<table>
<thead>
<tr>
<th>CONTENTS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>COURSE STRUCTURE</td>
<td>iv</td>
</tr>
<tr>
<td>ABOUT THIS BOOK</td>
<td>vi</td>
</tr>
<tr>
<td>QUALIFICATION AND ASSESSMENT OVERVIEW</td>
<td>viii</td>
</tr>
<tr>
<td>EXTRA ONLINE CONTENT</td>
<td>x</td>
</tr>
<tr>
<td>1 ALGORITHMS</td>
<td>1</td>
</tr>
<tr>
<td>2 GRAPHS AND NETWORKS</td>
<td>29</td>
</tr>
<tr>
<td>3 ALGORITHMS ON GRAPHS</td>
<td>49</td>
</tr>
<tr>
<td>REVIEW EXERCISE 1</td>
<td>76</td>
</tr>
<tr>
<td>4 ROUTE INSPECTION</td>
<td>81</td>
</tr>
<tr>
<td>5 THE TRAVELLING SALESMAN PROBLEM</td>
<td>95</td>
</tr>
<tr>
<td>6 CRITICAL PATH ANALYSIS</td>
<td>121</td>
</tr>
<tr>
<td>7 LINEAR PROGRAMMING</td>
<td>154</td>
</tr>
<tr>
<td>REVIEW EXERCISE 2</td>
<td>187</td>
</tr>
<tr>
<td>EXAM PRACTICE</td>
<td>199</td>
</tr>
<tr>
<td>GLOSSARY</td>
<td>205</td>
</tr>
<tr>
<td>ANSWERS</td>
<td>208</td>
</tr>
<tr>
<td>INDEX</td>
<td>255</td>
</tr>
</tbody>
</table>
The following three themes have been fully integrated throughout the Pearson Edexcel International Advanced Level in Mathematics series, so they can be applied alongside your learning.

1. Mathematical argument, language and proof
   - Rigorous and consistent approach throughout
   - Notation boxes explain key mathematical language and symbols

2. Mathematical problem-solving
   - Hundreds of problem-solving questions, fully integrated into the main exercises
   - Problem-solving boxes provide tips and strategies
   - Challenge questions provide extra stretch

3. Transferable skills
   - Transferable skills are embedded throughout this book, in the exercises and in some examples
   - These skills are signposted to show students which skills they are using and developing

Finding your way around the book

Each chapter starts with a list of Learning objectives.
The Prior knowledge check helps make sure you are ready to start the chapter.
Glossary terms will be identified by bold blue text on their first appearance.

Each chapter is mapped to the specification content for easy reference.
The real world applications of the mathematics you are about to learn are highlighted at the start of the chapter.
About this book

Each section begins with explanation and key learning points.

Exercise questions are carefully graded so they increase in difficulty and gradually bring you up to exam standard.

Problem-solving boxes provide hints, tips and strategies, and Watch out boxes highlight areas where students often lose marks in their exams.

Exam-style questions are flagged with "E".

Transferable skills are signposted where they naturally occur in the exercises and examples.

Challenge boxes give you a chance to tackle some more difficult questions.

Step-by-step worked examples focus on the key types of questions you’ll need to tackle.

Exam practice

International Advanced Subsidiary/Advanced Level Decision Mathematics 1

Time: 1 hour 30 minutes

Intermediate: Exam-style questions,.Tables,Calculator

Answer ALL questions

Review exercise

1. An algorithm is described by the flow chart below.

   a. What is the value of the final output variable, X, after the algorithm terminates?
   b. Calculate the number of times the decision block is executed.
   c. Explain how the algorithm could be modified to ensure that X is always calculated correctly.

   a. The value of X after the algorithm terminates is 50.
   b. The decision block is executed 5 times.
   c. The algorithm can be modified by adding a check for the value of X before it is calculated, ensuring it is calculated correctly.

A full practice paper at the back of the book helps you prepare for the real thing.
QUALIFICATION AND ASSESSMENT OVERVIEW

Qualification and content overview

Decision Mathematics 1 (D1) is an optional unit in the following qualifications:
- International Advanced Subsidiary in Mathematics
- International Advanced Subsidiary in Further Mathematics
- International Advanced Level in Mathematics
- International Advanced Level in Further Mathematics

Assessment overview

The following table gives an overview of the assessment for this unit.

We recommend that you study this information closely to help ensure that you are fully prepared for this course and know exactly what to expect in the assessment.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Percentage</th>
<th>Mark</th>
<th>Time</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1: Decision Mathematics 1</td>
<td>33 ⅓ % of IAS</td>
<td>75</td>
<td>1 hour 30 mins</td>
<td>January and June</td>
</tr>
<tr>
<td>Paper code WDM11/01</td>
<td>16 ⅔ % of IAL</td>
<td></td>
<td></td>
<td>First assessment June 2019</td>
</tr>
</tbody>
</table>

IAS: International Advanced Subsidiary, IAL: International Advanced A Level.

Assessment objectives and weightings

| AO1 | Recall, select and use their knowledge of mathematical facts, concepts and techniques in a variety of contexts. | 30% |
| AO2 | Construct rigorous mathematical arguments and proofs through use of precise statements, logical deduction and inference and by the manipulation of mathematical expressions, including the construction of extended arguments for handling substantial problems presented in unstructured form. | 30% |
| AO3 | Recall, select and use their knowledge of standard mathematical models to represent situations in the real world; recognise and understand given representations involving standard models; present and interpret results from such models in terms of the original situation, including discussion of the assumptions made and refinement of such models. | 10% |
| AO4 | Comprehend translations of common realistic contexts into mathematics; use the results of calculations to make predictions, or comment on the context; and, where appropriate, read critically and comprehend longer mathematical arguments or examples of applications. | 5% |
| AO5 | Use contemporary calculator technology and other permitted resources (such as formulae booklets or statistical tables) accurately and efficiently; understand when not to use such technology, and its limitations. Give answers to appropriate accuracy. | 5% |
Relationship of assessment objectives to units

<table>
<thead>
<tr>
<th>D1</th>
<th>AO1</th>
<th>AO2</th>
<th>AO3</th>
<th>AO4</th>
<th>AO5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marks out of 75</td>
<td>20–25</td>
<td>20–25</td>
<td>15–20</td>
<td>5–10</td>
<td>5–10</td>
</tr>
<tr>
<td>%</td>
<td>$26\frac{2}{3}$–33(\frac{1}{3})</td>
<td>$26\frac{2}{3}$–33(\frac{1}{3})</td>
<td>$20–26\frac{2}{3}$</td>
<td>$6\frac{2}{3}$–13(\frac{1}{3})</td>
<td>$6\frac{2}{3}$–13(\frac{1}{3})</td>
</tr>
</tbody>
</table>

Calculators

Students may use a calculator in assessments for these qualifications. Centres are responsible for making sure that calculators used by their students meet the requirements given in the table below.

Students are expected to have available a calculator with at least the following keys: +, −, ×, ÷, π, $x^2$, $\sqrt{x}$, $\frac{1}{x}$, $x^n$, ln $x$, $e^x$, $x!$, sine, cosine and tangent and their inverses in degrees and decimals of a degree, and in radians; memory.

Prohibitions

Calculators with any of the following facilities are prohibited in all examinations:

- databanks
- retrieval of text or formulae
- built-in symbolic algebra manipulations
- symbolic differentiation and/or integration
- language translators
- communication with other machines or the internet
**Extra online content**

Whenever you see an *Online* box, it means that there is extra online content available to support you.

**SolutionBank**

SolutionBank provides a full worked solution for questions in the book. Download all the solutions as a PDF or quickly find the solution you need online.

**Use of technology**

Explore topics in more detail, visualise problems and consolidate your understanding. Use pre-made GeoGebra activities or Casio resources for a graphic calculator.

**GeoGebra**

GeoGebra-powered interactives

Interact with the mathematics you are learning using GeoGebra’s easy-to-use tools

**CASIO**

Graphic calculator interactives

Explore the mathematics you are learning and gain confidence in using a graphic calculator

**Calculator tutorials**

Our helpful video tutorials will guide you through how to use your calculator in the exams. They cover both Casio’s scientific and colour graphic calculators.

**Online**

Work out each coefficient quickly using the $^nC_r$ and power functions on your calculator.
1 ALGORITHMS

Learning objectives

After completing this chapter you should be able to:

- Use and understand an algorithm given in words → pages 2–5
- Understand how flow charts can be used to describe algorithms → pages 5–10
- Carry out a bubble sort → pages 10–13
- Carry out a quick sort → pages 13–16
- Carry out the three bin-packing algorithms and understand their strengths and weaknesses → pages 16–22
- Determine the order of an algorithm → pages 22–24

Prior knowledge check

1. Here is a function machine.
   Input → × 4 → + 3 → Output

   Find:
   a. the output when the input is 11
   b. the input when the output is 99.

2. Given that $x_1 = 5$, use the formula $x_{n+1} = \sqrt{x_n + 4}$ to find $x_2$, $x_3$ and $x_4$, giving your answers to 3 s.f.

Efficient sorting algorithms such as the quick sort allow software that governs self-drive cars to prioritise input information and react quickly and safely.
1.1 Using and understanding algorithms

- An **algorithm** is a **finite** sequence of step-by-step instructions carried out to solve a problem.

Algorithms can be given in words or in flow charts.

You need to be able to understand and use an algorithm given in words.

You have been using algorithms since you started school. Some examples of mathematical algorithms that you will be familiar with are:

- how to add several two-digit numbers
- how to multiply two two-digit numbers
- how to add, subtract, multiply or divide fractions.

It can be quite challenging to write a sequence of instructions for someone else to follow accurately.

Here are some more examples:

At the end of the street turn right and go straight over the crossroads, take the third left after the school, then ...

Affix base (B) to leg (A) using screw (F) and ...

Dice two large onions.
Slice 100 g mushrooms.
Grate 100 g cheese.

**Example 1**

A ‘happy’ number is defined by the algorithm:

- write down any integer
- square its digits and find the sum of the squares
- repeat until either the answer is 1 (in which case the number is ‘happy’) or until you get trapped in a cycle (in which case the number is ‘unhappy’)

Show that:

a 70 is happy  
b 4 is unhappy

\[
\begin{align*}
\text{a} & \quad 7^2 + 0^2 = 49 \\
& \quad 4^2 + 9^2 = 97 \\
& \quad 9^2 + 7^2 = 130 \\
& \quad 1^2 + 3^2 + 0^2 = 10 \\
& \quad 1^2 + 0^2 = 1 \\
& \quad \text{so } 70 \text{ is happy}
\end{align*}
\]

\[
\begin{align*}
\text{b} & \quad 4^2 = 16 \\
& \quad 1^2 + 6^2 = 37 \\
& \quad 3^2 + 7^2 = 58 \\
& \quad 5^2 + 8^2 = 89 \\
& \quad 8^2 + 9^2 = 145 \\
& \quad 1^2 + 4^2 + 5^2 = 42 \\
& \quad 4^2 + 2^2 = 20 \\
& \quad 2^2 + 0^2 = 4 \\
& \quad 4^2 = 16 \\
& \quad \text{so } 4 \text{ is unhappy}
\end{align*}
\]

**Watch out** You will need to be able to understand, describe and apply specific algorithms in your exam. You do not need to learn any of the algorithms in this section.

As soon as the sum of the squares matches a previous result, all of the steps in-between will be repeated, creating a cycle.
Example 2

a. Apply this algorithm.

1. Let \( n = 1 \), \( A = 1 \), \( B = 1 \).
2. Print \( A \) and \( B \).
3. Let \( C = A + B \).
4. Print \( C \).
5. Let \( n = n + 1 \), \( A = B \), \( B = C \).
6. If \( n < 5 \), go to step 3.
7. If \( n = 5 \), stop.

These are not equations. They are instructions that mean:
- replace \( n \) by \( n + 1 \) (add 1 to \( n \))
- \( A \) takes \( B \)'s current value
- \( B \) takes \( C \)'s current value

b. Describe the numbers that are generated by this algorithm.

A trace table is used to record the values of each variable as the algorithm is run.

You may be asked to complete a printed trace table in your exam. Just obey each instruction, in order.

You may be asked what the algorithm does.

<table>
<thead>
<tr>
<th>Step</th>
<th>( n )</th>
<th>( A )</th>
<th>( B )</th>
<th>( C )</th>
<th>Print</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>1, 1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Go to step 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Go to step 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Go to step 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Continue to step 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Stop</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 3

This algorithm multiplies the two numbers \( A \) and \( B \).

1. Make a table with two columns. Write \( A \) in the top row of the left-hand column and \( B \) in the top row of the right-hand column. In the next row, write the values for \( A \) and \( B \).
2. In successive rows, write:
   - in the left-hand column, the number that is half of \( A \), ignoring remainders
   - in the right-hand column, the number that is double \( B \)
3. Repeat step 2 until you reach the row which has a 1 in the left-hand column.
4. Delete all rows where the number in the left-hand column is even.
5. Find the sum of the non-deleted numbers in the right-hand column. This is the product \( AB \).

Apply this algorithm when:

a. \( A = 29 \) and \( B = 34 \)
   
   \[
   \begin{array}{c|c}
   \hline
   A & B \\
   \hline
   29 & 34 \\
   14 & 68 \\
   7 & 136 \\
   3 & 272 \\
   1 & 544 \\
   \text{Total} & 986 \\
   \hline
   \end{array}
   \]
   
   So \( 29 \times 34 = 986 \)

b. \( A = 66 \) and \( B = 56 \)

   \[
   \begin{array}{c|c}
   \hline
   A & B \\
   \hline
   66 & 56 \\
   33 & 112 \\
   16 & 224 \\
   8 & 448 \\
   4 & 896 \\
   2 & 1792 \\
   1 & 3584 \\
   \text{Total} & 3696 \\
   \hline
   \end{array}
   \]
   
   So \( 66 \times 56 = 3696 \)
Exercise 1A SKILLS INTERPRETATION

1 Use the algorithm in Example 3 to evaluate:
   a 244 × 125  
   b 125 × 244  
   c 256 × 123

2 The box below describes an algorithm.

   1 Write the input numbers in the form $\frac{a}{b}$ and $\frac{c}{d}$.
   2 Let $e = ad$.
   3 Let $f = bc$.
   4 Print $\frac{e}{f}$.

   a Apply this algorithm with the input numbers $2\frac{1}{4}$ and $1\frac{1}{3}$.
   b What does this algorithm do?

3 The box below describes an algorithm.

   1 Let $A = 1$, $n = 1$.
   2 Print $A$.
   3 Let $A = A + 2n + 1$.
   4 Let $n = n + 1$.
   5 If $n \leq 10$, go to 2.
   6 Stop.

   a Apply the algorithm.
   b Describe the numbers produced by the algorithm.

4 The box below describes an algorithm.

   1 Input $A$, $r$.
   2 Let $C = \frac{A}{r}$ to 3 d.p.
   3 If $|r - C| \leq 10^{-2}$ go to 7.
   4 Let $s = \frac{1}{2}(r + C)$ to 3 d.p.
   5 Let $r = s$.
   6 Go to 2.
   7 Print $r$.
   8 Stop.

   a Use a trace table to apply the algorithm above when:
      i $A = 253$ and $r = 12$   ii $A = 79$ and $r = 10$   iii $A = 4275$ and $r = 50$
   b What does the algorithm produce?

1.2 Flow charts

You need to be able to apply an algorithm given as a flow chart.

- In a flow chart, the shape of each box tells you about its function.

The boxes in a flow chart are linked by arrowed lines. As with an algorithm written in words, you need to follow each step in order.
The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \)?

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.

**Example 4**

The flow chart below describes an algorithm.

Box 1  
Start

Box 2  
Let \( n = 0 \)

Box 3  
Let \( n = n + 1 \)

Box 4  
Let \( E = 2n \)

Box 5  
Print \( E \)

Box 6  
Is \( n \geq 10 \) Yes

Box 7  
Stop

\( n \) is acting as a counter. It ensures that the algorithm stops after this loop has been completed 10 times.

A decision box will contain a question to which the answer is either ‘yes’ or ‘no’.
Example 5

This flow chart can be used to find the roots of an equation of the form $ax^2 + bx + c = 0$.

```
Start

Input $a$, $b$, $c$

Let $d = b^2 - 4ac$

Is $d < 0$?

Yes Print 'no real roots'

No Is $d = 0$?

Yes Let $x = -\frac{b}{2a}$ Print 'equal roots are' $x$

No

Let $x_1 = \frac{-b + \sqrt{d}}{2a}$

Let $x_2 = \frac{-b - \sqrt{d}}{2a}$

Print 'roots are' $x_1$ 'and' $x_2$

Stop
```

Demonstrate this algorithm for these equations:

a) $6x^2 - 5x - 11 = 0$

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
<th>$d &lt; 0$?</th>
<th>$d = 0$?</th>
<th>$x_1$</th>
<th>$x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>6</td>
<td>-5</td>
<td>-11</td>
<td>289</td>
<td>No</td>
<td>No</td>
<td>11/6</td>
<td>-1</td>
</tr>
</tbody>
</table>

roots are $\frac{11}{6}$ and $-1$

b) $x^2 - 6x + 9 = 0$

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
<th>$d &lt; 0$?</th>
<th>$d = 0$?</th>
<th>$x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>1</td>
<td>-6</td>
<td>9</td>
<td>0</td>
<td>No</td>
<td>Yes</td>
<td>3</td>
</tr>
</tbody>
</table>

equal roots are 3

c) $4x^2 + 3x + 8 = 0$

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
<th>$d &lt; 0$?</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td>-119</td>
<td>Yes</td>
</tr>
</tbody>
</table>

no real roots
Example 6

Apply the algorithm shown by the flow chart on the right to the data:

\( u_1 = 10, \ u_2 = 15, \ u_3 = 9, \ u_4 = 7, \ u_5 = 11. \)

What does the algorithm do?

