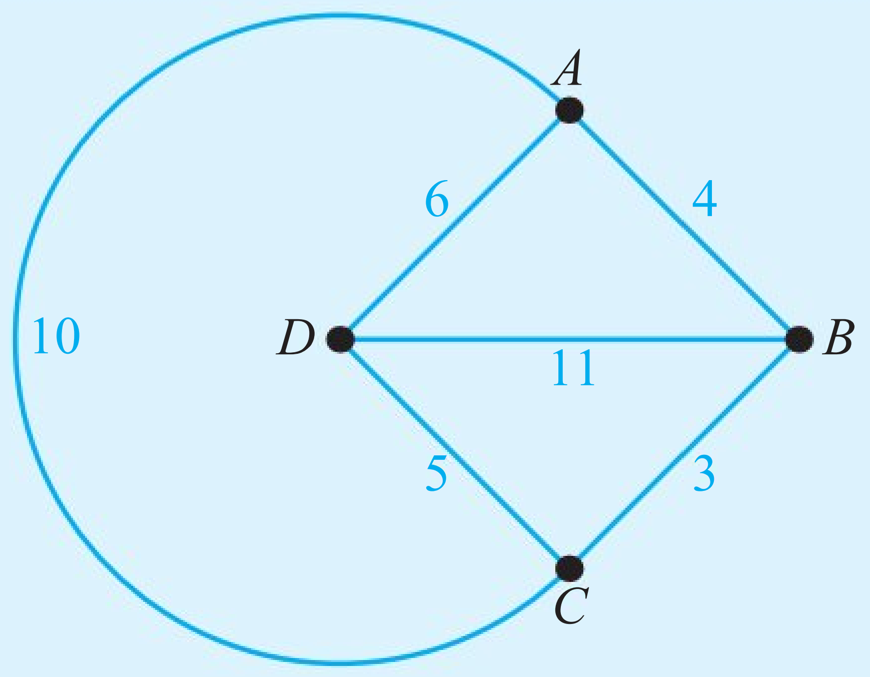
8 **a**



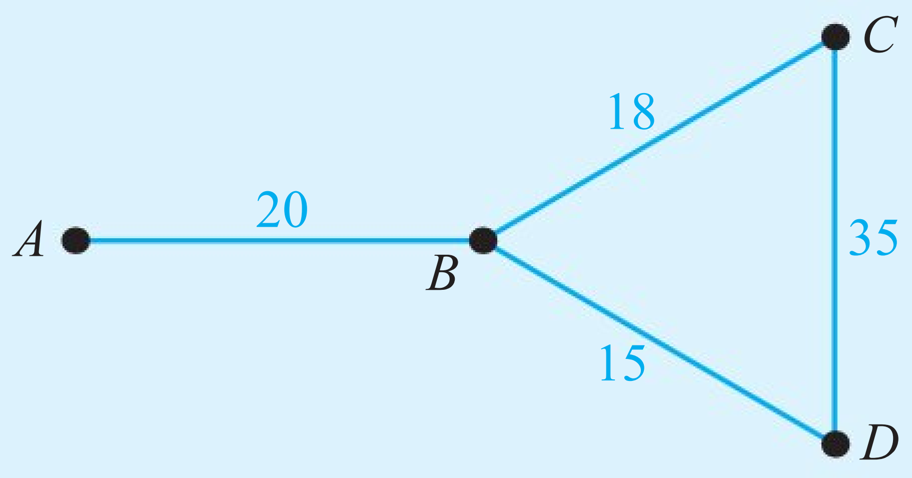
b



9 **a**



b



10 Graph G is given by the following cost adjacency matrix.

	A	**B**	**C**	**D**
A	–	16	12	8
B	16	–	18	8
C	12	18	–	9
D	8	8	9	–

a List all distinct Hamiltonian cycles starting and finishing at A. You only need to give each cycle in one direction.

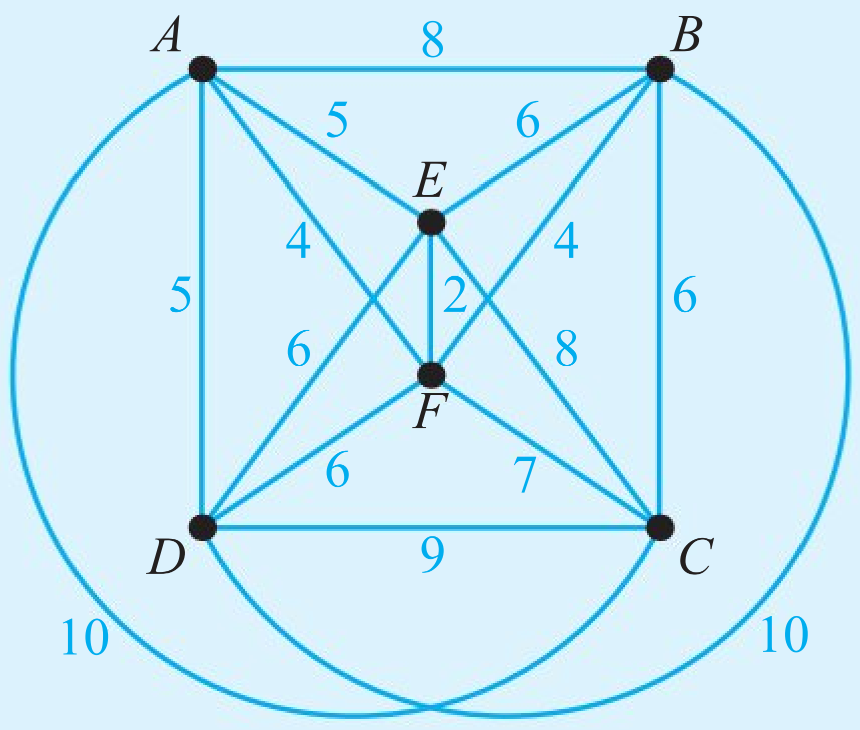
b Hence solve the travelling salesman problem for graph G.

11 The graph alongside shows six places of interest in a nature reserve and a network of paths connecting them. The distances are given in kilometres.

a Use the nearest neighbour algorithm starting at A to find an upper bound for the travelling salesman problem for this graph.

b Use the nearest neighbour algorithm starting at B to find a different upper bound.

c A visitor wishes to visit all six sites and return to the starting point. What can you conclude about the length of the shortest possible route they can use?



12 The table shows the times, in minutes, required to walk between five different stores on a high street. A shopper wants to visit all five stores and return to the starting point.

	A	**B**	**C**	**D**	**E**
A	–	4	3	7	6
B	4	–	3	9	7
C	3	3	–	8	5
D	7	9	8	–	9
E	6	7	5	9	–

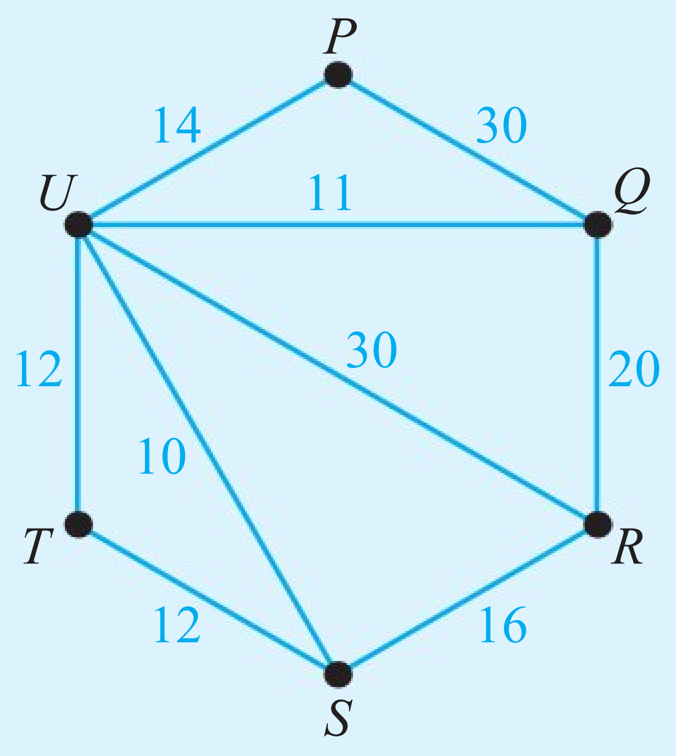
a Explain how you know that the fastest possible route takes at most 29 minutes.

b Use the nearest neighbour algorithm, starting with each vertex in turn, to find an improved upper bound.

c By deleting vertex D from the graph, find a lower bound.

d What can you conclude about the shortest possible time required to visit all five stores and return to the starting point?