<table>
<thead>
<tr>
<th>Box 1</th>
<th>Box 2</th>
<th>Box 3</th>
<th>Box 4</th>
<th>Box 5</th>
<th>Box 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Output is 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The algorithm selects the smallest number from a list.

Exercise 1B

1. Apply the flow chart in Example 5 to the following equations.
   
   a \( 4x^2 - 12x + 9 = 0 \)  
   b \( -6x^2 + 13x + 5 = 0 \)  
   c \( 3x^2 - 8x + 11 = 0 \)

2. a. Apply the flow chart in Example 6 to the following sets of data.
   
   i. \( u_1 = 28, \ u_2 = 26, \ u_3 = 23, \ u_4 = 25, \ u_5 = 21 \)
   
   ii. \( u_1 = 11, \ u_2 = 8, \ u_3 = 9, \ u_4 = 8, \ u_5 = 5 \)

   b. If box 4 is altered to \( T > A \), how will this affect the output?

   c. Which box would need to be altered if the algorithm was to be applied to a list of 8 numbers?
3 The flow chart describes an algorithm that can be used to find the roots of the equation $2x^3 + x^2 - 15 = 0$.
   a Use $a = 2$ to find a root of the equation.
   b Use $a = 20$ to find a root of the equation.
   Comment on your answer.

4 The flow chart on the right describes how to apply Euclid’s algorithm to two non-zero integers, $a$ and $b$.
   a Apply Euclid’s algorithm to:
      i $507$ and $52$ (2 marks)
      ii $884$ and $85$ (2 marks)
      iii $4845$ and $3795$ (2 marks)
   b Explain what Euclid’s algorithm does. (2 marks)

5 The flow chart describes an algorithm.
   a Copy and complete this table, using the flow chart with $A = 18$ and $B = 7$.

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>$A &lt; B$?</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
   (4 marks)

   b Explain what is achieved by this flow chart. (2 marks)

   c Given that $A = kB$ for some positive integer $k$, write down the output of the flow chart. (1 mark)
1.3 Bubble sort

A common data processing task is to sort an unordered list (a list which is not in order) into alphabetical or numerical order.

Lists can be put into ascending (increasing) or descending (decreasing) order.
- Unordered lists can be sorted using a bubble sort or a quick sort.
- In a bubble sort, we work through the list by comparing pairs of adjacent items (items that are next to each other) in the list.
  - If the items are in the correct order, leave them
  - If the items are not in the correct order, swap them

Once we have done this to all of the items in the list, we have completed the first pass.

If sorting the list into ascending order, the first pass will place the largest item in its correct position in the list.

If sorting the list into descending order, the first pass will place the smallest item in its correct position in the list.

We then repeat this until no swaps are made in a pass. If no swaps are made then the list is in order. You will need to write that no swaps have been made.

As the bubble sort develops, it is helpful to consider the original list as being divided into a working list, where comparisons are made, and a sorted list containing the items that are in their final positions. To start with, all items are in the working list.

This is the bubble sort algorithm:

1. Start at the beginning of the working list and move from left to right, comparing adjacent items.
   a. If they are in order, leave them.
   b. If they are not in order, swap them.
2. When you get to the end of the working list, the last item will be in its final position. This item is then no longer in the working list.
3. If you have made some swaps in the last pass, repeat step 1.
4. When a pass is completed without any swaps, every item is in its final position and the list is in order.

You need to learn the bubble sort algorithm.
Example 7  SKILLS  ANALYSIS

Use a bubble sort algorithm to arrange the list below into ascending order.

24  18  37  11  15  30

<table>
<thead>
<tr>
<th>24 18 37 11 15 30</th>
<th>1st comparison: swap</th>
</tr>
</thead>
<tbody>
<tr>
<td>18 24 37 11 15 30</td>
<td>2nd comparison: leave</td>
</tr>
<tr>
<td>18 24 37 11 15 30</td>
<td>3rd comparison: swap</td>
</tr>
<tr>
<td>18 24 11 37 15 30</td>
<td>4th comparison: swap</td>
</tr>
<tr>
<td>18 24 11 15 37 30</td>
<td>5th comparison: swap</td>
</tr>
<tr>
<td>18 24 11 15 30 37</td>
<td>End of first pass</td>
</tr>
</tbody>
</table>

After the second pass the list becomes
18  11  15  24  30  37

After the third pass the list is
11  15  18  24  30  37

After the fourth pass the list is
11  15  18  24  30  37

No swaps were made in the fourth pass, so the list is in order.

Example 8  SKILLS  REASONING/ARGUMENTATION

A list of \( n \) letters is to be sorted into alphabetical order, starting at the left-hand end of the list.

a Describe how to carry out the first pass of a bubble sort on the letters in the list.

b Carry out the first pass of a bubble sort to arrange the letters in the word ALGORITHM into alphabetical order, showing every step of the working.

c Show the order of the letters at the end of the second pass.

a Starting at the beginning of the list, compare the first two letters. If they are in alphabetical order, leave them in position, otherwise swap them. Continue through the list, to the end, comparing every pair of letters in the same way.

b A L G O R I T H M 1st comparison: leave
A L G O R I T H M 2nd comparison: swap
A G L O R I T H M 3rd comparison: leave
A G L O R I T H M 4th comparison: leave
A G L O I R H T M 5th comparison: swap
A G L O I R H T M 6th comparison: leave
A G L O I R H T M 7th comparison: swap
A G L O I R H T M 8th comparison: swap
A G L I O H M R T

At the end of the first pass, the last letter is guaranteed to be in its correct place.
Example 9

Use a bubble sort to arrange these numbers into descending order.

39 57 72 39 17 24 48

39 57 72 39 17 24 48
39 < 57 so swap
57 39 72 39 17 24 48
39 < 72 so swap
57 72 39 39 17 24 48
39 < 39 so leave
57 72 39 39 17 24 48
17 < 24 so swap
57 72 39 39 17 48 24
17 < 48 so swap
57 72 39 39 24 17 48

After 1st pass: 57 72 39 39 24 48 17
After 2nd pass: 72 57 39 39 48 24 17
After 3rd pass: 72 57 39 48 39 24 17
After 4th pass: 72 57 48 39 39 24 17
After 5th pass: 72 57 48 39 39 24 17

No swaps in 5th pass, so the list is in order.

Watch out Read the question carefully. You need to sort the list into descending order.

Note that the 48 is now between the two 39s.
Do not treat the two 39s as one term.

Make sure that you make a statement like this to show that no swaps have been made and you have completed the algorithm.

Exercise 1C

1 Apply a bubble sort to arrange each list below into:
   a ascending order       b descending order
   i 23 16 15 34 18 25 11 19
   ii N E T W O R K
   iii A5 D3 D2 A1 B4 C7 C2 B3

For each part, you need to show the state of the list only at the end of each pass.

Hint For part iii, order alphabetically then numerically. So C2 comes after A5 but before C7.

2 Perform a bubble sort to arrange these place names into alphabetical order.

Chester York Stafford Bridlington Burton Cranleigh Evesham

3 A list of $n$ items is to be written in ascending order using a bubble sort.
   a State the minimum number of passes needed.
   b Describe the circumstances in which this number of passes would be sufficient.
   c State the maximum number of passes needed.
   d Describe the circumstances in which this number of passes would be needed.
4 Here is a list of exam scores:

   63  48  57  55  32  48  72  49  61  39

The scores are to be put in order, highest first, using a bubble sort.

a Describe how to carry out the first pass.  
(2 marks)

b Apply a bubble sort to put the scores in the required order.  
Only show the state of the list at the end of each pass.  
(4 marks)

1.4 Quick sort

The quick sort algorithm can be used to arrange a list into alphabetical or numerical order. In many cases, a quick sort is faster to perform than a bubble sort. We can thus say that it is more efficient.

In a quick sort, we choose an item which we call a pivot, and split the items into two sublists:

- One sublist contains items less than the pivot.
- The other sublist contains items greater than the pivot.

Once we have done this we have completed the first pass.

In doing the quick sort, the first pass will place the pivot item in its correct position in the list.

We then repeat this until all items are chosen as pivots, and then the list is in order. You will need to write that all items are chosen as pivots, which means that they are in order.

Here is the quick sort algorithm, used to sort a list into ascending order.

1 Choose the item at the midpoint of the list to be the first pivot.
2 Write down all the items that are less than the pivot, keeping their order, in a sublist.
3 Write down the pivot.
4 Write down the remaining items (those greater than the pivot) in a sublist.
5 Apply steps 1 to 4 to each sublist.
6 When all items have been chosen as pivots, stop.

The number of pivots could double at each pass. There is 1 pivot at the first pass, there could be 2 at the second, 4 at the third, 8 at the fourth, and so on.
Example 10

SKILLS ANALYSIS

Use the quick sort algorithm to arrange the numbers below into ascending order.

21 24 42 29 23 13 8 39 38

For \( n \) items, the pivot will be the \( \frac{n+1}{2} \)th item, rounding up if necessary.

There are 9 numbers in the list so the midpoint will be \( \frac{9+1}{2} = 5 \), so the pivot is the 5th number in the list. Circle it.

Write all the numbers less than 23.

Write the pivot in a box, then write the remaining numbers.

Now select a pivot in each sublist.

There are now four sublists so we choose four pivots (circled).

We can choose only two pivots this time. Each number has been chosen as a pivot, so the list is in order.

Example 11

SKILLS PROBLEM-SOLVING

Use the quick sort algorithm to arrange the list below into descending order.

37 20 17 26 44 41 27 28 50 17

There are 10 items in the list so the midpoint will be \( \frac{10+1}{2} = 5.5 \), and so the pivot is the 6th number in the list. Circle it.

Numbers greater than the pivot are to the left of the pivot, those smaller than the pivot are to the right, keeping the numbers in order. Numbers equal to the pivot may go either side, but must be dealt with in the same way each time you do a pass.

Two pivots are chosen, one for each sublist.

Now three pivots are selected.

We now choose the next two pivots, even if the sublist is in order.

The final pivots are chosen to give the list in order.

Watch out: Colour is used here to make the method clear, but colours should not be used in your exam.
1. Use a bubble sort to arrange the list of numbers below into:
   a. ascending order
   b. descending order
   8 3 4 6 5 7 2

2. Use the quick sort algorithm to arrange the list below into:
   a. ascending order
   b. descending order
   22 17 25 30 11 18 20 14 7 29

3. Sort the letters below into alphabetical order using:
   a. a bubble sort
   b. a quick sort
   N H R K S C J E M P L

4. The list shows the test results of a group of students.
   Alex 33    Hetal 9
   Alison 56   Janelle 89
   Amy 93     Josh 37
   Annie 51   Lucy 57
   Dewei 77   Masingur 19
   Greg 91    Sam 29
   Harry 49   Sophie 77

   Produce a list of students, in descending order of their marks, using:
   a. a bubble sort
   b. a quick sort

5. A list of \( n \) items is to be written in ascending order using the bubble sort algorithm.
   a. Find an expression, in terms of \( n \), for the maximum number of comparisons to be made.  
      (2 marks)
   b. Describe a situation where a bubble sort might be quicker than a quick sort.  
      (2 marks)
   c. Decide whether a bubble sort or a quick sort will be quicker in the following cases:
      i. 1 2 3 7 4 5 6
      ii. 2 3 4 5 6 7 1

      Explain how you made your decisions.  
      (4 marks)
6 The table shows a list of nine names of students in a dance class.

<table>
<thead>
<tr>
<th>Hassler</th>
<th>Sauver</th>
<th>Finch</th>
<th>Giannini</th>
<th>Mellor</th>
<th>Clopton</th>
<th>Miranti</th>
<th>Worth</th>
<th>Argi</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>S</td>
<td>F</td>
<td>G</td>
<td>Me</td>
<td>C</td>
<td>Mi</td>
<td>W</td>
<td>A</td>
</tr>
</tbody>
</table>

(a) Explain how to carry out the first pass of a quick sort algorithm to order the list alphabetically. 

(b) Carry out the first two passes of a quick sort on this list, writing down the pivots used in each pass.

(2 marks) 

(3 marks)

Challenge

You will need a pack of ordinary playing cards, with any jokers removed. A pack of playing cards has 52 cards, split into 4 suits: 

- Hearts ♥
- Diamonds ♦
- Clubs♣
- Spades♠

There are 13 cards in each suit, as follows: 

- Ace (1), 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack (11), Queen (12), King (13)

(a) Use the quick sort algorithm to sort the cards into ascending order, from Ace to King within each suit and with the suits in the order: Hearts, Clubs, Diamonds, Spades. Follow these steps:

1. Shuffle the pack thoroughly and hold it face up.
2. Remove the 27th card and place it face up. This is your pivot card.
3. Go through the pack from the top. Place the cards into two piles depending on whether they are lower or higher than the pivot card.
4. Repeat these steps with each new pile, choosing the card halfway through the pile as the pivot card.

Record the total number of passes needed to sort the deck completely.

(b) Once the cards are in order, what single change could be made so that a bubble sort would require 51 passes to put the cards back in order?

Hint: The final order should be: 

A♥, 2♥, ..., K♥, A♣, 2♣, ..., K♣, A♦, ..., K♦, A♠, ..., K♠

1.5 Bin-packing algorithms

Bin packing refers to a whole class of problems.

The easiest way of thinking about this algorithm is to imagine boxes of fixed width \(a\) and length \(b\), but varying heights, and stacking them into bins of width \(a\) and length \(b\), using the minimum number of bins.
Similar problems are: loading cars of different lengths onto a ferry with several lanes of equal length, a plumber needing to cut sections from lengths of copper pipe, or recording music tracks onto a set of CDs.

You need to be able to apply three different bin-packing algorithms, and be aware of their strengths and weaknesses.

- The three bin-packing algorithms are: first-fit, first-fit decreasing and full-bin.

It is useful to first find a lower bound for the number of bins needed. There is no guarantee that you will be able to pack the items into this number of bins, but it will tell you if you have found an optimal solution.

**Example 12**

Nine boxes of fixed cross-section have heights, in metres, as follows.

\[
0.3 \quad 0.7 \quad 0.8 \quad 0.8 \quad 1.0 \quad 1.1 \quad 1.1 \quad 1.2 \quad 1.5
\]

They are to be packed into bins with the same fixed cross-section and height 2 m. Determine the lower bound for the number of bins needed.

Sum the heights and divide by the bin size. You must always round up to determine the lower bound.

\[
\frac{0.3 + 0.7 + 0.8 + 0.8 + 1.0 + 1.1 + 1.1 + 1.2 + 1.5}{2} = \frac{8.5}{2} = 4.25 \text{ bins}
\]

So a minimum of 5 bins will be needed.

With small amounts of data it is often possible to ‘spot’ an optimal answer.

The algorithms you will learn in this chapter will not necessarily find an optimal solution, but can be applied quickly.

- The first-fit algorithm works by considering items in the order they are given.

**First-fit algorithm**

1. Take the items in the order given.
2. Place each item in the first available bin that can take it. Start from bin 1 each time.

Advantage: It is quick to apply.
Disadvantage: It is not likely to lead to a good solution.
Use the first-fit algorithm to pack the following items into bins of size 20. (The numbers in brackets are the size of the item.) State the number of bins used and the amount of wasted space.

\[
\begin{align*}
A(8) & \\
B(7) & \\
C(14) & \\
D(9) & \\
E(6) & \\
F(9) & \\
G(5) & \\
H(15) & \\
I(6) & \\
J(7) & \\
K(8) & \\
\end{align*}
\]

This used 6 bins and there are \[2 + 5 + 7 + 12 = 26\] units of waste of space.

- The first-fit decreasing algorithm requires the items to be in descending order before applying the algorithm.

**First-fit decreasing algorithm**

1. Sort the items so that they are in descending order.
2. Apply the first-fit algorithm to the reordered list.

Advantages: You usually get a fairly good solution.
- It is easy to apply.

Disadvantage: You may not get an optimal solution.

Apply the first-fit decreasing algorithm to the data given in Example 13.

Sort the data into descending order:

\[
\begin{align*}
H(15) & \\
C(14) & \\
D(9) & \\
F(9) & \\
A(8) & \\
K(8) & \\
B(7) & \\
J(7) & \\
E(6) & \\
I(6) & \\
G(5) & \\
\end{align*}
\]

This used 5 bins and there are \[2 + 4 = 6\] units of wasted space.
Full-bin packing uses **inspection** to select items that will combine to fill bins. Remaining items are packed using the first-fit algorithm.

**Full-bin packing**

1. Use observation to find combinations of items that will fill a bin. Pack these items first.
2. Any remaining items are packed using the first-fit algorithm.

**Advantage:** You usually get a good solution.

**Disadvantage:** It is difficult to do, especially when the numbers are plentiful and awkward.

---

**Example 15**


The items above are to be packed in bins of size 25.

- **a** Determine the lower bound for the number of bins.
- **b** Apply the full-bin algorithm.
- **c** Is your solution optimal? Give a reason for your answer.