13 The diagram shows a network of roads connecting six towns, with distances given in kilometres. A delivery company is based at Q. They want to find the shortest possible route that visits every town and returns to Q.



a Explain why the shortest route between P and R is 40 km long.

b Find the shortest route between P and Q and state its length.

c Copy and complete this table showing the shortest distance between each pair of towns.

	P	**Q**	**R**	**S**	**T**	**U**
P	–	–	40	–	–	14
Q	–	–	20	21	23	11
R	40	20	–	–	–	26
S	–	21	–	–	–	10
T	–	23	–	–	–	12
U	14	11	26	10	12	–

d Use the nearest neighbour algorithm starting at Q to find an upper bound for the shortest possible route.

e By deleting vertex Q, find a lower bound for the length of the shortest possible route.

14 Consider the graph given by the following cost adjacency matrix.

	A	**B**	**C**	**D**	**E**	**F**
A	–	5	10	11	8	7
B	5	–	6	7	8	7
C	10	6	–	7	10	12
D	11	7	7	–	9	10
E	8	8	10	9	–	6
F	7	7	12	10	6	–

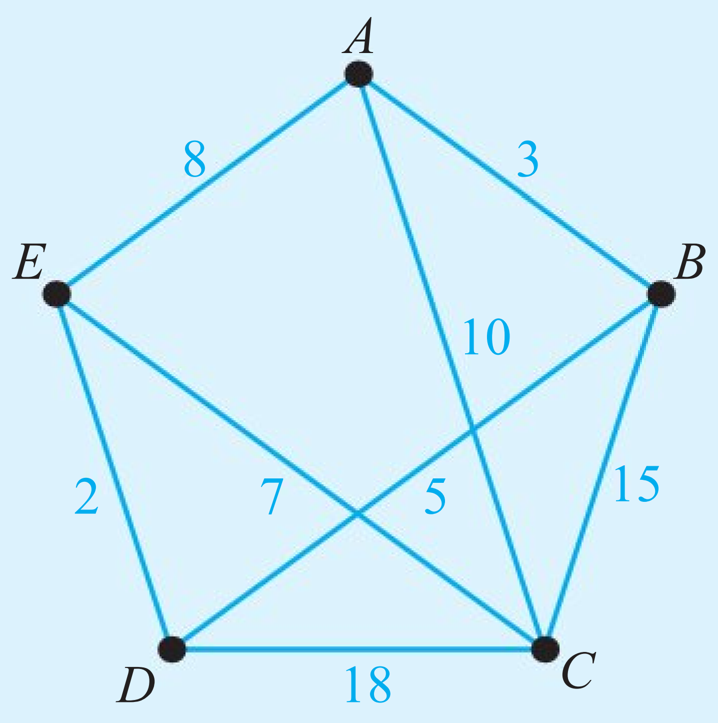
a Use nearest neighbour algorithm starting at A to find an upper bound for the travelling salesman problem.

b By deleting vertex B and using Kruskal's algorithm to find the minimum spanning tree for the resulting graph, find a lower bound for the travelling salesman problem.

c Explain how you know that you have found a solution to the travelling salesman problem, and write down the weight of the optimal route.

d Write down one possible Hamiltonian cycle of shortest length.

15 The graph shows the times, in minutes, to walk between five houses. Some of the houses are not connected by direct paths.



a State the quickest possible routes, and their lengths, between the following pairs of houses.

i B and E

ii B and C

iii C and D

b Draw a complete graph showing the times of the quickest routes between the houses.

A student lives in house C and wants to deliver party invitations to her friends at the other four houses, starting and ending at her own house.

c Use the nearest neighbour algorithm starting at C to find an upper bound for the minimum time she can take.

d By deleting vertex C, find a lower bound.

e By considering the options for the outward and return path connecting to C, explain how you know that this lower bound cannot be achieved for a path beginning and ending at C.

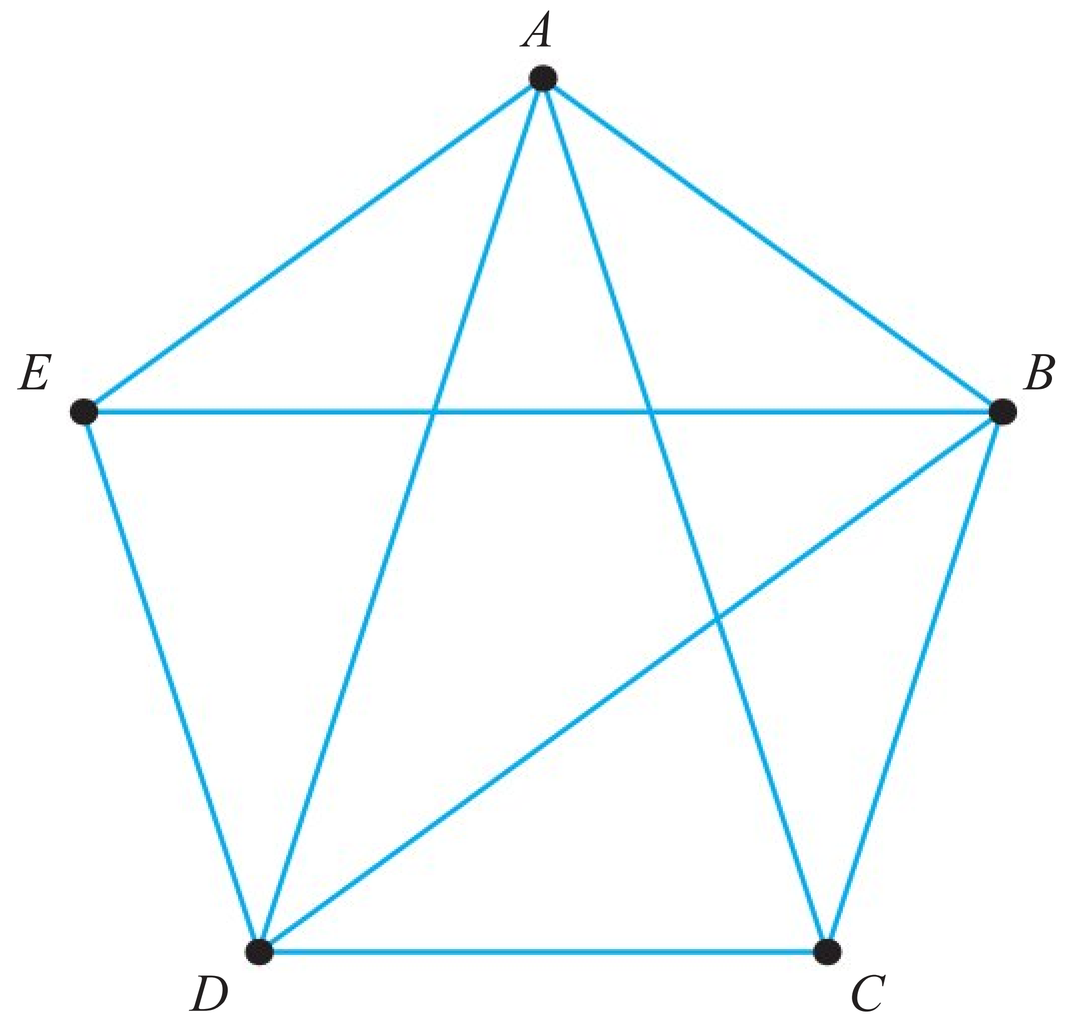
f Find the quickest possible route for the original graph and state its length.

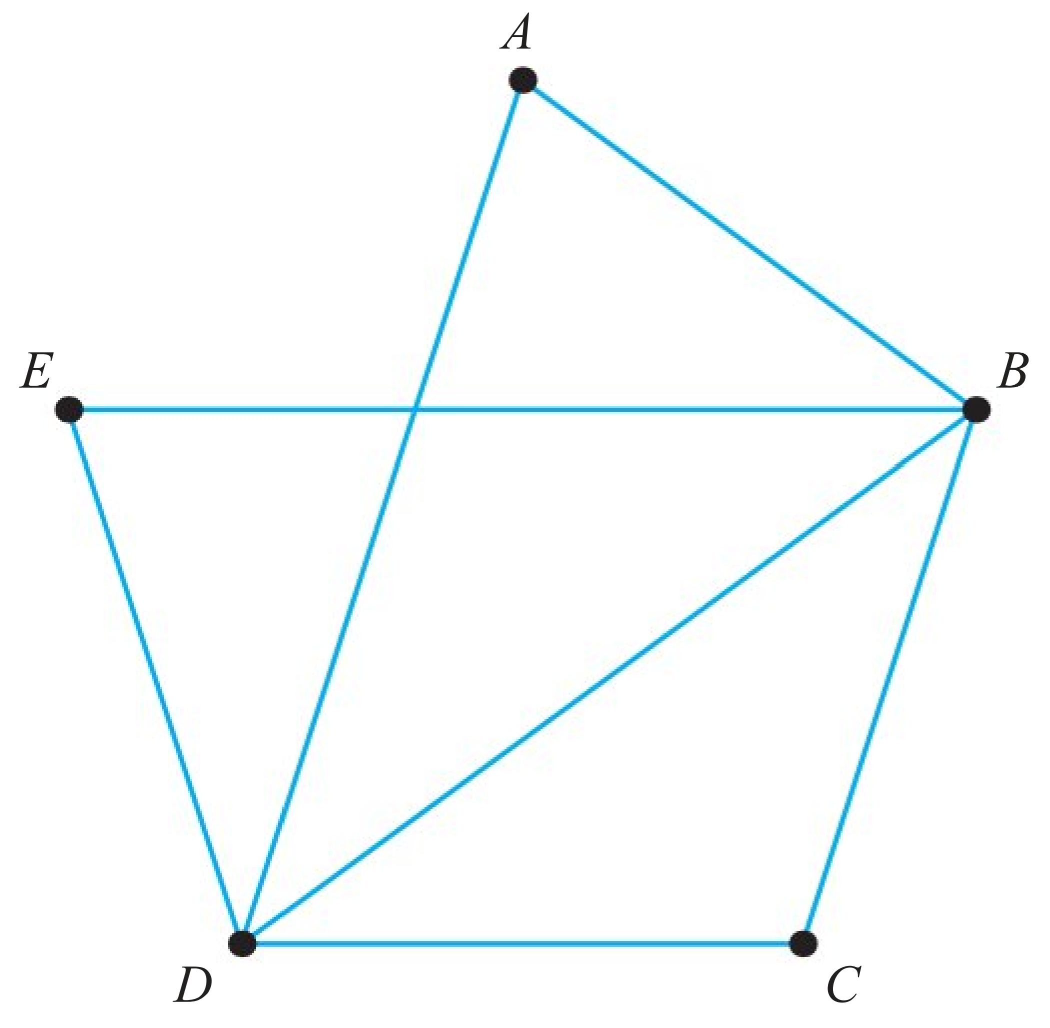
Checklist

- You need to be able to distinguish between undirected and directed, and unweighted and weighted graphs, and be able to classify graphs as simple, connected (or strongly connected), complete or trees.
- You need to be able construct an adjacency matrix for a graph and use it to
  - find the degree of each vertex
  - find the number of walks between two vertices.
- You need to be able to construct a transition matrix for a graph and use it to
  - find the probability of being at a certain vertex after a given number of steps
  - determine long-term probabilities in a random walk
  - use the PageRank algorithm to rank web pages in order of importance.
- You need to be able to identify various types of walks in a graph. In particular:
  - An Eulerian trail uses each edge exactly once, and an Eulerian circuit then returns to the starting point.
  - A graph is Eulerian if all vertices have even degree.
  - A Hamiltonian graph has a Hamiltonian cycle, which visits each vertex exactly once and returns to the starting point.
- You need to be able to find a minimum spanning tree of a weighted graph. This is a subgraph of smallest possible weight which is also a tree.
  - Kruskal's algorithm for finding a minimum spanning tree involves adding edges in order of weight and skipping edges which would form a cycle.
  - Prim's algorithm for finding a minimum spanning tree involves choosing a starting vertex and then adding vertices one at a time so that we always join an unconnected vertex to a connected vertex using the edge of smallest possible weight.
  - When using Prim's algorithm in a table, the labels for each vertex are recorded in a box, with permanent labels in the top row.
- You need to understand the Chinese postman problem for finding the shortest route around a graph which uses each edge at least once and returns to the starting vertex.
  - If the graph is Eulerian, the solution is any Eulerian path.
  - If the graph is not Eulerian, we need to find the shortest path between pairs of vertices odd degree and repeat those edges.
- You need to understand the travelling salesman problem for finding the Hamiltonian cycle of least weight in a complete graph (i.e. the shortest route which visits every vertex and returns to the starting point).
  - If the graph is not complete, you need to complete it by creating a table of shortest distances.
  - The exact solution can only be found by checking all Hamiltonian cycles, which is inefficient.
  - You need to know methods for finding upper and lower bounds (the nearest neighbour algorithm and the deleted vertex algorithm).
  - The required least weight of a Hamiltonian cycle is somewhere between the lower bound and the upper bound, $LB \leqslant L \leqslant UB$. You need to make the lower bound as large as possible and the upper bound as small as possible.

Mixed Practice

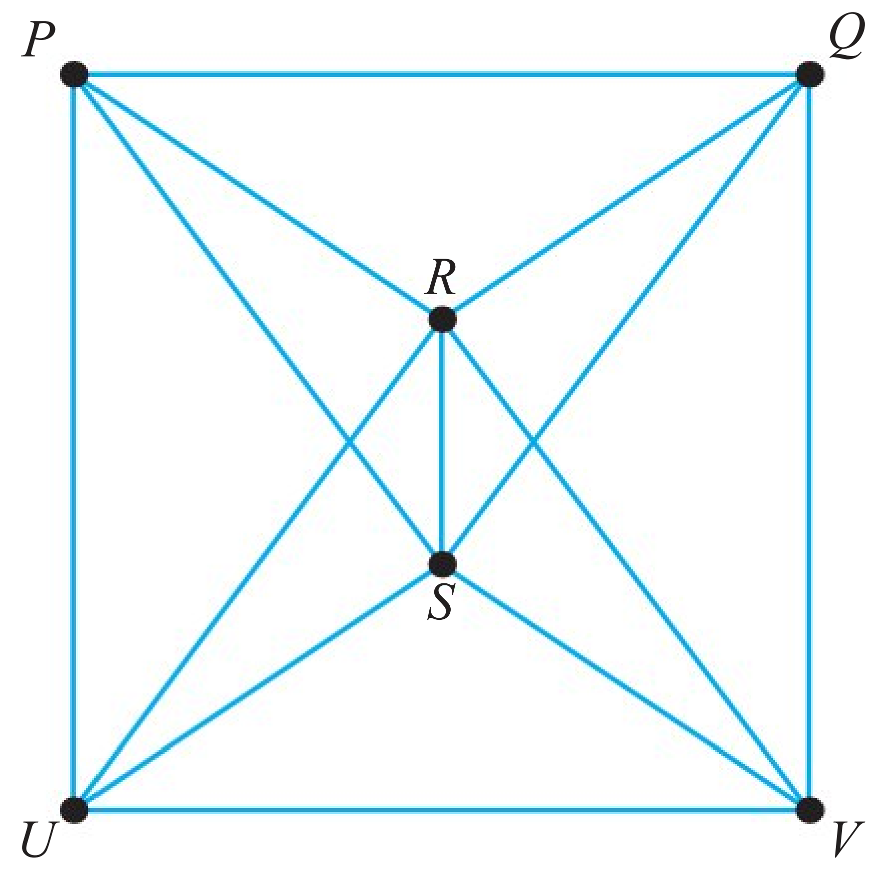
1 The two graphs below show a network of train lines connecting five cities. Sachin wants to use each line exactly once, starting and finishing at the same point.