**a**

\[
\frac{111}{25} = 4.44
\]

So lower bound is 5 bins.

**b**

Three groupings of numbers that sum to 25 can be made as follows:

\[
\begin{align*}
8 + 7 + 10 &= 25 \\
11 + 14 &= 25 \\
13 + 12 &= 25
\end{align*}
\]

The first three bins are full bins.

**c**

The lower bound is 5, and 5 bins were used, so the solution is optimal.

We now apply the first-fit algorithm to the remainder.

- F(17) goes into bin 4, leaving space of 8.
- G(4) goes into bin 4, leaving space of 4.
- H(6) goes into bin 5, leaving space of 19.
- K(9) goes into bin 5, leaving space of 10.
Example 16  

A plumber needs to cut the following lengths of copper pipe. (Lengths are in metres.)

A(0.8)  B(0.8)  C(1.4)  D(1.1)  E(1.3)  F(0.9)  G(0.8)  H(0.9)  I(0.8)  J(0.9)

The pipe comes in lengths of 2.5 m.

a Calculate the lower bound of the number of lengths of pipe needed.

b Use the first-fit decreasing algorithm to determine how the required lengths may be cut from the 2.5 m lengths.

c Use full-bin packing to find an optimal solution.

---

### a

\[
0.8 + 0.8 + 1.4 + 1.1 + 1.3 + 0.9 + 0.8 + 0.8 + 0.9
= 9.68
\]

So at least 4 lengths are required.

### b

Sorting into descending order,

C(1.4), E(1.3), D(1.1), F(0.9), H(0.9), J(0.9), A(0.8), B(0.8), G(0.8), I(0.8)

<table>
<thead>
<tr>
<th>Bin 1</th>
<th>Bin 2</th>
<th>Bin 3</th>
<th>Bin 4</th>
<th>Bin 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>D(1.1)</td>
<td>F(0.9)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C(1.4)</td>
<td>E(1.3)</td>
<td>H(0.9)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>J(0.9)</td>
<td>G(0.8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A(0.8)</td>
<td>I(0.8)</td>
</tr>
</tbody>
</table>

- C goes into bin 1, leaving space of 1.1.
- E goes into bin 2, leaving space of 1.2.
- D goes into bin 1, leaving space of 0.
- F goes into bin 2, leaving space of 0.3.
- H goes into bin 3, leaving space of 1.6.
- J goes into bin 3, leaving space of 0.7.
- A goes into bin 4, leaving space of 1.7.
- B goes into bin 4, leaving space of 0.9.
- G goes into bin 4, leaving space of 0.1.
- I goes into bin 5, leaving space of 1.7.

Since a sort was not asked for, this can be done by inspection.

### c

By inspection,

\[
C(1.4) + D(1.1) = 2.5 \\
F(0.9) + A(0.8) + B(0.8) = 2.5 \\
J(0.9) + G(0.8) + I(0.8) = 2.5
\]

A full-bin solution is:

<table>
<thead>
<tr>
<th>Bin 1</th>
<th>Bin 2</th>
<th>Bin 3</th>
<th>Bin 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>D(1.1)</td>
<td>B(0.8)</td>
<td>I(0.8)</td>
<td>H(0.9)</td>
</tr>
<tr>
<td>C(1.4)</td>
<td>A(0.8)</td>
<td>G(0.8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F(0.9)</td>
<td>J(0.9)</td>
<td>E(1.3)</td>
</tr>
</tbody>
</table>

In part a we found that at least 4 lengths would be needed, so this solution is optimal since it uses 4 lengths.
1. The above items are to be packed in bins of size 50.
   a. Calculate the lower bound for the number of bins.
   b. Pack the items into the bins using:
      i. the first-fit algorithm
      ii. the first-fit decreasing algorithm
      iii. the full-bin algorithm

2. Laura hosts an internet music channel and wishes to play the 13 pieces of music listed below. Each day, she hosts a session which is at most 3 hours long.
<table>
<thead>
<tr>
<th>Piece of music</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (minutes)</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>45</td>
<td>45</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>75</td>
<td>90</td>
<td>120</td>
<td>120</td>
</tr>
</tbody>
</table>
   a. Apply the first-fit algorithm, in the order A to M, to determine the number of days that need to be used. State which music is played on each day.
   b. Repeat part a using the first-fit decreasing algorithm.
   c. Is your answer to part b optimal? Give a reason for your answer.
   Laura finds that her session time is now reduced to only 2 hours.
   d. Use the full-bin algorithm to determine the number of days she needs to use. State which music is played on each day.

3. A small ferry loads vehicles into 30 m lanes. The vehicles are loaded bumper to bumper.

<table>
<thead>
<tr>
<th>Vehicle</th>
<th>Length (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4 m</td>
</tr>
<tr>
<td>B</td>
<td>7 m</td>
</tr>
<tr>
<td>C</td>
<td>13 m</td>
</tr>
<tr>
<td>D</td>
<td>6 m</td>
</tr>
<tr>
<td>E</td>
<td>13 m</td>
</tr>
<tr>
<td>F</td>
<td>4 m</td>
</tr>
<tr>
<td>G</td>
<td>12 m</td>
</tr>
<tr>
<td>H</td>
<td>14 m</td>
</tr>
<tr>
<td>I</td>
<td>6 m</td>
</tr>
<tr>
<td>J</td>
<td>11 m</td>
</tr>
</tbody>
</table>
   a. Describe one difference between the first-fit and full-bin methods of bin packing. (1 mark)
   b. Use the first-fit algorithm to determine the number of lanes needed to load all the vehicles onto the ferry. (4 marks)
   c. Use a full-bin method to obtain an optimal solution using the minimum number of lanes. Explain why your solution is optimal. (4 marks)

4. The ground floor of an office block is to be fully recarpeted, with specially made carpet incorporating the firm’s logo. The carpet comes in rolls of 15 m.
   The following lengths are required.
<table>
<thead>
<tr>
<th>Length (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 3 m</td>
</tr>
<tr>
<td>B 3 m</td>
</tr>
<tr>
<td>C 4 m</td>
</tr>
<tr>
<td>D 4 m</td>
</tr>
<tr>
<td>E 4 m</td>
</tr>
<tr>
<td>F 4 m</td>
</tr>
<tr>
<td>G 5 m</td>
</tr>
<tr>
<td>H 5 m</td>
</tr>
<tr>
<td>I 5 m</td>
</tr>
<tr>
<td>J 7 m</td>
</tr>
<tr>
<td>K 8 m</td>
</tr>
<tr>
<td>L 8 m</td>
</tr>
</tbody>
</table>
The lengths are arranged in **ascending** order of size.

a. Obtain a lower bound for the number of rolls of carpet needed.  
   **(2 marks)**

b. Use the first-fit decreasing bin-packing algorithm to determine the number of rolls needed.  
   State the length of carpet that is wasted using this method.  
   **(3 marks)**

c. Give one disadvantage of the first-fit decreasing bin-packing algorithm.  
   **(1 mark)**

d. Use a full-bin method to obtain an optimal solution, and state the total length of wasted carpet using this method.  
   **(4 marks)**

---

### E/P

#### 5

Eight computer programs need to be copied onto 40 GB USB sticks. The size of each program is given below.

<table>
<thead>
<tr>
<th>Program</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size (GB)</td>
<td>8</td>
<td>16</td>
<td>17</td>
<td>21</td>
<td>22</td>
<td>24</td>
<td>25</td>
<td>25</td>
</tr>
</tbody>
</table>

a. Use the first-fit decreasing algorithm to determine which programs should be recorded onto each USB stick.  
   **(3 marks)**

b. Calculate a lower bound for the number of USB sticks needed.  
   **(2 marks)**

c. Explain why it is not possible to record these programs on the number of USB sticks found in part b.  
   **(1 mark)**

---

### 1.6 Binary search

You need to be able to carry out a binary search.

A binary search will look through an **ordered** list to find out whether or not an item you are trying to find is in the list. If the item is in the list, the binary search will locate its position within the list.

If the list is not in order, then you may need to use a bubble sort or quick sort to put the items into order first.

- In a binary search, we look at halving the size of the list each time we perform a **pass**.
  - In a binary search, we locate the midpoint of the list using \( \frac{n + 1}{2} \). We call this the **pivot**.
    - Like with the quick sort, we round this up if it is not an integer.
    - We compare this midpoint with the item we are trying to locate; this will help us decide which half of the list to choose.
    - Eventually we will get to one item – it will either be the item we are trying to locate, or it will not be. In this case we can say that the item we were trying to locate is not in the list.

Here is the binary search algorithm to locate an item in a list:

1. Select the midpoint of the list using \( \frac{n + 1}{2} \) and round up if necessary. This is the pivot.
2. a. If the pivot is the item we are locating, then the search is complete.
   b. If the pivot is after the item we are locating, then we look in the first half of the list.
   c. If the pivot is before the item we are locating, then we look in the second half of the list.
3. Repeat steps 1 and 2 to each remaining list until the item is located. If the item is not found, then it is not in the list.
Use the binary search algorithm to try to locate these names in the list that follows.

a) Robinson
   1. Acharya
   2. Blackstock
   3. Cheung
   4. Coetzee
   5. Fowler
   6. Laing
   7. Leung
   8. Robinson
   9. Saludo
   10. Xiao

   b) Davies
   1. Acharya
   2. Blackstock
   3. Cheung
   4. Coetzee
   5. Fowler
   6. Laing
   7. Leung
   8. Robinson
   9. Saludo
   10. Xiao

Remember that a search can be unsuccessful. You may be asked to try to locate something that is not in the list. You must be able to show that the item is not in the list.

Watch out: Remember to round up if \( \frac{n+1}{2} \) is not an integer.

Since Robinson is after Laing, Robinson cannot be in the first part of the list and so we consider the list after the pivot.

Robinson is before Saludo so it cannot be in the second list and so we consider the list before the pivot.

It is important to write this down.

Consider the list before the pivot.

Consider the list after the pivot.
The middle name is the \( \left( \frac{n + 1}{2} = 1.5 \right) \) 2nd name:
2 Fowler
Davies is before Fowler so the list reduces to:
1 Coetzee
The list has only one item which is not Davies.
Therefore Davies is not in the list.

Consider the list before the pivot.
It is important to write this down.

Exercise 1F

1 Use the binary search algorithm to try to locate these names in the list that follows:
   
a Connock
   b Walkey
   c Peabody
   1 Berry
   2 Connock
   3 Li
   4 Sully
   5 Tapner
   6 Walkey
   7 Wilson
   8 Wu

2 Use the binary search algorithm to try to locate these numbers in the list that follows:
   
a 21
   b 5
   1 3 7 10 15 18 21
   2 4 9 13 17 20 24

3 The binary search algorithm is applied to an ordered list of \( n \) items.
   Find the maximum number of times the algorithm is run when \( n \) is equal to:
   
a 100
   b 1000
   c 10000

4 a Use the quick sort algorithm to put the list below into ascending order.
   
   1 Adam
   2 Ed
   3 Lei
   4 Lottie
   5 Saul
   6 Ramin
   7 Alex
   8 Emily
   9 Felix
   10 Leo
   11 Oli
   12 Lotus
   13 Des
   14 George
   15 Jess
   16 Miranda
   17 Matt
   18 Katie
   19 Doug
   20 Hongmei

b Use the binary search algorithm to try to locate:
   
i George     ii David     iii Jess

Chapter review 1

1 Use the bubble-sort algorithm to sort, in ascending order, the list
   
   27 15 2 38 16 1

giving the state of the list at each stage. (4 marks)
Use the bubble-sort algorithm to sort, in descending order, the list
25 42 31 22 26 41

giving the state of the list on each occasion when two or more values are interchanged (swapped).
(4 marks)

Find the maximum number of interchanges needed to sort a list of six pieces of data using the bubble-sort algorithm.
(2 marks)

This list of numbers is to be sorted into ascending order.
Perform a quick sort to obtain the sorted list, giving the state of the list after each rearrangement.
(5 marks)

The list of numbers above is to be sorted into descending order.
Perform a quick sort to obtain the sorted list and indicating the pivot elements used.
(5 marks)

Use the first-fit decreasing bin-packing algorithm to fit the data into bins of size 200.
(3 marks)

Explain how you decided in which bin to place the number 77.
(1 mark)

Trishna wishes to record eight television programmes. The lengths of the programmes, in minutes, are:
75 100 52 92 30 84 42 60
Trishna decides to use 2-hour (120 minute) DVDs only to record all of these programmes.

Explain how to apply the first-fit decreasing bin-packing algorithm.
(2 marks)

Use this algorithm to fit these programmes onto the smallest number of DVDs possible, stating the total amount of unused space on the DVDs.
(3 marks)

Trishna wants to record an additional two 25-minute programmes.

Determine whether she can do this using only 5 DVDs, giving reasons for your answer.
(3 marks)

A DIY enthusiast requires the following 14 pieces of wood as shown in the table.

<table>
<thead>
<tr>
<th>Length in metres</th>
<th>0.4</th>
<th>0.6</th>
<th>1</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of pieces</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The DIY store sells wood in 2 m and 2.4 m lengths. He considers buying six 2 m lengths of wood.

Explain why he will not be able to cut all of the lengths he requires from these six 2 m lengths.
(2 marks)

He eventually decides to buy 2.4 m lengths. Use a first-fit decreasing bin-packing algorithm to show how he could use six 2.4 m lengths to obtain the pieces he requires.
(4 marks)

Obtain a solution that requires only five 2.4 m lengths.
(4 marks)
7 The algorithm described by the flow chart below is to be applied to the five pieces of data below.

\[ u_1 = 6.1, u_2 = 6.9, u_3 = 5.7, u_4 = 4.8, u_5 = 5.3 \]

a Obtain the final output of the algorithm using the five values given for \( u_1 \) to \( u_5 \).

b In general, for any set of values \( u_1 \) to \( u_5 \), explain what the algorithm achieves.

\[
\begin{align*}
\text{Start} \\
\text{Box 1} & \quad i = 1, A = u_1, \quad \text{Temp} = |5 - u_1| \\
\text{Box 2} & \quad i = i + 1 \\
\text{Box 3} & \quad M = |5 - u_i| \\
\text{Box 4} & \quad \text{Is } M < \text{Temp}? \\
\text{Box 5} & \quad \text{No} \quad A = u_i, \quad \text{Temp} = M \\
\text{Box 6} & \quad \text{Is } i < 5? \\
\text{Box 7} & \quad \text{Print } A \\
\text{Stop}
\end{align*}
\]

c If Box 4 in the flow chart is altered to ‘Is \( M > \text{Temp} \)?’ state what the algorithm achieves now.

8 A plumber is cutting lengths of PVC pipe for a bathroom. The lengths needed, in metres, are:

\[ 0.3 \quad 2.0 \quad 1.3 \quad 1.6 \quad 0.3 \quad 1.3 \quad 0.2 \quad 0.1 \quad 2.0 \quad 0.5 \]

The pipe is sold in 2 m lengths.

a Carry out a bubble sort to produce a list of the lengths needed in descending order. Give the state of the list after each pass.

b Apply the first-fit decreasing bin-packing algorithm to your ordered list to determine the total number of 2 m lengths of pipe needed.

c Does the answer to part b use the minimum number of 2 m lengths? You must justify your answer.
9 Here are the names of eight students in an A level group:

Manisha, Vivien, Cath, Alex, Da Ming, Beth, Kandis, Sze-To

Use a quick sort to put the names in alphabetical order. Show the result of each pass and identify the pivots. (5 marks)

Challenge

10 A binary search is to be performed on a list of names to try to locate Kim.

1 Jenny  6 Hyo
2 Merry  7 Kim
3 Charles  8 Richard
4 Ben  9 Greg
5 Toby  10 Freya

a Explain why a binary search cannot be performed with the list in its present form. (1 mark)

b Using an appropriate algorithm, alter the list so that a binary search can be performed, showing the state of the list after each complete iteration. State the name of the algorithm you have used. (4 marks)

c Use the binary search algorithm to locate the name Kim in the list you obtained in b. You must make your method clear. (4 marks)
Summary of key points

1. An algorithm is a finite sequence of step-by-step instructions carried out to solve a problem.

2. In a flow chart, the shape of each box tells you about its function.

3. Unordered lists can be sorted using a bubble sort or a quick sort.

4. In a bubble sort, you compare adjacent items in a list:
   - If they are in order, leave them.
   - If they are not in order, swap them.
   - The list is in order when a pass is completed without any swaps.

5. In a quick sort, you select a pivot and then split the items into two sublists:
   - One sublist contains items less than the pivot.
   - The other sublist contains items greater than the pivot.
   - You then select further pivots from within each sublist and repeat the process.

6. The three bin-packing algorithms are first-fit, first-fit decreasing, and full-bin:
   - The first-fit algorithm works by considering items in the order they are given.
   - The first-fit decreasing algorithm requires the items to be in descending order before applying the algorithm.
   - Full-bin packing uses inspection to select items that combine to fill bins completely. Remaining items are packed using the first-fit algorithm.