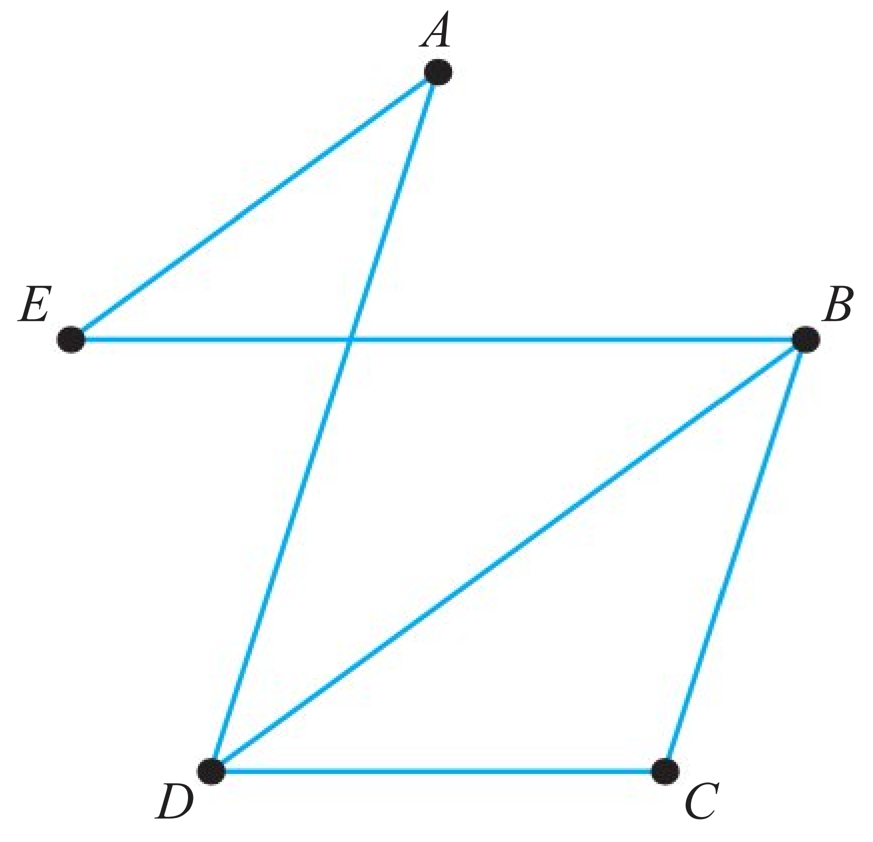


a For which of the two networks is it possible to do this? Justify your answer.
b Find a possible route Sachin could take, starting and finishing at A.
c For the other network, Sachin still wants to use each line exactly once, but can start and finish at different cities. At which city could he start?

2 Graph K is shown in the diagram.
a Which edge should be removed from K so that the resulting graph is Eulerian?
b Find a spanning tree for K.
c Show that K has a Hamiltonian cycle.



3 Consider this graph.

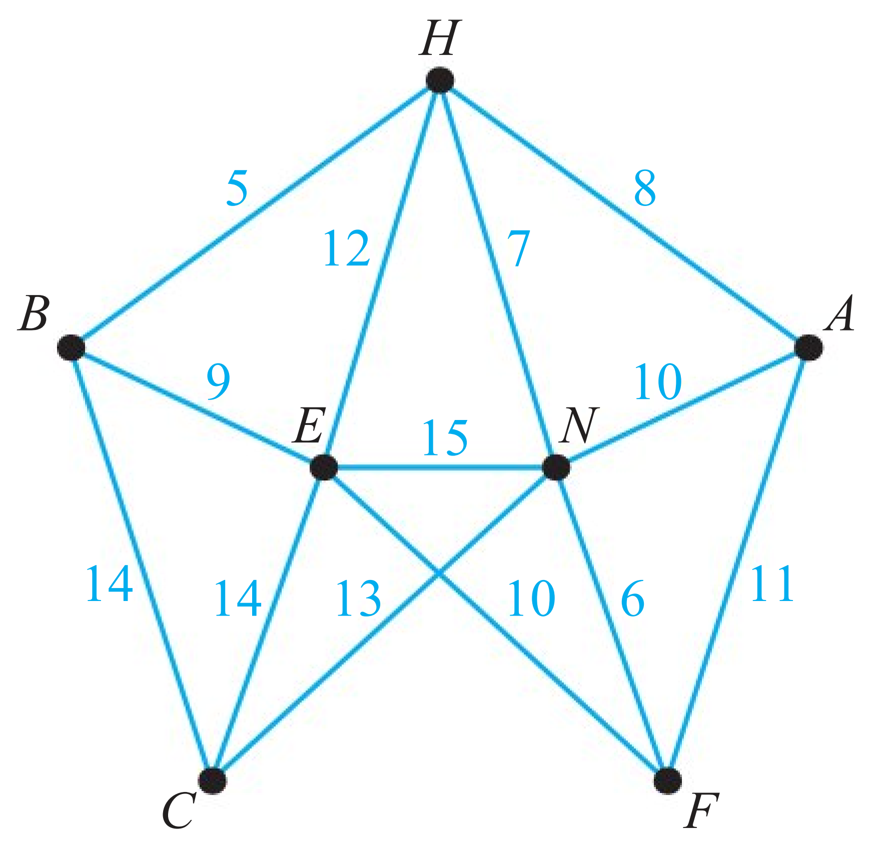


a Construct an adjacency matrix.
b Find the number of walks of length 5 from A to B.
c Find two vertices which are connected by seven different paths of length 4.

4 **a** Explain what is meant by a minimum spanning tree.

b Use Kruskal's algorithm to find the minimum spanning tree for the graph shown. List edges in the order you added them and state the weight of your tree.

c The graph represents a network of roads connecting seven mountain villages, with distances given in kilometres. After a snow storm, some of the roads need to be cleared so that each village can be reached from any other village. State which roads should be selected in order to minimize the total length of roads to clear?

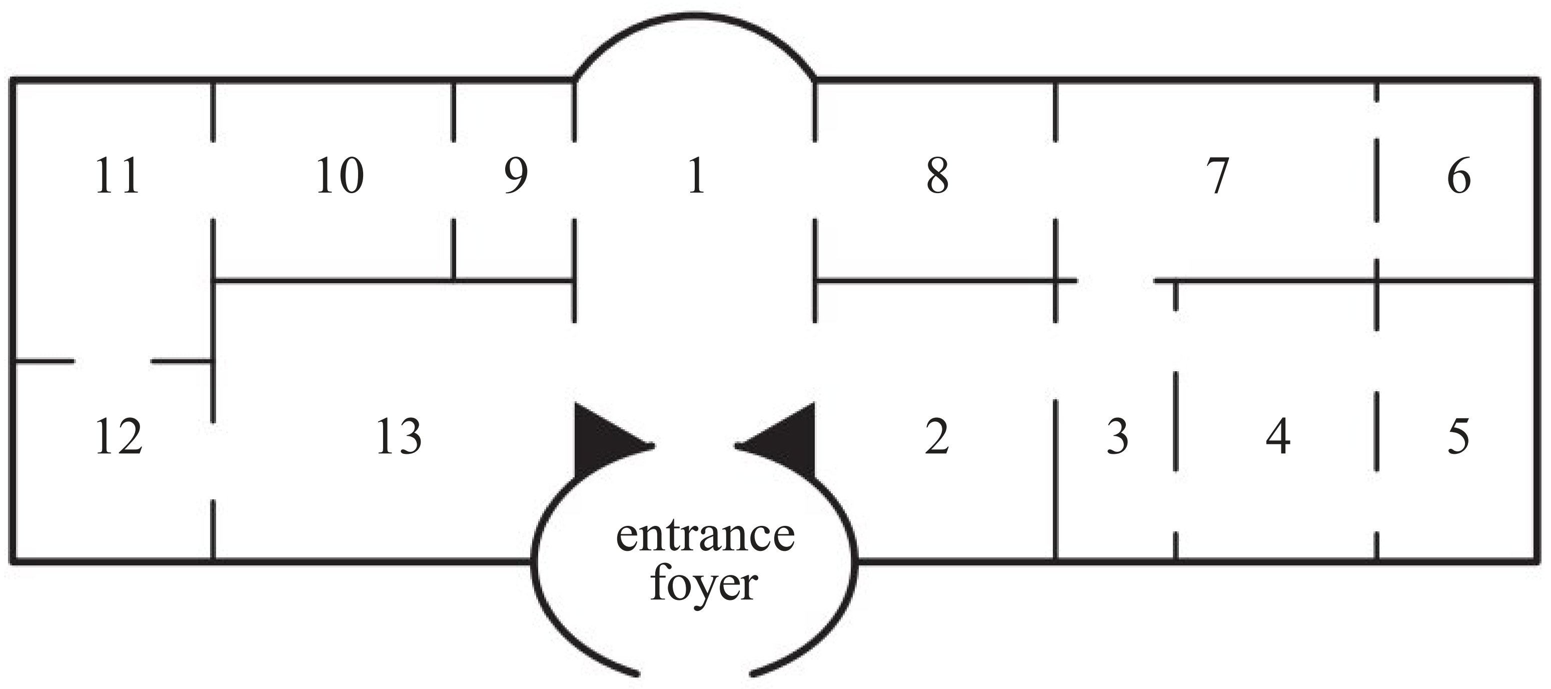


5 **a** Explain briefly the difference between Kruskal's and Prim's algorithms for finding the minimum spanning tree.

b Use Prim's algorithm, starting at *A*, to find the minimum spanning tree for the graph with the following weighted adjacency table. Draw the tree and state its weight.

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
A	–	14	–	–	–	–	–	–	15.5	15
B	14	–	8.5	–	–	–	–	–	12	–
C	–	8.5	–	22.5	–	–	–	18.5	13	–
D	–	–	22.5	–	21	10	25	–	–	–
E	–	–	–	21	–	8	16	–	–	–
F	–	–	–	10	8	–	20.5	–	–	–
G	–	–	–	25	16	20.5	–	19	–	–
H	–	–	18.5	–	–	–	19	–	18	–
I	15.5	12	13	–	–	–	–	18	–	11.5
J	15	–	–	–	–	–	–	–	11.5	–

6 The following figure shows the floor plan of a museum.



a **i** Draw a graph *G* that represents the plan of the museum where each exhibition room is represented by a vertex labelled with the exhibition room number and each door between exhibition rooms is represented by an edge. Do not consider the entrance foyer as a museum exhibition room.

ii Write down the degrees of the vertices that represent each exhibition room.

iii Virginia enters the museum through the entrance foyer. Use your answers to **i** and **ii** to justify why it is possible for her to visit the thirteen exhibition rooms going through each internal doorway exactly once.

b Let G and H be two graphs whose adjacency matrices are represented below.

G

	A	B	C	D	E	F
A	0	2	0	2	0	0
B	2	0	1	1	0	1
C	0	1	0	1	2	1
D	2	1	1	0	2	0
E	0	0	2	2	0	2
F	0	1	1	0	2	0

H

	A	B	C	D	E	F
A	0	1	3	0	1	2
B	1	0	1	3	2	0
C	3	1	0	2	1	3
D	0	3	2	0	2	0
E	1	2	1	2	0	1
F	2	0	3	0	1	0

Using the adjacency matrices,

i find the number of edges of each graph

ii show that exactly one of the graphs has a Eulerian trail

iii show that neither of the graphs has a Eulerian circuit.

Mathematics HL November 2013 Paper 3 Q2

7 The graph alongside shows the time required to walk between different classrooms in a school.
A teacher, located in classroom A, wants to take pens to each classroom and return to A.

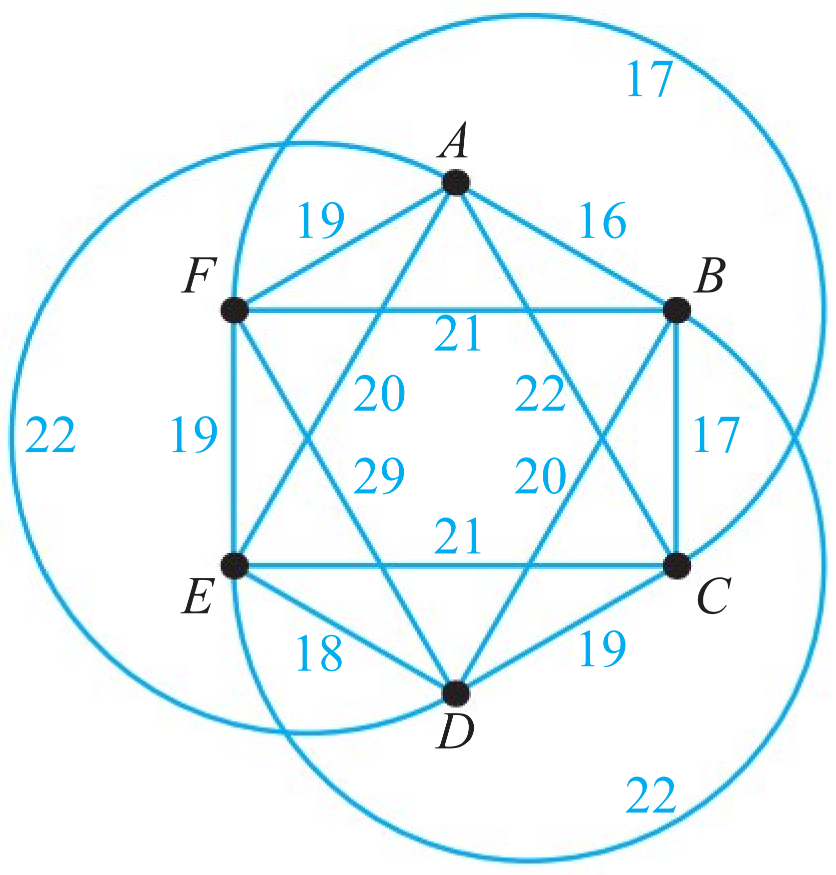
a Which of the following does the teacher need to find: an Eulerian circuit or a Hamiltonian cycle?

Let T seconds be the minimum possible time the teacher needs to take.

b Use the nearest neighbour algorithm, starting at vertex B, to find an upper bound for T.

c By removing vertex B find a lower bound.

d Write down an inequality satisfied by T.



8 **a** State the number of edges in a tree with n vertices.

b The table shows distances (in metres) between five workstations in an office.

	A	B	C	D	E
A	–	3	6	3	8
B	3	–	3	5	5
C	6	3	–	4	6
D	3	5	4	–	3
E	8	5	6	3	–

i Use Prim's algorithm, starting at A, to find the minimum spanning tree for the graph represented by this table.