7. The three bin-packing algorithms have the following advantages and disadvantages:

<table>
<thead>
<tr>
<th>Type of algorithm</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-fit</td>
<td>Quick to apply</td>
<td>Not likely to lead to a good solution</td>
</tr>
<tr>
<td>First-fit decreasing</td>
<td>Usually a good solution; easy to apply</td>
<td>May not get an optimal solution</td>
</tr>
<tr>
<td>Full-bin</td>
<td>Usually a good solution</td>
<td>Difficult to do, especially when the numbers are plentiful or awkward</td>
</tr>
</tbody>
</table>

8. A binary search will search an ordered list to find out whether an item is in the list. If it is in the list, it will locate its position in the list.

   In a binary search, the pivot is the middle item of the list. If the target item is not the pivot, the pivot and half of the list are discarded. The list length halves at each pass.

   The middle of \( n \) items is found by \( \frac{n + 1}{2} \), rounding up if necessary.
2 GRAPHS AND NETWORKS

Learning objectives

After completing this chapter you should be able to:

- Know how graphs and networks can be used to create mathematical models → pages 30–33
- Be familiar with basic graph theory terminology → pages 34–38
- Know some special types of graph → pages 38–40
- Understand how graphs and networks can be represented using matrices → pages 42–44

Prior knowledge check

1. In the diagram, triangle $ABC$ has sides of lengths $a$, $b$ and $c$.

   Explain why $a < b + c$.

2. Find the shortest path between $A$ and $D$ and write down its length.

   In decision mathematics, a graph or network refers to a set of points joined by edges (line segments). This graph models an underground train network. The lines show which stations are connected but their lengths do not represent distances in real life.
2.1 Modelling with graphs

You need to know how graphs and networks can be used to create mathematical models. In decision mathematics, the word graph has a very specific meaning.

- A graph consists of points (called vertices or nodes) which are connected by lines (edges or arcs).
- If a graph has a number associated with each edge (usually called its weight), then the graph is known as a weighted graph or network.

**Example 1**  
**SKILLS REASONING/ARGUMENTATION**

This graph shows the routes flown by an airline.

a. Explain why this is a graph.
b. Describe what is represented by:
   i. the vertices
   ii. the edges.
c. Describe two possible flight routes from Johannesburg to Libreville.

---

**Example 2**  
**SKILLS INTERPRETATION**

This network shows the lengths of pipes running between taps on a camp site. The lengths on the edges are given in metres.

a. Write down the length of the pipe connecting tap B to tap D.
b. Write a list of the taps that are directly connected to tap E.

---

This is an example of a weighted network.
Example 3

The network shows the times taken, in minutes, by a car to travel along some sections of road.

![Network Diagram]

Calculate the minimum time needed to travel from $A$ to $D$ and give the route taken.

<table>
<thead>
<tr>
<th>Route</th>
<th>Time needed (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ABCD$</td>
<td>$5 + 8 + 1 = 14$</td>
</tr>
<tr>
<td>$ABED$</td>
<td>$5 + 2 + 5 = 12$</td>
</tr>
<tr>
<td>$AED$</td>
<td>$10 + 5 = 15$</td>
</tr>
</tbody>
</table>

The minimum time required is 12 minutes, along the route $ABED$.

Watch out: A weighted network is not usually drawn to scale. The quickest route may not be the route which appears to be the shortest; you need to consider the weight on each arc.

Exercise 2A

1. This graph represents the friendships within a group of students.

   a. State what is represented by each:
      i. vertex
      ii. arc.

   b. List all of Chris’s friends within the group.

   c. State two new friendships that could be created to ensure that every pair of friends has at least one friend in common.
2 This table gives information about the subjects studied by some students.

<table>
<thead>
<tr>
<th>Student</th>
<th>Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mathematics, Chemistry</td>
</tr>
<tr>
<td>B</td>
<td>Mathematics, Chemistry, Biology, Art</td>
</tr>
<tr>
<td>C</td>
<td>Physics, Chemistry, Biology</td>
</tr>
<tr>
<td>D</td>
<td>Mathematics, Physics, Art, English</td>
</tr>
<tr>
<td>E</td>
<td>Biology, English, Art</td>
</tr>
<tr>
<td>F</td>
<td>English, Mathematics, Art, Physics</td>
</tr>
</tbody>
</table>

a Copy and complete the graph to represent this information. Use the arcs to show which subjects each student studies.

b Which subjects are studied by the most students?

3 Here is a map of part of the London underground train system. The colours indicate the different train lines and a circle indicates where you can change from one line to another without leaving the underground.
To go from Lancaster Gate to Hyde Park Corner, for example, you would take the red line from Lancaster Gate to Bond Street, change onto the grey line to Green Park, and finally take the dark blue line to Hyde Park Corner.

a) Work out a possible route from Chancery Lane to Victoria:
   i) that requires a minimum number of changes
   ii) that passes through a minimum number of stations.

A student estimates that it takes 80 seconds to travel from one station to the next, and that it takes 3 minutes to change trains at one of the junction stations.

b) Work out the quickest possible journey time from:
   i) Marylebone to Waterloo
   ii) Victoria to Baker Street
   iii) Holborn to St James’s Park.