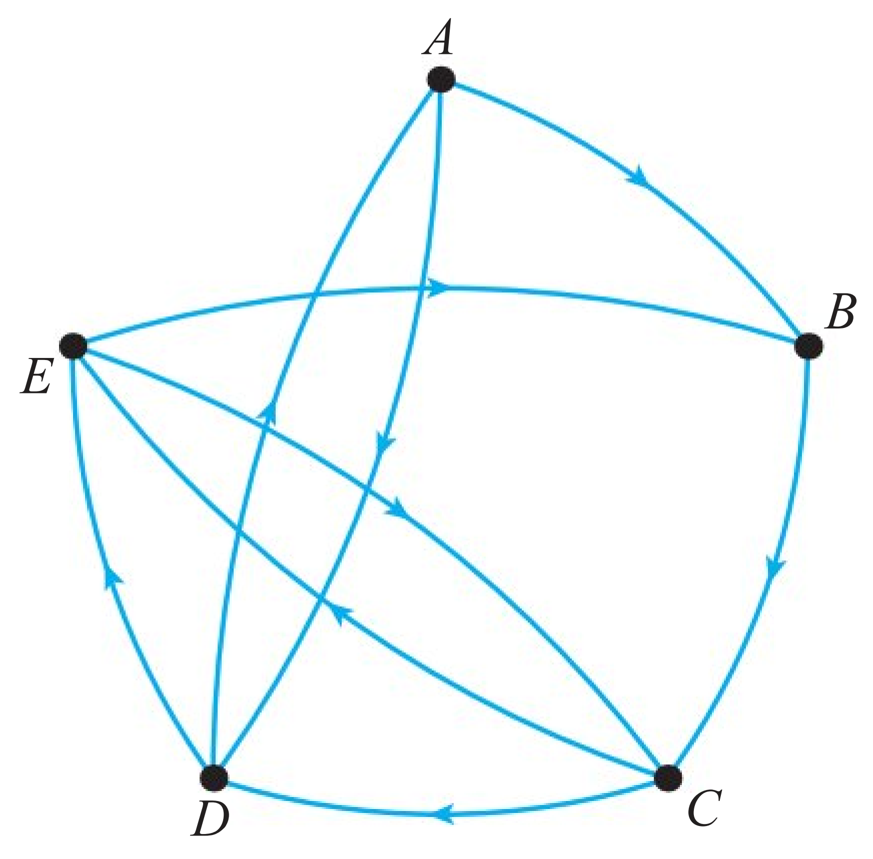
ii State the minimum length of cable required to connect all the workstations.

c **i** Explain how you can tell that the graph is Eulerian.

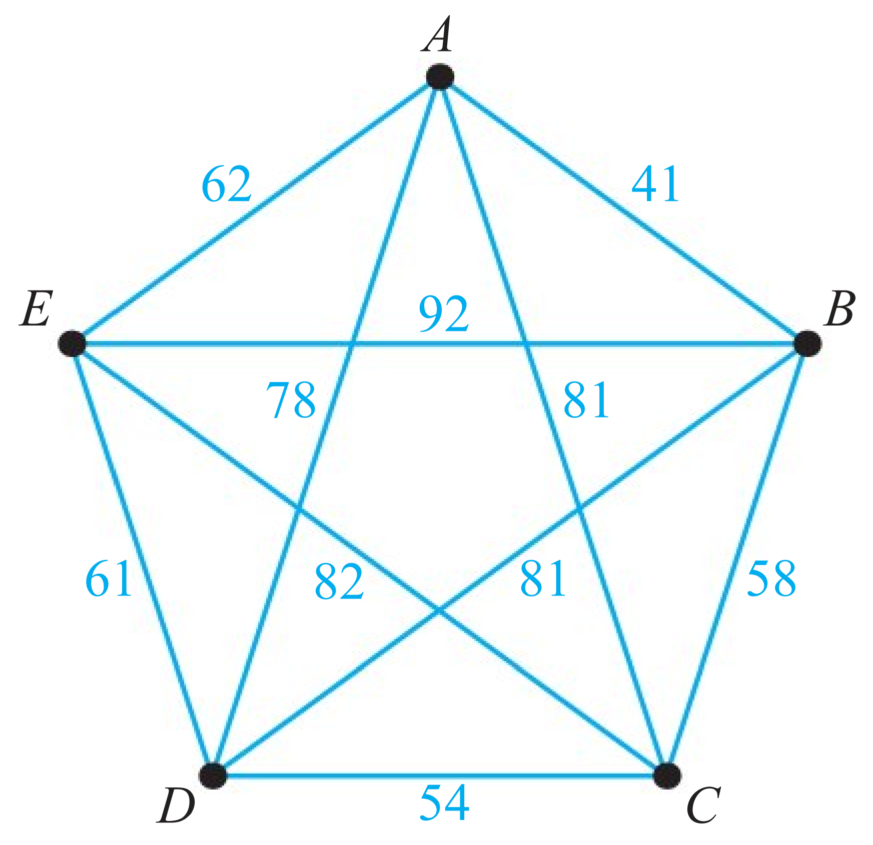
ii Find an Eulerian cycle starting at A and state its length.

9 The graph shows links between five web pages; for example, page A contains links to pages B and D.

Use the PageRank algorithm to rank the five pages in order of importance.



10 By deleting each vertex in turn, find a lower bound for the travelling salesman problem for the graph alongside.



11 The diagram shows a plan of a house, with doors connecting adjacent rooms.

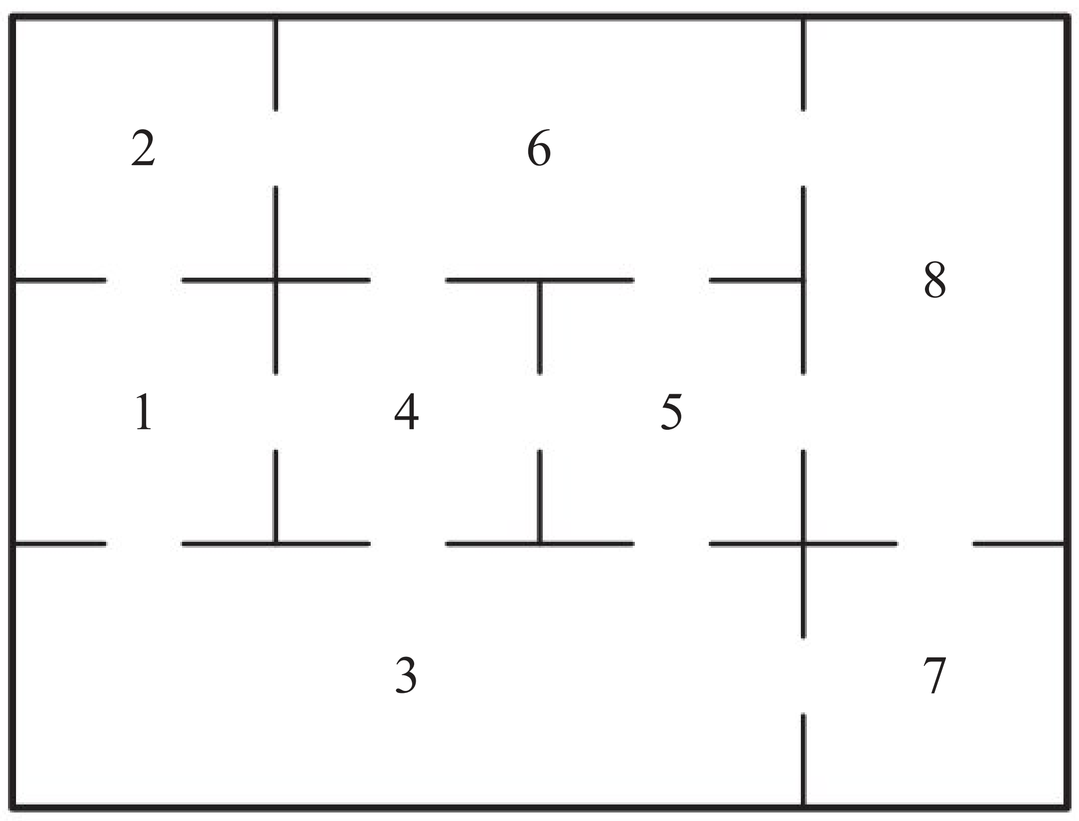
a Represent the plan on a graph, with vertices representing rooms and edges representing doors.

b Explain why it is not possible to walk through each door once and return to the starting room.

c Construct a transition matrix for the graph. A child starts in Room 1 and walks around randomly.

d Find the probability that the third room the child enters is Room 3.

e If the child continues to walk around the house for a long time, what percentage of time will they spend in Room 8?

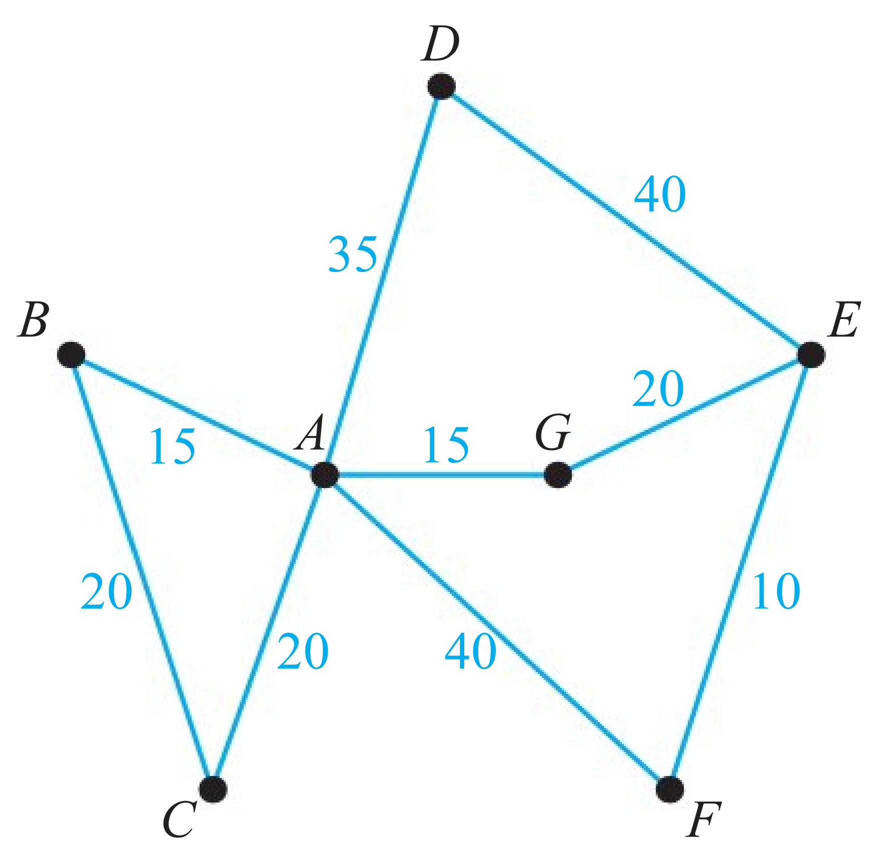


12 The graph below represents a network of paths connecting seven fountains in a park, their lengths given in metres.

a Explain why it is not possible to start from fountain A, walk along each path exactly once and return to A.

b Find the length of the shortest route which uses each path at least once and returns to A. Show your method clearly.

c A new path is to be built so that it becomes possible to walk along each path exactly once and return to the starting point. Between which two fountains should this new path be built?



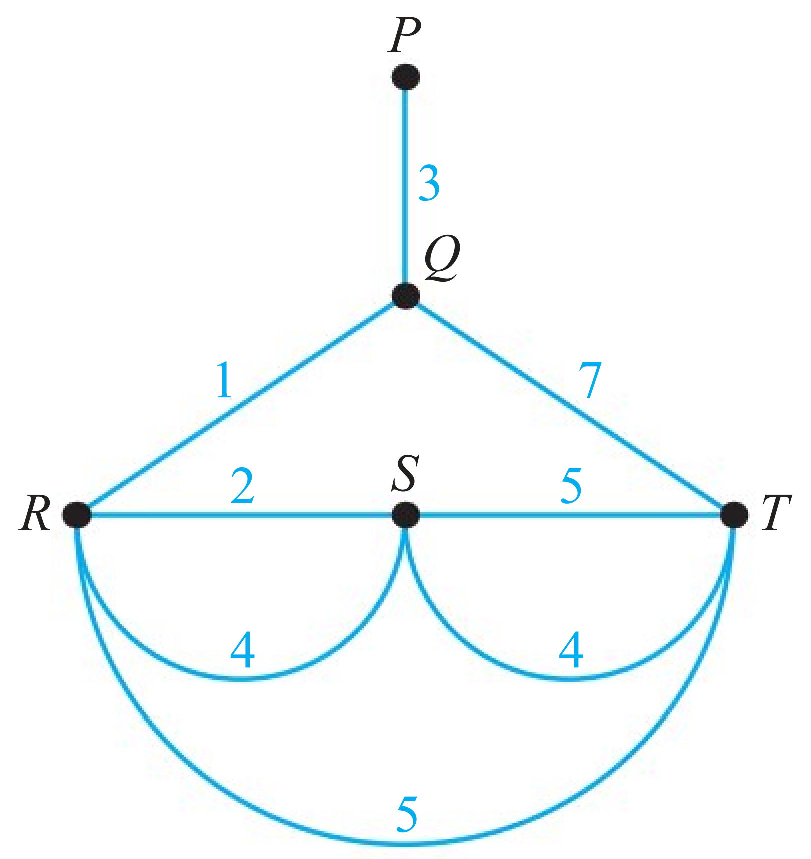
13 **a** Consider the following weighted graph.

i Write down a solution to the Chinese postman problem for this graph.

ii Calculate the total weight of the solution.

b **i** State the travelling salesman problem.

ii Explain why there is no solution to the travelling salesman problem for this graph.



Mathematics HL May 2015 Paper 3 Q1 parts b and c

14 The graph G has adjacency matrix $\mathbf{M}$ given below.

$$\begin{array}{c|cccccc} & A & B & C & D & E & F \\ \hline A & 0 & 1 & 0 & 0 & 0 & 1 \\ B & 1 & 0 & 1 & 0 & 1 & 0 \\ C & 0 & 1 & 0 & 1 & 0 & 0 \\ D & 0 & 0 & 1 & 0 & 1 & 0 \\ E & 0 & 1 & 0 & 1 & 0 & 1 \\ F & 1 & 0 & 0 & 0 & 1 & 0 \end{array}$$

a Draw the graph G.

b What information about G is contained in the diagonal elements of $\mathbf{M}^2$?

c Find the number of walks of length 4 starting at A and ending at C.

d List the trails of length 4 starting at A and ending at C.

Mathematics HL May 2012 Paper 3 Q3

15 The complete graph H has the following cost adjacency matrix.

	A	*B*	*C*	*D*	*E*
A	–	19	17	10	15
B	19	–	11	16	13
C	17	11	–	14	13
D	10	16	14	–	18
E	15	13	13	18	–

Consider the travelling salesman problem for H.

a By first finding a minimum spanning tree on the subgraph of H formed by deleting vertex A and all edges connected to A, find a lower bound for this problem.

b Find the total weight of the cycle $ADCBEA$.

c What do you conclude from your results?