4 This weighted network shows the flight times of a particular airline, in minutes, between some airports in the UK and Ireland.

```
<table>
<thead>
<tr>
<th></th>
<th>Aberdeen</th>
<th>Belfast</th>
<th>Cork</th>
<th>Dublin</th>
<th>Edinburgh</th>
<th>Glasgow</th>
<th>Leeds-Bradford</th>
<th>Manchester</th>
<th>Newcastle</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

a) Write down the flight time between Glasgow and Manchester.

b) Name the route with the longest flight time.

c) Suggest which city might be the home base of this airline. Give a reason for your answer.

5 This network shows the lengths, in km, of a network of mountain bike trails.

A student says that the shortest possible route to cycle from P to V is 27 km long.

a) State which route the student has chosen to calculate this distance.

b) State, with a reason, whether or not the student is correct.

Challenge

The diagram shows a wire network in the shape of a cube. An ant walks from vertex A to vertex G along the cube’s edges. Find the total number of possible different routes which:

a) are of minimum length

b) do not pass through any vertex more than once.
2.2 Graph theory

Graph theory is a rapidly growing area of mathematics, mainly due to its applications in many areas. Satellite navigation systems use graph theory to find the fastest or shortest journey between two places. Many project management companies use critical path analysis to schedule how a project can be organised. The famous four colour theorem has also been proved using graph theory ideas.

First, we need to introduce some of the basic ideas and definitions in graph theory.

There are two different types of notation used to describe graphs, as follows:

In graph $G$, above:
- the vertices (nodes) are: $A$, $B$, $C$, $D$, $E$ and $F$ (the list of vertices is sometimes called the vertex set).
- the edges (arcs) are: $AB$, $AC$, $AF$, $BC$, $BD$, $CE$ and $DE$ (the list of edges is sometimes called the edge set).

■ A subgraph of $G$ is a graph, each of whose vertices belongs to $G$ and each of whose edges belongs to $G$. It is simply a part of the original graph.

■ The degree of a vertex is the number of edges that meet at that vertex.

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>$J$</td>
<td>2</td>
</tr>
<tr>
<td>$K$</td>
<td>2</td>
</tr>
<tr>
<td>$L$</td>
<td>1</td>
</tr>
<tr>
<td>$M$</td>
<td>4</td>
</tr>
<tr>
<td>$N$</td>
<td>1</td>
</tr>
</tbody>
</table>

- If the degree of a vertex is even, it has even degree, so $J$, $K$ and $M$ have even degree. Vertices $L$ and $N$ have odd degree.
- A walk is a route through a graph along edges from one vertex to the next.
- A **path** is a walk in which no vertex is visited more than once.
- A **trail** is a walk in which no edge is visited more than once.
- A **cycle** is a walk in which the end vertex is the same as the start vertex and no other vertex is visited more than once.
- A **Hamiltonian cycle** is a cycle that includes every vertex.

In the graph below, an example of:
- a walk is **RSUVW** — On a walk, it is okay to include a vertex or an edge more than once.
- a path is **RSUVW**
- a trail is **RUSVUW**
- a cycle is **RSUR** — In a cycle, it is not necessary to include every vertex.
- a Hamiltonian cycle is **RSUVWTR**.

- Two vertices are **connected** if there is a path between them.
  A **graph** is connected if all its vertices are connected.

This is a **connected** graph. A path can be found between any two vertices.
This graph is **not connected**. There is no path from *R* to *V*, for example.

- A **loop** is an edge that starts and finishes at the same vertex.

This contains a **loop** from *C* to *C*.

- Vertex *C* has degree 3.

- A **simple graph** is one in which there are no loops and there is no more than one edge connecting any pair of vertices.

This is not a simple graph because it has two edges connecting *D* and *E.*
If the edges of a graph have a direction associated with them they are known as **directed edges** and the graph is known as a **directed graph**, often abbreviated to **digraph**.

In any undirected graph, the sum of the degrees of the vertices is equal to $2 \times$ the number of edges. As a result, the number of odd vertices must be even, including possibly zero. This result is known as **Euler’s handshaking lemma**.

### Example 4

**a** Find the sum of the valencies of the vertices in this graph.

**b** Verify that your answer to part **a** is twice the number of edges in the graph.

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Valency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>3</td>
</tr>
<tr>
<td>$B$</td>
<td>3</td>
</tr>
<tr>
<td>$C$</td>
<td>3</td>
</tr>
<tr>
<td>$D$</td>
<td>3</td>
</tr>
<tr>
<td>$E$</td>
<td>4</td>
</tr>
<tr>
<td>$F$</td>
<td>2</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>18</strong></td>
</tr>
</tbody>
</table>

**b** The graph has 9 edges.

Sum of valencies $= 18 = 2 \times 9$

$= 2 \times $ number of edges

Notice that the number of vertices of odd degree is even (there are four of them).

If each vertex represents a person and each edge indicates that two people have shaken hands, then the number of people who have shaken hands an odd number of times must be even.

### Example 5

A graph has 5 nodes and 8 edges. The valencies of the nodes are $x, x - 1, x + 1, 2x - 1$ and $x - 1$. Find the value of $x$.

Sum of valencies

$= x + x - 1 + x + 1 + 2x - 1 + x - 1$

$= 6x - 2$

Sum of valencies $= 2 \times $ number of edges

$= 2 \times 8 = 16$

$6x - 2 = 16$

$6x = 18$

$x = 3$

**Problem-solving**

Use Euler’s handshaking lemma to formulate an equation in $x$. 

A **lemma** is a small theorem used as a stepping stone to more important results.
1. Draw a connected graph with:
   a. one vertex of degree 4 and four vertices of degree 1
   b. three vertices of degree 2, one of degree 3 and one of degree 1
   c. two vertices of degree 2, two of degree 3 and one of degree 4.

2. Which of the graphs below are not simple?
   a. A
   b. F
   c. T
   d. UV

3. In question 2, which graphs are not connected?

4. For the graph on the right, state:
   a. four paths from F to D
   b. a cycle passing through F and D
   c. the degree of each vertex.
   Use the graph to:
   d. draw a subgraph
   e. confirm the handshaking lemma.

5. Repeat question 4 parts c, d and e using this graph:

6. Show that it is possible to draw a graph with:
   a. an even number of vertices of even degree
   b. an odd number of vertices of even degree.
   c. Explain why it is not possible to draw a graph with an odd number of vertices of odd degree.
7 Copy the diagram and complete a digraph for the relationship ‘is a factor of’.

8 a Explain what is meant by a Hamiltonian cycle. (2 marks)

b For this graph, list all of the Hamiltonian cycles that start from the vertex P. (4 marks)

c Explain what is meant by a subgraph. (2 marks)

d Draw a subgraph of this graph with 4 vertices. (1 mark)

9 A graph has 7 nodes and 14 edges. The valencies of the nodes are: x, x², x + 1, 3x – 1, 2x + 1, 3x – 2, 4x – 3. Find the value of x.

2.3 Special types of graph

You need to know about the following special types of graph.

- A **tree** is a connected graph with no cycles.

- A **spanning tree** of a graph G is a subgraph which includes all the vertices of G and is also a tree.
A complete graph is a graph in which every vertex is directly connected by a single edge to each of the other vertices.

Isomorphic graphs are graphs which show the same information but may be drawn differently.

For the following graph, draw two spanning trees.

A spanning tree must be a tree (no cycles) which means all nodes are connected by arcs.

There are many other possible spanning trees.

Example 6

SKILLS INTERPRETATION

Possible spanning trees are:

A complete graph is a graph in which every vertex is directly connected by a single edge to each of the other vertices.

Notation

The complete graph with $n$ vertices is written as $K_n$. 
For two graphs to be isomorphic, they must have the same number of vertices of the same degree, and these vertices must also be connected together in the same ways. If there is an edge between two vertices in one graph, then there is also an edge between the two corresponding vertices in the other graph.

**Example 7**

Show that the two graphs below are isomorphic.

Both graphs have the same number of vertices. Each graph has the same number of vertices of the same degree.

<table>
<thead>
<tr>
<th>First graph</th>
<th>Second graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex</td>
<td>Degree</td>
</tr>
<tr>
<td>N</td>
<td>4</td>
</tr>
<tr>
<td>P</td>
<td>3</td>
</tr>
<tr>
<td>K</td>
<td>2</td>
</tr>
<tr>
<td>L</td>
<td>2</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
</tr>
<tr>
<td>M</td>
<td>1</td>
</tr>
<tr>
<td>Q</td>
<td>1</td>
</tr>
</tbody>
</table>

N can be paired with S.
P can be paired with U.
K can be paired with V.
L can be paired with X.
J can be paired with R.
M can be paired with T.
Q can be paired with W.

Since the pairing can be made, the two graphs are isomorphic.

Pair up the vertices, starting with N and S as they are the only vertices which have degree 4.

You will need to check that the pairings create the graphs in the original question.

If graphs are isomorphic it is possible to pair equivalent vertices. The pairs of equivalent vertices for the graphs above are shown in this table.

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Vertex paired with in first graph</th>
<th>Vertex paired with in second graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>J</td>
<td>R</td>
</tr>
<tr>
<td>B</td>
<td>M</td>
<td>T</td>
</tr>
<tr>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>D</td>
<td>L</td>
<td>V</td>
</tr>
<tr>
<td>E</td>
<td>Q</td>
<td>W</td>
</tr>
<tr>
<td>F</td>
<td>P</td>
<td>U</td>
</tr>
<tr>
<td>G</td>
<td>K</td>
<td>X</td>
</tr>
</tbody>
</table>
Exercise 2C

1. State which of the following graphs are trees.
   a)
   b)
   c)
   d)

2. There are 11 spanning trees for the graph shown below. Draw them all.

   A
   B
   C
   E
   D

3. For the graph on the right, identify a complete subgraph that has 4 vertices.

4. Identify which of graphs A, B and C are isomorphic to graph D.

5. a) Define the terms:
   i) tree
   ii) spanning tree.
   b) Explain why it is not possible to construct a spanning tree for this graph.

   b) State the degree of each vertex in the graph $K_n$.
   c) Find the total number of edges in the graph $K_{20}$.

Challenge: Draw all possible connected graphs with three edges.
2.4 Representing graphs and networks using matrices

You can use an **adjacency matrix** to represent a graph or network. The adjacency matrix provides information about the connections between the vertices in a graph.

- Each entry in an adjacency matrix describes the number of arcs joining the corresponding vertices.

**Example 8**

Use an adjacency matrix to represent this graph.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

This indicates that there are 2 direct connections between B and E.

This indicates a loop from F to F. It could be travelled in either direction, and hence counts as 2.

The **matrix** associated with a weighted graph is called a **distance matrix**.

- In a distance matrix, the entries represent the weight of each arc, not the number of arcs.

**Example 9**

Use a **distance matrix** to represent this network.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>—</td>
<td>17</td>
<td>18</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B</td>
<td>17</td>
<td>—</td>
<td>15</td>
<td>19</td>
<td>23</td>
</tr>
<tr>
<td>C</td>
<td>18</td>
<td>15</td>
<td>—</td>
<td>20</td>
<td>—</td>
</tr>
<tr>
<td>D</td>
<td>—</td>
<td>19</td>
<td>20</td>
<td>—</td>
<td>16</td>
</tr>
<tr>
<td>E</td>
<td>23</td>
<td>—</td>
<td>16</td>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>

Notice that the matrix is symmetrical about the leading diagonal (top left to bottom right). This will be the case for any non-directed network.

**Watch out**

- You should be able to write down the adjacency matrix given a graph, and draw a graph given the adjacency matrix.
- You should be able to write down the distance matrix given the network, and draw the network given the distance matrix.
**Example 10**

Use a distance matrix to represent this directed network.

<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>S</th>
<th>T</th>
<th>U</th>
<th>V</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>—</td>
<td>4</td>
<td>11</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>S</td>
<td>4</td>
<td>—</td>
<td>9</td>
<td>—</td>
<td>10</td>
<td>—</td>
</tr>
<tr>
<td>T</td>
<td>11</td>
<td>—</td>
<td>8</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>U</td>
<td>—</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>V</td>
<td>—</td>
<td>—</td>
<td>7</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>W</td>
<td>—</td>
<td>10</td>
<td>—</td>
<td>—</td>
<td>9</td>
<td>—</td>
</tr>
</tbody>
</table>

**Watch out** This matrix is not symmetrical about the leading diagonal.

This indicates a direct link of weight 6 from U to V.

This indicates a direct link of weight 7 from V to U.

This shows a direct link of weight 9 from W to V.

This indicates that V is not directly linked to W.

**Exercise 2D**

1. Use an adjacency matrix to represent the graph below.

![Graph](image)

2. Draw a graph corresponding to each adjacency matrix.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>d</td>
<td>A</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
3. Draw the network corresponding to each distance matrix.

\[
\begin{array}{|c|ccccc|}
\hline
& A & B & C & D & E \\
\hline
A & - & 21 & - & 20 & 23 \\
B & 21 & - & 17 & 23 & - \\
C & - & 17 & - & 18 & 41 \\
D & 20 & 23 & 18 & - & 22 \\
E & 23 & - & 41 & 22 & - \\
\hline
\end{array}
\]

\[
\begin{array}{|c|cccccc|}
\hline
& A & B & C & D & E & F \\
\hline
A & - & - & - & - & 15 & 8 \\
B & - & - & 9 & 13 & 17 & - \\
C & - & 9 & - & 8 & - & - \\
D & 13 & 8 & - & - & 10 & - \\
E & 15 & 17 & - & 10 & - & - \\
F & 8 & 11 & - & - & - & - \\
\hline
\end{array}
\]

4. Use a distance matrix to represent the directed network below.

5. a. Draw a graph to represent this adjacency matrix.

\[
\begin{array}{|c|ccccc|}
\hline
& A & B & C & D & E \\
\hline
A & 0 & 2 & 0 & 0 & 1 \\
B & 2 & 0 & 1 & 0 & 1 \\
C & 0 & 1 & 0 & 1 & 1 \\
D & 0 & 0 & 1 & 0 & 1 \\
E & 1 & 1 & 1 & 1 & 0 \\
\hline
\end{array}
\]

\textbf{Watch out} This is an adjacency matrix, not a distance matrix.

b. For this graph, a spanning tree may be drawn such that one vertex connects to every edge. Which vertex is this?

c. Draw the spanning tree referred to in part b.

\textbf{Chapter review 2}

1. \(A, B, C\) and \(D\) are four towns. The distance matrix shows the direct distances by road between the towns in kilometres.

\[
\begin{array}{|c|cccc|}
\hline
& A & B & C & D \\
\hline
A & - & 11 & 15 & 9 \\
B & 11 & - & - & 8 \\
C & 15 & - & - & 16 \\
D & 9 & 8 & 16 & - \\
\hline
\end{array}
\]

Draw a weighted graph to show this information.
2 Which of these graphs are isomorphic?

a  

b  

c  

d  

e  

f  

g  

h  

i  

3 a Draw a graph with eight vertices, all of degree 1.

b Draw a graph with eight vertices, all of degree 2, so that the graph is:

i connected and simple  
ii not connected and simple  
iii not connected and not simple.

4 a Describe the difference between a distance matrix and an adjacency matrix.

b Use a distance matrix to represent the network below.

![Network Diagram](image)

c Draw a spanning tree for this network and state the weight of your spanning tree.

5 Write an expression for the number of edges in a spanning tree containing \( v \) vertices.

6 Write down all the possible routes from \( P \) to \( R \) in this digraph.

![Digraph Diagram](image)
7. The diagram shows a graph.

a. Write down the valency of vertex A. (1 mark)

b. Identify:
   i. a path from B to D (1 mark)
   ii. a cycle. (1 mark)

c. Sketch:
   i. a spanning tree for this graph (1 mark)
   ii. a complete subgraph of this graph. (2 marks)

8. a. Explain why it is not possible to draw a graph with exactly four vertices with degrees 3, 1, 2, and 1. (1 mark)

   A connected graph has exactly four vertices and 10 edges.
   The degrees of the vertices are $k^2 - 3k$, $k + 1$, $8 - k$ and $k - 4$ respectively.

b. Find the value of $k$. (3 marks)

9. A Hamiltonian cycle for this graph begins $AGB…$

a. Complete the Hamiltonian cycle. (2 marks)

b. For the graph below, write down a Hamiltonian cycle starting from A. (2 marks)
A graph consists of points (called vertices or nodes) connected by lines (edges or arcs).

If a graph has a number associated with each edge (usually called its weight), then the graph is known as a weighted graph or network.

A subgraph is a graph, each of whose vertices belongs to the original graph and each of whose edges belongs to the original graph. It is part of the original graph.

The degree (or valency or order) of a vertex is the number of edges that meet at that vertex.

If the degree of a vertex is even, you say that it has even degree. If the degree of a vertex is odd it has odd degree.
6 A walk is a route through a graph along edges from one vertex to the next.

7 A path is a walk in which no vertex is visited more than once.

8 A trail is a walk in which no edge is visited more than once.

9 A cycle is a walk in which the end vertex is the same as the start vertex and no other vertex is visited more than once.

10 A Hamiltonian cycle is a cycle that includes every vertex.

11 Two vertices are connected if there is a path between them. A graph is connected if all its vertices are connected.

12 A loop is an edge that starts and finishes at the same vertex.

13 A simple graph is one in which there are no loops and there is at most one edge connecting any pair of vertices.

14 If the edges of a graph have a direction associated with them they are known as directed edges and the graph is known as a directed graph (or digraph).

15 In any undirected graph, the sum of the degrees of the vertices is equal to 2 × the number of edges. As a consequence, the number of odd nodes must be even. This result is called Euler’s handshaking lemma.

16 A tree is a connected graph with no cycles.

17 A spanning tree of a graph is a subgraph which includes all the vertices of the original graph and is also a tree.

18 A complete graph is a graph in which every vertex is directly connected by a single edge to each of the other vertices.

19 Isomorphic graphs are graphs which show the same information but may be drawn differently.

20 Each entry in an adjacency matrix describes the number of arcs joining the corresponding vertices.

21 In a distance matrix, the entries represent the weight of each arc, not the number of arcs.
3 ALGORITHMS ON GRAPHS

Learning objectives

After completing this chapter you should be able to:

- Use Kruskal’s algorithm to find a minimum spanning tree → pages 50–54
- Use Prim’s algorithm on a network to find a minimum spanning tree → pages 54–56
- Apply Prim’s algorithm to a distance matrix → pages 57–60
- Apply the nearest neighbour algorithm to find a short path → pages 60–63
- Use Dijkstra’s algorithm to find the shortest path between two vertices in a network → pages 64–71

Prior knowledge check

1 The diagram shows a weighted network.

a State the order of vertex C.

b Draw a distance matrix for this network.

c Find a spanning tree with total weight less than 40. State clearly which arcs are included in your spanning tree and its total weight. ← Decision 1 Chapter 2

Map apps model roads as arcs on a graph, and add weightings depending on traffic conditions. By finding the shortest path on the graph, the app can compute the quickest route to a destination.
3.1 Kruskal’s algorithm