Mathematics HL May 2011 Paper 3 Q2

16 The table shows the lengths (in km) of main roads between eight villages.

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*
A	–	7	9	–	9	–	–	–
B	7	–	10	12	–	12	8	–
C	9	10	–	–	11	–	–	–
D	–	12	–	–	10	8	8	5
E	9	–	11	10	–	12	–	7
F	–	12	–	8	12	–	11	6
G	–	8	–	8	–	11	–	8
H	–	–	–	5	7	6	8	–

a A road inspector needs to drive along each road to inspect it.

i Explain why he cannot return to the starting village without using some roads more than once.

ii Can he use each road exactly once if he does not have to return to the starting village? Justify your answer.

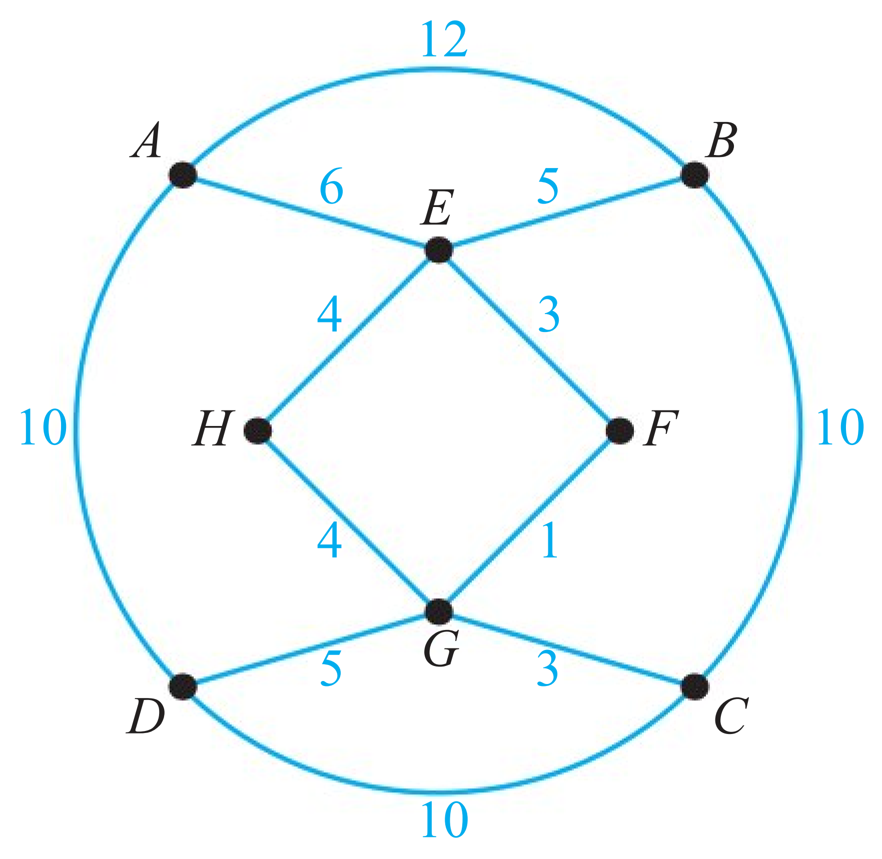
b A snowplough needs to clear all the roads. To do this, it must travel along each road twice (not necessarily in opposite directions).

i Explain why the snowplough can start from village A, travel along each road exactly twice, and return back to A.

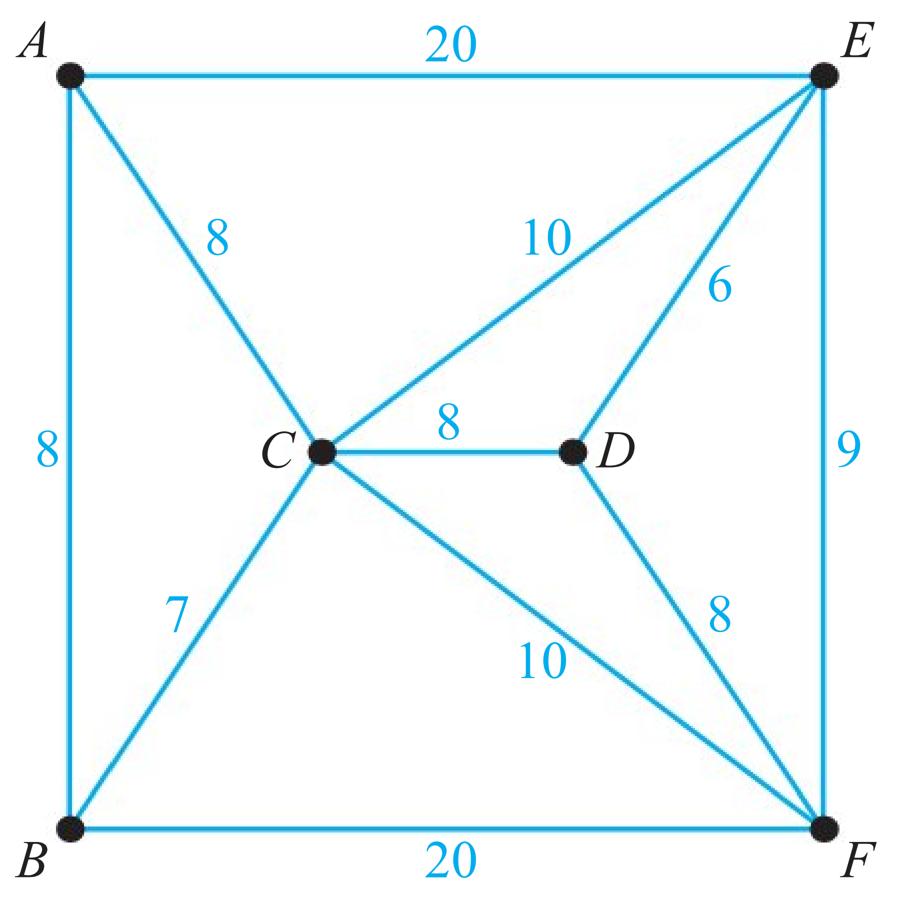
ii Find the distance the snowplough needs to cover.

17 The diagram shows a network of bus routes connecting cities A to H. The weight of the edges represent the cost, in \$, of travel on each route. A passenger wants to travel along each route at least once and return to the starting point.

Find the cheapest possible way to do this and state which routes need to be used more than once.



18 The graph shows boat routes between six islands and the time, in minutes, to sail each route.



A boat starts at port A and needs to deliver food to every island before returning to A.

a Construct a table showing the shortest travel times between each pair of islands.

The shortest time for the boat's tour is T minutes.

b By removing vertex B from the graph, find a lower bound for T.

c Use the nearest neighbour algorithm, starting at C, to find an upper bound for T.

d Write down an inequality satisfied by T.

e Find a tour of the islands which corresponds to your upper bound from part **c**.

19 Consider the graph with the following cost adjacency matrix.

	A	**B**	**C**	**D**	**E**	**F**
A	–	36	41	44	50	52
B	36	–	38	42	48	40
C	41	38	–	35	41	52
D	44	42	35	–	44	50
E	50	48	41	44	–	48
F	52	40	52	50	48	–

a Use Kruskal's algorithm to find the minimum spanning tree.

b For the travelling salesman problem on this graph:

i Remove vertex A to find a lower bound.

ii Remove vertex B to find another lower bound.

iii State which of the two lower bounds is better.

c By considering the cycle $ABCDEFA$, find an upper bound, and hence write an inequality satisfied by the solution of the travelling salesman problem.

20 Graph G is represented by the following weighted adjacency matrix.

$$\begin{pmatrix} - & 6 & 10 & 9 & 7 \\ 6 & - & 9 & 8 & 8 \\ 10 & 9 & - & 6 & 10 \\ 9 & 8 & 6 & - & 5 \\ 7 & 8 & 10 & 5 & - \end{pmatrix}$$

a By deleting each vertex in turn, find a lower bound for the travelling salesman problem for G.

b Explain why this lower bound is, in fact, the solution to the travelling salesman problem.