You can use Kruskal’s algorithm to find a minimum spanning tree. This can tell you the shortest, cheapest or fastest way of linking all the nodes in a network.

- A minimum spanning tree (MST) is a spanning tree such that the total length of its arcs (edges) is as small as possible.
- Kruskal’s algorithm can be used to find a minimum spanning tree:

1. Sort all the arcs (edges) into ascending order of weight.
2. Select the arc of least weight to start the tree.
3. Consider the next arc of least weight.
   - If it would form a cycle with the arcs already selected, reject it.
   - If it does not form a cycle, add it to the tree.
   - If there is a choice of equal arcs, consider each in turn.
4. Repeat step 3 until all vertices are connected.

Example 1

Use Kruskal’s algorithm to find a minimum spanning tree for this network. List the arcs in the order that you consider them. State the weight of your tree.

In a network of \( n \) vertices, a spanning tree will always have \((n - 1)\) arcs. In this case there will be 4 arcs in the spanning tree.

By inspection, the order of the arcs is \(DE(4), AE(5), BC(5), AD(6), BD(6), AB(7), CD(8)\). Start with \(DE\).

\(AE\) and \(BC\) could have been written in either order, as both have weight 5. Likewise, \(AD\) and \(BD\) could have been written in either order.

You do not need to draw each of these diagrams. The list of arcs, in order, with your decision about rejecting or adding them, is sufficient to make your method clear.

Online Explore Kruskal’s algorithm using GeoGebra.
Example 2

Use Kruskal’s algorithm and show that there are four minimum spanning trees for this network. State their weight.

By inspection, the order of the arcs is $AD(8)$, $BC(8)$, $AC(10)$, $CD(10)$, $EF(11)$, $CE(12)$, $DE(12)$, $DF(13)$, $BE(14)$.

Start with $AD$.

Add $BC$.

Reject $AD$ (it would make the cycle $AEDA$).

Add $BD$.

All vertices are connected so this is a minimum spanning tree.
Its weight is $5 + 4 + 6 + 5 = 20$

Watch out The spanning tree does not necessarily remain connected as it grows, but the finished spanning tree must be connected.

All the vertices are now connected so you can stop.

If you continued to work through the list of arcs, each would be rejected because they would make cycles.

Remember that you can choose $AD$ and $BC$ in either order, $AC$ and $CD$ in either order, and $CE$ and $DE$ in either order.

You may find it helpful to draw out the tree as you go. It makes it easier to check for cycles.

Watch out You need to consider all the arcs in turn, even when they have equal weight. If you start with $AD$ then you have to consider $BC$ next.
Add $AC$.

Reject $CD$ (it forms a cycle $ACDA$).

Add $EF$.

Add $CE$.

One solution is:

The other three solutions are:

The weight of each tree is $8 + 8 + 10 + 11 + 12 = 49$

Remember, the spanning tree does not necessarily remain connected as it grows, but the finished spanning tree must be connected.

The minimum spanning tree that you create will depend on how you choose the arcs. Each one is a solution because they will all have the least weight.

All four solutions are minimum spanning trees, so they will all have the same total weight.
1. Use Kruskal’s algorithm to find minimum spanning trees for each of these networks. State the weight of each tree. You must list the arcs in the order in which you consider them.

a

\[ \begin{array}{ccc}
A & 25 & B \\
14 & C & 16 \\
H & 20 & C \\
18 & 21 & D \\
G & 12 & E \\
F & 17 & 11 \\
B & 18 & A \\
D & 24 & H \\
E & 15 & G \\
A & 18 & F \\
\end{array} \]

b

\[ \begin{array}{ccc}
A & 7 & C \\
4 & B & 6 \\
H & 5 & 7 \\
G & 3 & 5 \\
E & 2 & D \\
F & 17 & G \\
B & 18 & E \\
D & 16 & F \\
C & 15 & H \\
F & 11 & J \\
C & 8 & D \\
\end{array} \]

c

\[ \begin{array}{ccc}
A & 3.8 & B \\
4.3 & H & 4.2 \\
G & 3.1 & 3.8 \\
J & 2.1 & 2.3 \\
C & 2.2 & 1.7 \\
F & 2.1 & 1.4 \\
D & 2.1 & E \\
H & 4.1 & G \\
J & 4.1 & H \\
F & 3.2 & G \\
\end{array} \]

2. a State what is meant by:
   i. a tree
   ii. a minimum spanning tree.

b Use Kruskal’s algorithm to find a minimum spanning tree for this network.

\[ \begin{array}{ccc}
S & 18 & X \\
22 & U & 18 \\
T & 20 & 19 \\
23 & 16 & 18 \\
V & 17 & 18 \\
W & 17 & 15 \\
X & 17 & Y \\
Y & 15 & Z \\
\end{array} \]

c Draw the minimum spanning tree found in part b.

d State, giving a reason, whether or not this minimum spanning tree is unique.

3. Draw a network in which:
   a. the three shortest edges form part of the minimum connector (MST)
   b. not all of the three shortest edges form part of the minimum connector.
4. The diagram shows nine villages and the distances between them in kilometres. A cable TV company plans to link up the villages.

![Diagram of nine villages with distances between them.]

a. Find a minimum spanning tree for the network using Kruskal’s algorithm. List the arcs in the order that they were added to the tree. 

b. Use your answer to part a to find the minimum length of cable required to link all the villages together. 

(4 marks)

(1 mark)

### 3.2 Prim’s algorithm

- Prim’s algorithm can be used to find a minimum spanning tree:

1. Choose any vertex to start the tree.
2. - Select an arc of least weight that joins a vertex already in the tree to a vertex not yet in the tree. 
   - If there is a choice of arcs of equal weight, choose any of them. 
3. Repeat step 2 until all the vertices are connected. 
4. List the arcs in the order that they were added.

**Notation**

Prim’s algorithm considers **vertices**, whereas Kruskal’s algorithm considers **edges**. Because we are considering adding an arc from an unconnected vertex to the graph, a cycle will never be formed and so we don’t need to check for cycles.

**Example 3**

Use Prim’s algorithm to find a minimum spanning tree for the network above. List the arcs in the order in which you add them to your tree.

**Online**

Explore Prim’s algorithm using GeoGebra.

**Watch out**

In your exam, you may be asked to use Prim’s or Kruskal’s algorithm. You must know which is which.
Choose to start the tree at $A$.
Add the arc of least weight, $AF$, to the tree.

The arcs to consider are those linking $A$ to another vertex: $AF(8)$, $AB(9)$, $AD(9)$, $AE(11)$ and $AC(12)$. Add the one of least weight, $AF$, to the tree.

Now consider arcs that link either $A$ or $F$ to a vertex not in the tree: $AB(9)$, $AD(9)$, $AE(11)$, $AC(12)$, $FE(12)$ and $FB(15)$.

Add the arc of least weight, from $A$ or $F$, that introduces a new vertex to the tree. In this case there are two arcs of least weight, $AD$ and $AB$. You can choose either. In this case, $AD$ is chosen.

The arcs linking $A$, $F$ and $D$ to the remaining vertices are $AB(9)$, $AE(11)$, $AC(12)$, $FE(12)$, $FB(15)$, $DC(8)$, $DE(14)$. The one of least weight is $DC$.

Add $AD$ to the tree.

The arcs linking $A$, $F$, $D$ and $C$ to the remaining vertices are $AB(9)$, $AE(11)$, $FE(12)$, $FB(15)$, $DE(14)$, $CB(10)$. The one of least weight is $AB$.

Add $DC$ to the tree.

The arcs linking $A$, $F$, $D$, $C$ and $B$ to the remaining vertex, $E$, are $AE(11)$, $FE(12)$, $DE(14)$. The one of least weight is $AE$.

Add $AB$ to the tree.

Add $AE$ to the tree.

Arcs added in this order: $AF$, $AD$, $DC$, $AB$, $AE$.

Notice that with Prim’s algorithm the tree always grows in a connected fashion. The arcs do not jump around as they sometimes do with Kruskal’s algorithm.
**Exercise 3B SKILLS PROBLEM-SOLVING**

1. Repeat Question 1 in Exercise 3A using Prim’s algorithm. Start at vertex $A$ each time.

2. Describe two differences between Prim’s algorithm and Kruskal’s algorithm.

3. The network shows the distances, in kilometres, between eight weather monitoring stations. The eight stations need to be linked together with underground cables.
   
   a. Use Prim’s algorithm, starting at $A$, to find a minimum spanning tree. You must make your order of arc selection clear.  
   
   b. Given that cable costs £850 per kilometre to lay, find the minimum cost of linking these weather stations.

4. The network shows ten villages and the costs, in thousands of dollars, of connecting them with a new energy supply.
   
   a. Use Prim’s algorithm, starting at $P$, to find the energy supply network that would connect all ten villages for minimum cost.
   
   b. Draw your minimum connector and state its cost.
   
   c. Unforeseen problems with the link between villages $W$ and $X$ mean that the cost of connecting them rises to $34,000. Explain how this affects your minimum spanning tree.

5. a. Explain why it is not necessary to check for cycles when using Prim’s algorithm.

   b. Use Prim’s algorithm, starting at $A$, to find a minimum spanning tree for this network. You must make your order of arc selection clear.

   c. State, with a reason, whether this minimum spanning tree is unique.
3.3 Applying Prim’s algorithm to a distance matrix

Networks (especially large ones) are often described using distance matrices. This is a convenient way to input a large network into a computer. You can apply Prim’s algorithm directly to a distance matrix, which makes it more convenient for computer applications.

The distance matrix form of Prim’s algorithm is:

1. Choose any vertex to start the tree.
2. Delete the row in the matrix for the chosen vertex.
3. Number the column in the matrix for the chosen vertex.
4. Circle the lowest undeleted entry in the numbered columns. (If there is an equal choice, choose randomly.)
5. The circled entry becomes the next arc to be added to the tree.
6. Repeat steps 2, 3, 4 and 5 until all rows have been deleted.
7. List the arcs in the order that they were added.

Example

Apply Prim’s algorithm to the distance matrix to find a minimum spanning tree. Start at A.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>–</td>
<td>27</td>
<td>12</td>
<td>23</td>
<td>74</td>
</tr>
<tr>
<td>B</td>
<td>27</td>
<td>–</td>
<td>47</td>
<td>15</td>
<td>71</td>
</tr>
<tr>
<td>C</td>
<td>12</td>
<td>47</td>
<td>–</td>
<td>28</td>
<td>87</td>
</tr>
<tr>
<td>D</td>
<td>23</td>
<td>15</td>
<td>28</td>
<td>–</td>
<td>75</td>
</tr>
<tr>
<td>E</td>
<td>74</td>
<td>71</td>
<td>87</td>
<td>75</td>
<td>–</td>
</tr>
</tbody>
</table>

1. Choose any vertex to start the tree.
2. Delete the row in the matrix for the chosen vertex.
3. Number the column in the matrix for the chosen vertex.
4. Circle the lowest undeleted entry in the numbered columns. (If there is an equal choice, choose randomly.)
5. The circled entry becomes the next arc to be added to the tree.
6. Repeat steps 2, 3, 4 and 5 until all rows have been deleted.
7. List the arcs in the order that they were added.

The first arc is AC.
The second arc is $AD$.

The third arc is $DB$.

The fourth arc is $BE$.

The new vertex is $C$. Delete row $C$, and number column $C$.

The lowest undeleted entry in columns $A$ and $C$ is 23, circle it. The second arc is $AD$.

The new vertex is $D$. Delete row $D$, and number column $D$.

The lowest undeleted entry in columns $A$, $C$ and $D$ is 15. Circle it. The third arc is $DB$.

The lowest undeleted entry in columns $A$, $C$, $D$ and $B$ is 71. Circle it. The fourth arc is $BE$.

The new vertex is $B$. Delete row $B$, and number column $B$.

The new vertex is $E$. Delete row $E$ and number column $E$.

All rows have now been deleted, so the algorithm is complete.

You do not need to show all of these tables. The final annotated table, plus a list of arcs in order, is sufficient to make your method clear.
1. Apply Prim’s algorithm to the distance matrices below. List the arcs in order of selection and state the weight of your minimum spanning tree.

\[
\begin{array}{ccccccc}
& A & B & C & D & E & F \\
A & - & 15 & 20 & 34 & 25 & 9 \\
B & 15 & - & 36 & 38 & 28 & 14 \\
C & 20 & 36 & - & 38 & 22 & 14 \\
D & 34 & 38 & 36 & - & 26 & 40 \\
E & 25 & 28 & 38 & 26 & - & 31 \\
F & 9 & 14 & 22 & 40 & 31 & - \\
\end{array}
\]

\[
\begin{array}{ccccccc}
& R & S & T & U & V \\
R & - & 28 & 30 & 31 & 41 & \\
S & 28 & - & 16 & 19 & 43 & \\
T & 30 & 16 & - & 22 & 41 & \\
U & 31 & 19 & 22 & - & 37 & \\
V & 41 & 43 & 41 & 37 & - & - \\
\end{array}
\]

2. The table shows the distance, in kilometres, between five cities. It is desired to link these five cities by a transit system.

<table>
<thead>
<tr>
<th></th>
<th>Birmingham</th>
<th>Nottingham</th>
<th>Lincoln</th>
<th>Stoke</th>
<th>Manchester</th>
</tr>
</thead>
<tbody>
<tr>
<td>Birmingham</td>
<td>-</td>
<td>262.4</td>
<td>160</td>
<td>78.4</td>
<td>140.8</td>
</tr>
<tr>
<td>Nottingham</td>
<td>262.4</td>
<td>-</td>
<td>59.2</td>
<td>89.6</td>
<td>118.4</td>
</tr>
<tr>
<td>Lincoln</td>
<td>160</td>
<td>59.2</td>
<td>-</td>
<td>144</td>
<td>137.6</td>
</tr>
<tr>
<td>Stoke</td>
<td>78.4</td>
<td>89.6</td>
<td>144</td>
<td>-</td>
<td>70.4</td>
</tr>
<tr>
<td>Manchester</td>
<td>140.8</td>
<td>118.4</td>
<td>137.6</td>
<td>70.4</td>
<td>-</td>
</tr>
</tbody>
</table>

Use Prim’s algorithm, starting at Birmingham, to find the minimum total length of a transit system linking all five cities. You must list the arcs in order of selection and state the weight of your tree.

3. The matrix shows the costs, in euros per 1000 words, of translating appliance instruction manuals between eight languages.

\[
\begin{array}{cccccccc}
& A & B & C & D & E & F & G & H \\
A & - & 84 & 53 & 35 & - & 47 & - & 42 \\
B & 84 & - & 71 & 113 & 142 & 61 & 75 & -  \\
C & 53 & 71 & - & - & - & - & 59 & - \\
D & 35 & 113 & - & - & 58 & 67 & 151 & - \\
E & - & 142 & - & 58 & - & 168 & 159 & 48 \\
F & 47 & 61 & - & 67 & 168 & - & - & 73 \\
G & - & 75 & 59 & 151 & 159 & - & - & 52 \\
H & 42 & - & - & 48 & 73 & 52 & - & - \\
\end{array}
\]

a. Use Prim’s algorithm, starting from language D, to find the cost of translating an instruction manual of 3000 words from D into the seven other languages. (4 marks)
**b** Draw your minimum spanning tree.  

A manual is written in language $E$ and needs to be translated into language $G$. The table shows that it costs 159 euros per 1000 words to translate from language $E$ to $G$.

c  Give a reason why:
   i  it might be decided not to translate directly from $E$ to $G$
   ii  it might be decided to translate directly.

(2 marks)

4 The table shows the distances, in kilometres, between nine oil rigs and the depot $X$. Pipes are to be laid to connect the rigs and the depot.  

<table>
<thead>
<tr>
<th></th>
<th>$X$</th>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
<th>$D$</th>
<th>$E$</th>
<th>$F$</th>
<th>$G$</th>
<th>$H$</th>
<th>$I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>–</td>
<td>65</td>
<td>80</td>
<td>89</td>
<td>74</td>
<td>26</td>
<td>71</td>
<td>41</td>
<td>41</td>
<td>74</td>
</tr>
<tr>
<td>$A$</td>
<td>65</td>
<td>–</td>
<td>27</td>
<td>41</td>
<td>22</td>
<td>37</td>
<td>20</td>
<td>29</td>
<td>25</td>
<td>43</td>
</tr>
<tr>
<td>$B$</td>
<td>80</td>
<td>27</td>
<td>–</td>
<td>30</td>
<td>24</td>
<td>55</td>
<td>16</td>
<td>46</td>
<td>40</td>
<td>42</td>
</tr>
<tr>
<td>$C$</td>
<td>89</td>
<td>41</td>
<td>30</td>
<td>–</td>
<td>50</td>
<td>84</td>
<td>24</td>
<td>70</td>
<td>49</td>
<td>26</td>
</tr>
<tr>
<td>$D$</td>
<td>74</td>
<td>22</td>
<td>24</td>
<td>50</td>
<td>–</td>
<td>51</td>
<td>35</td>
<td>34</td>
<td>47</td>
<td>63</td>
</tr>
<tr>
<td>$E$</td>
<td>26</td>
<td>37</td>
<td>55</td>
<td>84</td>
<td>51</td>
<td>–</td>
<td>52</td>
<td>18</td>
<td>23</td>
<td>68</td>
</tr>
<tr>
<td>$F$</td>
<td>71</td>
<td>20</td>
<td>16</td>
<td>24</td>
<td>35</td>
<td>52</td>
<td>–</td>
<td>45</td>
<td>31</td>
<td>27</td>
</tr>
<tr>
<td>$G$</td>
<td>41</td>
<td>29</td>
<td>46</td>
<td>70</td>
<td>34</td>
<td>18</td>
<td>45</td>
<td>–</td>
<td>25</td>
<td>64</td>
</tr>
<tr>
<td>$H$</td>
<td>41</td>
<td>25</td>
<td>40</td>
<td>49</td>
<td>47</td>
<td>23</td>
<td>31</td>
<td>25</td>
<td>–</td>
<td>44</td>
</tr>
<tr>
<td>$I$</td>
<td>74</td>
<td>43</td>
<td>42</td>
<td>26</td>
<td>63</td>
<td>68</td>
<td>27</td>
<td>64</td>
<td>44</td>
<td>–</td>
</tr>
</tbody>
</table>

a  Use Prim’s algorithm, starting at $X$, to find a minimum spanning tree for the network. You must make the order of arc selection clear.  

(3 marks)

b  $X$ must now be directly connected to $D$ and to $F$. Based on this condition, use Prim’s algorithm, starting at $A$, to find a minimum spanning tree. You must make the order of arc selection clear.  

(2 marks)

c  The oil company now requires that the depot must be directly connected to oil rig $D$. Find a minimum spanning tree that must meet this condition.  

(3 marks)

3.4 The nearest neighbour algorithm

The nearest neighbour algorithm can be used to find a short path connecting all vertices. You will see this again when we investigate the travelling salesman problem. The algorithm works only if we use a least distance matrix between each node. This will be expanded further in Chapter 5.

1  Choose any vertex to start the path.
2  Select an arc of least weight which joins a new vertex to the last vertex added only.
3  Repeat step 2 until all of the vertices are added.

The path may not be optimal, but is a key part in the travelling salesman problem.

In Example 4, we saw Prim’s algorithm applied to a distance matrix. Let us now see the nearest neighbour algorithm applied.

Watch out  Do not confuse the nearest neighbour algorithm with Prim’s algorithm. In Prim’s algorithm you look for the vertex nearest any of the vertices in your growing tree. In the nearest neighbour algorithm, you look for the vertex nearest to the last vertex chosen.
Apply the nearest neighbour algorithm to the distance matrix to find a short path connecting all vertices. Start at \( A \).

<table>
<thead>
<tr>
<th></th>
<th>( A )</th>
<th>( B )</th>
<th>( C )</th>
<th>( D )</th>
<th>( E )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>–</td>
<td>27</td>
<td>12</td>
<td>23</td>
<td>74</td>
</tr>
<tr>
<td>( B )</td>
<td>27</td>
<td>–</td>
<td>47</td>
<td>15</td>
<td>71</td>
</tr>
<tr>
<td>( C )</td>
<td>12</td>
<td>47</td>
<td>–</td>
<td>28</td>
<td>87</td>
</tr>
<tr>
<td>( D )</td>
<td>23</td>
<td>15</td>
<td>28</td>
<td>–</td>
<td>75</td>
</tr>
<tr>
<td>( E )</td>
<td>74</td>
<td>71</td>
<td>87</td>
<td>75</td>
<td>–</td>
</tr>
</tbody>
</table>

1 M

\[ \begin{array}{c|ccccc}
1 & \text{} & \text{} & \text{} & \text{} & \text{} \\
\hline
\text{} & A & B & C & D & E \\
\hline
A & 27 & 12 & 23 & 74 & \\
B & 27 & – & 47 & 15 & 71 \\
C & 12 & 47 & – & 28 & 87 \\
D & 23 & 15 & 28 & – & 75 \\
E & 74 & 71 & 87 & 75 & – \\
\end{array} \]

Delete row \( A \), and number column \( A \).

The lowest undeleted entry in column \( A \) only is 12, so circle it. The first arc is \( AC \).

1 M

\[ \begin{array}{c|ccccc}
1 & \text{} & \text{} & \text{} & \text{} & \text{} \\
\hline
\text{} & A & B & C & D & E \\
\hline
A & 27 & 12 & 23 & 74 & \\
B & 27 & – & 47 & 15 & 71 \\
C & 12 & 47 & – & 28 & 87 \\
D & 23 & 15 & 28 & – & 75 \\
E & 74 & 71 & 87 & 75 & – \\
\end{array} \]

The new vertex is \( C \). Delete row \( C \), and number column \( C \).

1 M

\[ \begin{array}{c|ccccc}
1 & \text{} & \text{} & \text{} & \text{} & \text{} \\
\hline
\text{} & A & B & C & D & E \\
\hline
A & 27 & 12 & 23 & 74 & \\
B & 27 & – & 47 & 15 & 71 \\
C & 12 & 47 & – & 28 & 87 \\
D & 23 & 15 & 28 & – & 75 \\
E & 74 & 71 & 87 & 75 & – \\
\end{array} \]

The lowest undeleted entry in column \( C \) only is 28, so circle it. The second arc is \( CD \).
The new vertex is $D$. Delete row $D$, and number column $D$.

The lowest undeleted entry in column $D$ is 15, so circle it. The third arc is $DB$.

The new vertex is $B$. Delete row $B$, and number column $B$.

The lowest undeleted entry in column $B$ only is 71, so circle it. The fourth arc is $BE$.

The new vertex is $E$. Delete row $E$ and number column $E$.

All rows have now been deleted, so the algorithm is complete.

The path created is: $A \rightarrow C \rightarrow D \rightarrow B \rightarrow E$

Its weight is $12 + 28 + 15 + 71 = 126$
1. Apply the nearest neighbour algorithm to find a short path connecting all vertices for the following matrices:

   a. Start at A
   
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>15</td>
<td>20</td>
<td>34</td>
<td>25</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>15</td>
<td></td>
<td>36</td>
<td>38</td>
<td>28</td>
<td>14</td>
</tr>
<tr>
<td>C</td>
<td>20</td>
<td>36</td>
<td></td>
<td>43</td>
<td>38</td>
<td>22</td>
</tr>
<tr>
<td>D</td>
<td>34</td>
<td>38</td>
<td>43</td>
<td></td>
<td>26</td>
<td>40</td>
</tr>
<tr>
<td>E</td>
<td>25</td>
<td>28</td>
<td>38</td>
<td>26</td>
<td></td>
<td>31</td>
</tr>
<tr>
<td>F</td>
<td>9</td>
<td>14</td>
<td>22</td>
<td>40</td>
<td>31</td>
<td></td>
</tr>
</tbody>
</table>

   b. Start at S
   
<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>S</th>
<th>T</th>
<th>U</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td></td>
<td>28</td>
<td>30</td>
<td>31</td>
<td>41</td>
</tr>
<tr>
<td>S</td>
<td>28</td>
<td></td>
<td>16</td>
<td>19</td>
<td>43</td>
</tr>
<tr>
<td>T</td>
<td>30</td>
<td>16</td>
<td></td>
<td>22</td>
<td>41</td>
</tr>
<tr>
<td>U</td>
<td>31</td>
<td>19</td>
<td>22</td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>V</td>
<td>41</td>
<td>43</td>
<td>41</td>
<td>37</td>
<td></td>
</tr>
</tbody>
</table>

2. After applying the nearest neighbour algorithm, adding an arc back to the start vertex will create a route. By applying the nearest neighbour algorithm, starting at each vertex in turn, find a shortest route and state its length.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>20</td>
<td>30</td>
<td>32</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>B</td>
<td>20</td>
<td></td>
<td>10</td>
<td>25</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>C</td>
<td>30</td>
<td>10</td>
<td></td>
<td>15</td>
<td>35</td>
<td>19</td>
</tr>
<tr>
<td>D</td>
<td>32</td>
<td>25</td>
<td>15</td>
<td></td>
<td>20</td>
<td>34</td>
</tr>
<tr>
<td>E</td>
<td>12</td>
<td>32</td>
<td>35</td>
<td>20</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>F</td>
<td>15</td>
<td>16</td>
<td>19</td>
<td>34</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

3. By applying the nearest neighbour algorithm, starting at each vertex in turn, find a shortest route and state its length.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>153</td>
<td>98</td>
<td>124</td>
<td>115</td>
</tr>
<tr>
<td>B</td>
<td>153</td>
<td></td>
<td>74</td>
<td>131</td>
<td>149</td>
</tr>
<tr>
<td>C</td>
<td>98</td>
<td>74</td>
<td></td>
<td>82</td>
<td>103</td>
</tr>
<tr>
<td>D</td>
<td>124</td>
<td>131</td>
<td>82</td>
<td></td>
<td>134</td>
</tr>
<tr>
<td>E</td>
<td>115</td>
<td>149</td>
<td>103</td>
<td>134</td>
<td></td>
</tr>
</tbody>
</table>
3.5 Using Dijkstra’s algorithm to find the shortest path

You can use Dijkstra’s algorithm to find the shortest path between a start vertex and any other in a network. This could be useful for finding the cheapest, shortest or quickest transportation route between two locations.

- Dijkstra’s algorithm can be used to find the shortest path from $S$ to $T$ through a network:

1. Label the start vertex, $S$, with the final label, 0.
2. Record a working value at every vertex, $Y$, that is directly connected to the vertex, that has just received its final label, $X$.
   - Working value at $Y = \text{final label at } X + \text{weight of arc } XY$
   - If there is already a working value at $Y$, it is only replaced if the new value is smaller.
   - Once a vertex has a final label, it is not revisited and its working values are no longer considered.
3. Look at the working values at all vertices without final labels. Select the smallest working value. This now becomes the final label at that vertex. (If two vertices have the same smallest working value, either may be given its final label first.)
4. Repeat steps 2 and 3 until the destination vertex, $T$, receives its final label.
5. To find the shortest path, trace back from $T$ to $S$. If $B$ already lies on the route, include arc $AB$ whenever:
   \[\text{final label of } B - \text{final label of } A = \text{weight of arc } AB\]

Notation
- Dijkstra is pronounced ‘Dike-Stra’.

The algorithm makes use of labels. Start at the initial vertex and move through the network, putting working values on each vertex. Each pass finds the shortest route to one of the vertices and records its final label (also called its permanent label). Once a vertex has its final label it is ‘fixed’ and its working values are no longer considered. Continue in this way until the destination vertex is reached.

Online
- Explore Dijkstra’s algorithm using GeoGebra.

Example 6

Use Dijkstra’s algorithm to find the shortest route from $S$ to $T$ in the network below.

To make the working clear, you replace the vertices with boxes like this:

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Order of labelling</th>
<th>Final label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Working values</td>
<td></td>
</tr>
</tbody>
</table>

Uncorrected proof, all content subject to change at publisher discretion. Not for resale, circulation or distribution in whole or in part. ©Pearson 2018
The network becomes:

Give working values to \(A\), \(B\) and \(C\) since they are directly connected to \(S\).

Look at the working values at \(A\), \(B\) and \(C\). The smallest is 2 (at \(C\)). This will become the next final label. \(C\) is now completed. It is the 2nd vertex to be completed.

Give vertex \(S\) a final label of 0. Indicate that this is the first vertex to receive its final label by completing the top boxes at \(S\) like this:

\[
\begin{array}{c|c}
S & 0 \\
\hline
1 & 0
\end{array}
\]

Add working values to each vertex that is directly connected to \(C\). Note that the algorithm has picked up the shortcut to \(B\). \(S \rightarrow B\) has weight 6, but \(S \rightarrow C \rightarrow B\) has total weight 4.
The smallest working value is 4 at B. This becomes the final label at B. B becomes the 3rd completed vertex. Add working values to D and T because these are directly connected to B.

The only working value to add would be a 9 at D. However this is larger than the 8 already there, so it does not need to be recorded.

The smallest working value is 8 at D. This becomes D’s final label. D becomes the 5th vertex to be completed. The only working value to add is the 11 at T.

The final diagram looks like this.

**Hint** You do not need to draw lots of diagrams to show your working. Working through just one diagram and completing all the boxes will make the method clear.
You can use Dijkstra's algorithm to find the shortest route between the start vertex and any other vertex with a final label.

**Example 7**

Use Dijkstra’s algorithm in this network to find the length of the shortest route:

a. from $A$ to $H$

b. from $A$ to $G$.

List the routes you use.

The length of the shortest route from $S$ to $T$ is 11.

To find the shortest route, start at $T$ and trace back looking at final values and arc lengths.

Check the arcs into $T$:

- $11 - 8 = 3$ so $CT$ is **not** on the route.
- $11 - 4 = 8$ so $BT$ is **not** on the route.
- $11 - 8 = 3$ so $DT$ is on the route.

To get to $T$ in 11 you must have come from $D$.

Continue working back from $D$.

The working is:

- $11 - 8 = 3$ $DT$
- $8 - 4 = 4$ $BD$
- $4 - 2 = 2$ $CB$
- $2 - 0 = 2$ $SC$

So the shortest route is $SCBDT$, length 11.

**Watch out**

Unless you are asked to explain how you obtained your final route, you do not need to show all of this working. You can identify the shortest route from your fully labelled diagram by inspection.
It is possible to use Dijkstra’s algorithm on networks with directed arcs. This is like trying to find a driving route where some of the roads are one-way streets.

**Example 8**

The network below represents part of a road system in a city. Some roads are one-way and these are indicated by directed arcs. The number on each arc represents the time, in minutes, to travel along that arc.

**a** Show that there are two quickest routes from \( A \) to \( I \). Explain how you found your routes from your labelled diagram. Road \( HI \) is closed due to roadworks.

**b** Find the quickest route from \( A \) to \( I \), avoiding \( HI \).

---

**The final diagram looks like this:**

- The length of the shortest route from \( A \) to \( H \) is 29.
  - There are two shortest routes: \( ABDEH \) and \( ACFH \).
- From the diagram, the length of the shortest route from \( A \) to \( G \) is 24.
  - The route is \( ABDG \).

**Problem-solving**

- \( G \) already has a final value, so you do not need to implement the algorithm again. Just work backwards from \( G \) to \( A \).

**Example-solving**

- Examiners check the order in which the numbers appear in the list of working values. It is important to put them in the order given by the algorithm.

**Problem-solving**

- Notice that 5 does not appear as a working value for vertex \( E \) since it is not possible to travel directly from \( A \) to \( E \).
The two quickest routes are

- **A C D F G H I**
  - since
  - 19 – 17 = 2 \( HI \)
  - 17 – 15 = 2 \( GH \)
  - 15 – 14 = 1 \( FG \)
  - 14 – 6 = 8 \( EF \)
  - 6 – 5 = 1 \( DE \)
  - 5 – 2 = 3 \( CD \)
  - 2 – 0 = 2 \( AC \)

- **ACDEFGHI**
  - since
  - 19 – 17 = 2 \( HI \)
  - 17 – 15 = 2 \( GH \)
  - 15 – 14 = 1 \( FG \)
  - 14 – 6 = 8 \( EF \)
  - 6 – 5 = 1 \( DE \)
  - 5 – 2 = 3 \( CD \)
  - 2 – 0 = 2 \( AC \)

Both are of length 19 minutes.

**b** Removing \( HI \) from the network would leave a final value of 20 at \( I \).

Start at \( I \) and find the route of length 20.

- 20 – 6 = 14 \( EI \)
- 6 – 5 = 1 \( DE \)
- 5 – 2 = 3 \( CD \)
- 2 – 0 = 2 \( AC \)

So the quickest route from \( A \) to \( I \), avoiding \( HI \), is \( ACDEI \), of length 20 minutes.

---

### Exercise 3E

1. Use Dijkstra’s algorithm to find the shortest route from \( S \) to \( T \) in each of the following networks. In each case, explain how you determined the shortest path from your fully labelled diagram.

   **a**
   ![Network A](image1)

   **b**
   ![Network B](image2)

2. The network shows part of a road system in a city. The number on each arc gives the time, in minutes, it takes to travel along that arc.

   Use Dijkstra’s algorithm to find:
   - **a** the quickest route from \( A \) to \( Q \) and its length
   - **b** the quickest route from \( A \) to \( L \) and its length
   - **c** the quickest route from \( M \) to \( A \) and its length
   - **d** the quickest route from \( P \) to \( A \) and its length.
3 Use Dijkstra’s algorithm to find the shortest route, and its length, from A to F in this directed network.

4 The network represents the lengths, in metres, of all the roads in a building site. A bulldozer is needed for one day at T. There are two bulldozers available on-site, one at S1, and the other at S2. One of the two bulldozers will be moved to T. In order to minimise the cost it is decided to move the bulldozer that is closest to T. Use Dijkstra’s algorithm to determine which bulldozer should be moved.

(7 marks)

Problem-solving
It is possible to solve this problem with only one application of Dijkstra’s algorithm. Think carefully about the starting point.

5 a Use Dijkstra’s algorithm to find the shortest route from A to H. Indicate how you obtained your shortest route from your labelled diagram.

(6 marks)

b Find the shortest route from A to H via G.

(2 marks)

c Find the shortest route from A to H, not using CE.

(2 marks)

6 Use Dijkstra’s algorithm to find the shortest route from S to T in this directed network. State the length of your route.

(6 marks)
A navigation app uses real-time data to determine driving times between different locations. The diagram shows a network of driving times, in minutes, along roads joining ten locations.

a Use Dijkstra’s algorithm to find the quickest route from S to T, and state the total driving time for this route. (6 marks)

A driver is following this quickest route. Due to traffic congestion, the travel time between H and T increases from 5 minutes to 13 minutes.

b State how this will affect the quickest route if this information is discovered:
   i before the start of the journey
   ii when the driver reaches point H. (4 marks)

1 The network represents a theme park with seven zones. The number on each arc shows a distance in metres.

Tramways are to be built to link the seven zones and the entrance.

a Find a minimum spanning tree using:
   i Kruskal’s algorithm (4 marks)
   ii Prim’s algorithm, starting at the entrance.

   You must make your order of arc selection clear.

b Draw your tree and state its weight. (2 marks)

2 The network represents eight observation points in a wildlife reserve and the possible paths connecting them. The number on each arc is the distance, in kilometres, along that path. It is decided to link the observation points by paths but, in order to minimise the impact on the wildlife reserve, the least total length of path is to be used.

a Find a minimum spanning tree for the network using:
   i Prim’s algorithm, starting at L (4 marks)
   ii Kruskal’s algorithm.

   In each case, list the arcs in the order in which you consider them.
b Given that paths TQ and RP already exist, and so must form part of the tree, state which algorithm, Prim’s or Kruskal’s, you would select to complete the spanning tree. Give a reason for your answer.  

3 The table shows the distances, in mm, between six nodes A to F in a network. 

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>–</td>
<td>124</td>
<td>52</td>
<td>87</td>
<td>58</td>
<td>97</td>
</tr>
<tr>
<td>B</td>
<td>124</td>
<td>–</td>
<td>114</td>
<td>111</td>
<td>115</td>
<td>84</td>
</tr>
<tr>
<td>C</td>
<td>52</td>
<td>114</td>
<td>–</td>
<td>67</td>
<td>103</td>
<td>98</td>
</tr>
<tr>
<td>D</td>
<td>87</td>
<td>111</td>
<td>67</td>
<td>–</td>
<td>41</td>
<td>117</td>
</tr>
<tr>
<td>E</td>
<td>58</td>
<td>115</td>
<td>103</td>
<td>41</td>
<td>–</td>
<td>121</td>
</tr>
<tr>
<td>F</td>
<td>97</td>
<td>84</td>
<td>98</td>
<td>117</td>
<td>121</td>
<td>–</td>
</tr>
</tbody>
</table>

a Use Prim’s algorithm, starting at A, to find a minimum spanning tree for this table of distances. You must explain your method carefully and indicate clearly the order in which you selected the arcs.  

b Draw a sketch showing the minimum spanning tree and find its length.  

4 It is intended to network five computers at a large race track. There is one computer at the office and one at each of the four different entrances. Cables need to be laid to link the computers. Cable laying is expensive, so a minimum total length of cable is required. 

The table shows the shortest distances, in metres, between the various sites. 

<table>
<thead>
<tr>
<th>Office</th>
<th>Entrance 1</th>
<th>Entrance 2</th>
<th>Entrance 3</th>
<th>Entrance 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office</td>
<td>–</td>
<td>1514</td>
<td>488</td>
<td>980</td>
</tr>
<tr>
<td>Entrance 1</td>
<td>1514</td>
<td>–</td>
<td>1724</td>
<td>2446</td>
</tr>
<tr>
<td>Entrance 2</td>
<td>488</td>
<td>1724</td>
<td>–</td>
<td>884</td>
</tr>
<tr>
<td>Entrance 3</td>
<td>980</td>
<td>2446</td>
<td>884</td>
<td>–</td>
</tr>
<tr>
<td>Entrance 4</td>
<td>945</td>
<td>2125</td>
<td>587</td>
<td>523</td>
</tr>
</tbody>
</table>

a Starting at Entrance 2, demonstrate the use of Prim’s algorithm and hence find a minimum spanning tree. You must make your method clear, indicating the order in which you selected the arcs in your final tree.  

b Calculate the minimum total length of cable required.  

5 The diagram above shows a weighted network. The weights of each arc are given in the following table: 

<table>
<thead>
<tr>
<th>AB</th>
<th>AE</th>
<th>AH</th>
<th>BC</th>
<th>CD</th>
<th>CF</th>
<th>CG</th>
<th>DG</th>
<th>EF</th>
<th>EJ</th>
<th>EK</th>
<th>FI</th>
<th>FJ</th>
<th>GI</th>
<th>HK</th>
<th>JK</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>27</td>
<td>30</td>
<td>19</td>
<td>9</td>
<td>26</td>
<td>17</td>
<td>18</td>
<td>8</td>
<td>12</td>
<td>24</td>
<td>20</td>
<td>14</td>
<td>14</td>
<td>22</td>
<td>26</td>
</tr>
</tbody>
</table>

a Use a quick sort to write these arcs in order of weight, smallest first.  

(2 marks)

(3 marks)

(3 marks)

(4 marks)

(4 marks)
b Use Kruskal's algorithm to find a minimum spanning tree for this network. (3 marks)
c Draw your minimum spanning tree and state its total weight. (2 marks)

For any connected network,
\[ e = \text{number of edges in the minimum spanning tree} \]
\[ v = \text{number of vertices in the network} \]
d Write down the relationship between \( e \) and \( v \). (2 marks)

6 A company is to install power lines to buildings on a large industrial estate. The lines are to be laid by the side of the roads on the estate. The estate is shown here as a network. The buildings are designated \( A, B, C, \ldots, N \) and the distances between them are given in hundreds of metres. The manager wants to minimise the total length of power line to be used.

a Use Kruskal's algorithm to obtain a minimum spanning tree for the network and hence determine the minimum length of power line needed. (4 marks)

Owing to a change of circumstances, the company modifies its plans for the estate. The result is that the road from \( F \) to \( G \) now has a length of 700 metres.

b Determine the new minimum total length of power line. (3 marks)

7 A weighted network is shown. The number on each arc indicates the weight of that arc.

a Use Dijkstra's algorithm to find a path of least weight from \( A \) to \( K \).

State clearly:
\[ \text{i} \quad \text{the order in which the vertices were labelled} \]
\[ \text{ii} \quad \text{how you determined the path of least weight from your labelling} \]

b List all alternative paths of least weight.

c Describe a practical problem that could be modelled by the network and solved using Dijkstra's algorithm.
8 The network here shows the distances, in kilometres, between nine cities. Use Dijkstra’s algorithm to determine the shortest route, and its length, between cities \( S \) and \( T \). You must indicate clearly the order in which the vertices are labelled and how you used your labelled diagram to decide which cities to include in the shortest route. (5 marks)

9 a Find a minimum spanning tree for this network using:
   i Prim’s algorithm
   ii Kruskal’s algorithm.
   b Compare the application of the two algorithms in this case. State which is easier.
   c Describe a condition under which Prim’s algorithm may be quicker to apply than Kruskal’s algorithm.

10 The table below shows the flight distances, in kilometres, between eight cities: Alice Springs (\( A \)), Brisbane (\( B \)), Cairns (\( C \)), Darwin (\( D \)), Eden (\( E \)), Fremantle (\( F \)), Gladstone (\( G \)) and Hobart (\( H \)). Both Prim’s algorithm and the nearest neighbour algorithm connect all of the cities together.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>–</td>
<td>1970</td>
<td>1450</td>
<td>1290</td>
<td>2130</td>
<td>2010</td>
<td>1770</td>
<td>2470</td>
</tr>
<tr>
<td>( B )</td>
<td>1970</td>
<td>–</td>
<td>1390</td>
<td>2850</td>
<td>1110</td>
<td>3620</td>
<td>440</td>
<td>1790</td>
</tr>
<tr>
<td>( C )</td>
<td>1450</td>
<td>1390</td>
<td>–</td>
<td>1680</td>
<td>2280</td>
<td>3460</td>
<td>960</td>
<td>2890</td>
</tr>
<tr>
<td>( D )</td>
<td>1290</td>
<td>2850</td>
<td>1680</td>
<td>–</td>
<td>3330</td>
<td>2670</td>
<td>2500</td>
<td>3740</td>
</tr>
<tr>
<td>( E )</td>
<td>2130</td>
<td>1110</td>
<td>2280</td>
<td>3330</td>
<td>–</td>
<td>3160</td>
<td>1480</td>
<td>680</td>
</tr>
<tr>
<td>( F )</td>
<td>2010</td>
<td>3620</td>
<td>3460</td>
<td>2670</td>
<td>3160</td>
<td>–</td>
<td>3590</td>
<td>3020</td>
</tr>
<tr>
<td>( G )</td>
<td>1770</td>
<td>440</td>
<td>960</td>
<td>2500</td>
<td>1480</td>
<td>3590</td>
<td>–</td>
<td>2150</td>
</tr>
<tr>
<td>( H )</td>
<td>2470</td>
<td>1790</td>
<td>2890</td>
<td>3740</td>
<td>680</td>
<td>3020</td>
<td>2150</td>
<td>–</td>
</tr>
</tbody>
</table>

a Starting at \( A \), use Prim’s algorithm to find a minimum spanning tree and state the weight of the minimum spanning tree.

b Starting at \( C \), use the nearest neighbour algorithm to find a short path and state its weight.

A news reporter wishes to visit all eight cities, starting from \( C \). One way of visiting all eight cities is to create a minimum spanning tree, then starting at \( C \), traverse each arc twice. Another way of visiting all eight cities is to use the nearest neighbour algorithm and, once completed, to return directly to \( C \).

c Using your answers to part a and b which method would give the shortest way of visiting all eight cities.
A minimum spanning tree (MST) is a spanning tree such that the total length of its arcs (edges) is as small as possible.

Kruskal’s algorithm can be used to find a minimum spanning tree.

• Sort the arcs into ascending order of weight and use the arc of least weight to start the tree. Then add arcs in order of ascending weight, unless an arc would form a cycle, in which case reject it.

Prim’s algorithm can be used to find a minimum spanning tree.

• Choose any vertex to start the tree. Then select an arc of least weight that joins a vertex already in the tree to a vertex not yet in the tree. Repeat this until all vertices are connected.

Prim's algorithm can be applied to a distance matrix.

• Choose any vertex to start the tree. Delete the row in the matrix for the chosen vertex and number the column in the matrix for the chosen vertex. Circle the lowest undeleted entry in the numbered columns, which becomes the next arc. Repeat this until all rows are deleted.

Dijkstra’s algorithm can be used to find the shortest path between two vertices in a network.

• Label the start vertex with final value 0.
• Record a working value at every vertex, \( Y \), that is directly connected to the vertex that has just received its final label, \( X \):
  
  \[
  \text{final label at } X + \text{weight of arc } XY
  \]
• Select the smallest working value. This is the final label for that vertex.
• Repeat until the destination vertex receives its final label.
• Find the shortest path by tracing back from destination to start. Include an arc \( AB \) on the route if \( B \) is already on the route and
  
  \[
  \text{final label } B - \text{final label } A = \text{weight of arc } AB
  \]

Each vertex is replaced by a box:

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Order of labelling</th>
<th>Final label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Dijkstra’s algorithm finds the shortest route between the start vertex and each intermediate vertex completed on the way to the destination vertex.

It is possible to use Dijkstra’s algorithm on networks with directed arcs, such as a route with one-way streets.
GLOSSARY

activity network (or arc network) activities represented by arcs, the completion of which, known as events, are shown as nodes.

adjacency matrix each entry in an adjacency matrix describes the number of arcs joining the corresponding vertices.

algorithm a finite (i.e. limited) sequence of step-by-step instructions carried out to solve a problem

arc a line segment connecting two nodes, also called an edge.

backward pass in critical path analysis, this is calculating all of the late event times.

bin-packing, first-fit an algorithm where each object is fitted into the first available bin.

bin-packing, first-fit decreasing an algorithm where objects from a decreasing ordered list is fitted into the first available bin.

bin-packing, full an algorithm where full bin combinations are found first.

bound a value which a set of numbers cannot go above (upper bound) or go below (lower bound).

bubble sort an algorithm that orders a list of objects.

classical travelling salesman problem a problem in which each vertex must be visited exactly once in the shortest possible route.

complete graph a graph in which each of the \( n \) vertices is connected to every other vertex.

connected two vertices are connected if there is a path between them. A graph is connected if all its vertices are linked in this way.

constraints things that will prevent you making or using an infinite (i.e. unlimited) number of each of the variables. Each constraint will give rise to one inequality.

corresponding an equivalent; connected with what you have just mentioned.

critical activity used to describe an activity if any increase in its duration results in a corresponding increase in the duration of the whole project.

critical path a path from the source node to the sink node made up of critical activities.

cycle (or circuit) a closed path, i.e. where the end vertex of the last edge is the start vertex of the first edge.

decision variables (in a linear programming problem) the numbers of each of the things that can be varied. The variables, which are often called \( x, y, z, \) etc., will be the ‘letters’ in the inequalities and objective function.

degree (or valency; of a vertex) the number of edges incident to the vertex, i.e. sharing a vertex.

dependency when one activity cannot proceed until another activity is completed.

digraph if the edges of a graph have a direction associated with them, they are known as ‘directed edges’ and the graph is known as a digraph.

Dijkstra's algorithm an algorithm to find the shortest path from a vertex to the source vertex.

distance matrix the entries in a distance matrix represent the weight of each arc, not the number of arcs.

dummy activity used to show dependencies between other activities.

duration the length of time taken or needed to complete an activity.

distance matrix the entries in a distance matrix represent the weight of each arc, not the number of arcs.

dummy activity used to show dependencies between other activities.

duration the length of time taken or needed to complete an activity.

early event time the earliest time of arrival at the event allowing for the completion of all preceding activities.

edge a line segment connecting two nodes, also called an edge.

edge set the set of all edges in a graph.

Eulerian cycle (or Eulerian circuit) a trail which visits every arc, starting and ending at the same vertex.

Eulerian graph a graph or network that contains a trail that includes every edge and starts and finishes at the same vertex.

event the start of finish of an activity.

feasible region the region of a graph that satisfies all the constraints of a linear programming problem.

feasible solution when you have found values for the decision variables that satisfy each constraint.

finite a value that is limited, not infinite.

flow chart a type of diagram that represents an algorithm by showing the steps in boxes, and their order by connecting the boxes with directional arrows.

formulate to develop a set of rules, expressing your ideas carefully.
**forward pass** calculating all of the early event time in critical path analysis.

**Gantt (cascade) chart** a graphic representation of the range of possible start and finish times for all the activities on a single diagram.

**graph** a graph $G$ consists of points (vertices or nodes) which are connected by lines (edges or arcs).

**Hamiltonian cycle** a cycle that includes every vertex.

**incident** two objects that touch each other are incident.

**infinite** a value that is unlimited, not finite.

**inspection** to use your knowledge in mathematics to know the solution just by looking at it.

**integer** a whole number. The symbol for an integer is $z$.

**isomorphic graph** a graph which shows the same information as a complete graph but may be drawn differently.

**Kruskal’s algorithm** an algorithm to find a minimum spanning tree.

**late event time** the latest time that an event can be left until without extending the time needed for the project.

**lemma** a small theorem used as a stepping stone to more important results.

**linear programming** a method of solving problems in involving inequalities and more than one variable.

**loop** an edge that starts and finishes at the same vertex.

**matrix** a rectangular array of numbers or other mathematical functions for which operations such as addition and multiplication are defined.

**maximise** to increase something as much as possible.

**minimise** to reduce something to the smallest possible amount.

**minimum connector** see minimum spanning tree.

**minimum spanning tree (also minimum connector)** a spanning tree such that the total length of its arcs is as small as possible.

**nearest neighbour algorithm** an algorithm used to find a short (but not necessarily the minimum) path in a network.

**node (also vertex)** a point on a graph.

**non-Eulerian graph** a connected graph where there is more than one pair of odd nodes. It is a graph which is neither Eulerian or semi-Eulerian.

**optimal solution** a feasible solution that meets the objective. There might be more than one optimal solution.

**pass** a single application of an algorithm, such as the bubble sort, where there is a repetition of the steps.

**path** a finite sequence of edges, such that the end vertex of one edge in the sequence is the start vertex of the next, and in which no vertex appears more than once.

**pivot** an object in a list or matrix which is selected by an algorithm.

**practical travelling salesman problem** a problem in which each vertex must be visited at least once before returning to the start.

**precede** to happen or exist before something else.

**precedence table (or dependence table)** a table which shows which activities must be completed before others are started.

**Prim’s algorithm** an algorithm to find a minimum spanning tree.

**quick sort** an algorithm that orders a list of objects.

**residual minimum spanning tree** found when a vertex is removed from the lower bound in the travelling salesman problem.

**route inspection algorithm** an algorithm to find the shortest route in a network.

**satisfy** when having many inequalities or conditions, and they are all true, we say that we have satisfied them.

**scheduling** a connected graph where there is one pair of odd vertices.

**semi-Eulerian graph** a connected graph where there is one pair of odd vertices.

**shortcuts** used to reduce the upper bound in the travelling salesman problem.

**simple graph** a graph in which there are no loops and there is no more than one edge connecting a pair of vertices.

**sink node** in critical path analysis, the final node.

**source node** in critical path analysis, the first node.

**spanning tree (of a graph $G$)** a subgraph which includes all the vertices of $G$ and is also a tree.

**subgraph (of $G$)** a graph, each of whose vertices belongs to $G$ and each of whose edges belongs to $G$.

**table of least distances** a table showing the shortest path between any two points in the network.
**total float (of an activity)** the amount of time that its start may be delayed without affecting the duration of the project.

**tour** a walk which visits every vertex, returning to its starting vertex.

**trace table** used to record the values of each variable as an algorithm is run.

**trail** a walk in which no edge is visited more than once.

**travelling salesman problem (also see classical and practical)** a problem where one must find a route minimum length which visits every vertex in an undirected network.

**traverse** to travel down an arc.

**tree** a connected graph with no cycles.

**triangle inequality** for three vertices $A$, $B$ and $C$, the triangular inequality is ‘length $AB < length AC + length CB$’, where $AB$ is a longest length.

**valency (or degree; of a vertex)** the number of edges incident to the vertex, i.e. sharing a vertex.

**vertex (also node)** a point on a graph.

**vertex set** the list of all vertices in a graph.

**walk (in a network)** a finite (limited) sequence of edges such that the end vertex of one edge is the start vertex of the next.

**weight** the sum of the arcs in a graph.

**weighted graph** a graph which has a number associated with each edge (usually called its weight).
CHAPTER 1

Prior knowledge check
1 a 47 b 24
2 \(x_2 = 3, x_3 = 2.65, x_4 = 2.58\)

Exercise 1A

| Step | A  | r  | C  | \(|r - C|\) | s  | Print r |
|------|----|----|----|-------------|----|---------|
| 1    | 79 | 10 |    |            |    |         |
| 2    | 7.900 |     |    |            |    |         |
| 3    | 2.1 |     |    |            |    |         |
| 4    | 8.950 |     |    |            |    |         |
| 5    | 8.889 |     |    |            |    |         |
| 6    | 8.887 |     |    |            |    |         |
| 3    | 0.002 |     |    |            |    |         |
| 7    | 8.889 |     |    |            |    |         |
| 2    | 8.827 |     |    |            |    |         |
| 3    | 0.123 |     |    |            |    |         |
| 4    | 8.889 |     |    |            |    |         |
| 5    | 8.889 |     |    |            |    |         |
| 6    | 8.887 |     |    |            |    |         |
| 3    | 0.002 |     |    |            |    |         |

**b** It divides the first fraction by the second fraction.

| Step | A  | r  | C  | \(|r - C|\) | s  | Print r |
|------|----|----|----|-------------|----|---------|
| 1    | 4275 | 50 |    |            |    |         |
| 2    | 85.500 |     |    |            |    |         |
| 3    | 35.5 |     |    |            |    |         |
| 4    | 67.75 |     |    |            |    |         |
| 5    | 67.75 |     |    |            |    |         |
| 6    | 63.100 |     |    |            |    |         |
| 3    | 4.65 |     |    |            |    |         |
| 4    | 65.425 |     |    |            |    |         |
| 5    | 65.425 |     |    |            |    |         |
| 6    | 65.342 |     |    |            |    |         |
| 3    | 0.083 |     |    |            |    |         |
| 4    | 65.384 |     |    |            |    |         |
| 5    | 65.384 |     |    |            |    |         |
| 6    | 65.383 |     |    |            |    |         |
| 3    | 0.001 |     |    |            |    |         |
| 7    | 65.384 |     |    |            |    |         |

**iii**

| Step | A  | r  | C  | \(|r - C|\) | s  | Print r |
|------|----|----|----|-------------|----|---------|
| 1    | 79 | 10 |    |            |    |         |
| 2    | 7.900 |     |    |            |    |         |
| 3    | 2.1 |     |    |            |    |         |
| 4    | 8.950 |     |    |            |    |         |
| 5    | 8.889 |     |    |            |    |         |
| 6    | 8.887 |     |    |            |    |         |
| 3    | 0.002 |     |    |            |    |         |
| 7    | 8.889 |     |    |            |    |         |
| 2    | 8.827 |     |    |            |    |         |
| 3    | 0.123 |     |    |            |    |         |
| 4    | 8.889 |     |    |            |    |         |
| 5    | 8.889 |     |    |            |    |         |
| 6    | 8.887 |     |    |            |    |         |
| 3    | 0.002 |     |    |            |    |         |
| 7    | 8.889 |     |    |            |    |         |

Exercise 1B

<table>
<thead>
<tr>
<th>Step</th>
<th>A</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>d &lt; 0?</th>
<th>d = 0?</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>-12</td>
<td>9</td>
<td>0</td>
<td>No</td>
<td>Yes</td>
<td>1.5</td>
</tr>
</tbody>
</table>

**Equal roots are 1.5**

<table>
<thead>
<tr>
<th>Step</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>d &lt; 0?</th>
<th>d = 0?</th>
<th>x1</th>
<th>x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-6</td>
<td>13</td>
<td>5</td>
<td>289</td>
<td>No</td>
<td>No</td>
<td>-(\frac{1}{3})</td>
<td>(\frac{5}{2})</td>
</tr>
</tbody>
</table>

**b** Roots are \(-\frac{1}{3}\) and \(\frac{5}{2}\)
Exercise 1C

1. **Exercise 1C**

   **i**  
   a 16 15 23 18 25 11 19 34  
   b 15 16 18 23 11 19 25 34  
   c 15 16 18 11 19 23 25 34  
   d 11 15 16 18 19 23 25 34

   **ii**  
   a E N T O R K S W  
   b E N O R K S T W

   **iii**  
   a A5 D2 A1 B4 C7 C2 B3 D3  
   b D3 D2 A5 B4 C7 C2 B3 A1

2. **Exercise 1D**

   **i**  
   a 2 3 4 5 6 7 8  
   b 8 7 6 5 4 3 2

   **ii**  
   a Myles 19  
   b Sam 29

3. **Challenge**

   **a** Answers will vary.  
   **b** Put the Ace of Hearts at the end.
Exercise 1E
1 a 5 bins
   b i Bin 1: 18 + 4 + 23 + 3
       Bin 2: 8 + 27
       Bin 3: 19 + 26
       Bin 4: 30
       Bin 5: 35
       Bin 6: 32
   ii Bin 1: 35 + 8 + 4 + 3
       Bin 2: 32 + 18
       Bin 3: 30 + 19
       Bin 4: 27 + 23
       Bin 5: 26
   iii Bin 1: 32 + 18
       Bin 2: 27 + 23
       Bin 3: 35 + 8 + 4 + 3
       Bin 4: 19 + 26
       Bin 5: 30
2 a Bin 1: A(30) + B(30) + C(30) + D(45) + E(45)
       Bin 2: F(60) + G(60) + H(60)
       Bin 3: I(60) + J(75)
       Bin 4: K(90)
       Bin 5: L(120)
       Bin 6: M(120)
   b Bin 1: M(120) + I(60)
       Bin 2: L(120) + H(60)
       Bin 3: K(90) + J(75)
       Bin 4: G(60) + F(60) + E(45)
       Bin 5: D(45) + C(30) + B(30) + A(30)
6 a Bin 1: H(25) + A(8)
       Bin 2: G(25)
       Bin 3: F(24) + B(16)
       Bin 4: E(22) + C(17)
       Bin 5: D(21)
   b Lower bound is 4.
   c There are 5 programs over 20 GB. It is not possible
      for any two of these to share a bin. So at least 5 bins
      will be needed, so 4 will be insufficient.

Exercise 1F
1 a Connock is in the list
   b Walkey is in the list
   c Peabody is not in the list
2 a 21 is in the list
   b 5 is not in the list
3 a 7
   b 10
   c 14
b Bin 1: 100
Bin 2: 92
Bin 3: 84 + 30
Bin 4: 75 + 42
Bin 5: 60 + 52
unused space on DVD: 65 minutes

c There is room on DVD 2 for one of the 25-minute programmes but no room on any DVD for the second programme.

6 a The two 1.2 m lengths cannot be ‘made up’ to 2 m bins since there are only 3 × 0.4 m lengths. Two of these can be used to make a full bin, 1.2 + 0.4 + 0.4, but the second 1.2 m cannot be made up to 2 m since there is only one remaining 0.4 m length.

b Bin 1: 1.6 + 0.6
Bin 2: 1.4 + 1
Bin 3: 1.2 + 1.2
Bin 4: 1 + 1 + 0.4
Bin 5: 0.6 + 0.6 + 0.6 + 0.6
Bin 6: 0.4

c Bin 1: 1.6 + 0.4 + 0.4
Bin 2: 1.4 + 1
Bin 3: 1.2 + 1.2
Bin 4: 1 + 1 + 0.4
Bin 5: 0.6 + 0.6 + 0.6 + 0.6

7 a Output 4.8
b It selects the number furthest from 5.

c It would select the number nearest to 5.

8 a 1st pass: 2.0 1.3 1.6 0.3 1.3 0.3 0.2 2.0 0.5 0.1
2nd pass: 2.0 1.6 1.3 1.3 0.3 0.3 0.2 2.0 0.5 0.2 0.1
3rd pass: 2.0 1.6 1.3 1.3 0.3 0.3 0.2 2.0 0.5 0.3 0.2 0.1
4th pass: 2.0 1.6 1.3 1.3 2.0 0.5 0.3 0.3 0.2 0.1
5th pass: 2.0 1.6 1.3 2.0 1.3 0.5 0.3 0.3 0.2 0.1
6th pass: 2.0 1.6 2.0 1.3 1.3 0.5 0.3 0.3 0.2 0.1
7th pass: 2.0 2.0 1.6 1.3 1.3 0.5 0.3 0.3 0.2 0.1
8th pass: 2.0 2.0 1.6 1.3 1.3 0.5 0.3 0.3 0.2 0.1
b Bin 1: 2.0; Bin 2: 2.0; Bin 3: 1.6 + 0.3 + 0.1
Bin 4: 1.3 + 2.0 + 0.2; Bin 5: 1.3 + 0.3
5 lengths of pipe needed

c Yes. Total length required is 9.6 m, so lower bound is 4.8, rounded up to 5 lengths of pipe.

9 M V C A D B K S
C A B D M V K S
A B C D K M V S
A B C D K S V

Challenge
a The names are not in ascending alphabetical order and so a binary search cannot be done.

b J M C B T H K R G F
J C B M H K R G F T
C B J H K M G F R T
B C H J K G F M R T
B C H J G F K M R T
B C H G F J K M R T
B C F G H J K M R T
B C F G H J K M R T

Middle of the list is Hyo. Kim follows Hyo. Consider sublist: J K M R T
Middle of list is Merry. Kim comes before Merry. Consider sublist: J K
Middle of list is Kim. Kim found. Kim is in the list.

CHAPTER 2

Prior knowledge check
1 If \( a = b + c \), then \( ABC \) would be on a straight line, and if \( a > b + c \), then sides \( b \) and \( c \) would not have sufficient combined length to meet at a single point \( A \)

2 \( AECD = 12 \)

Exercise 2A
1 a i a student ii that a pair of students are friends
b Banjit, Dhevan, Esme
c e.g. Adok and Chris, Esme and Fabio

2 a A B C D E F

b Mathematics and art

3 a i One change. For example, Chancery lane to Notting Hill Gate on red line, then Notting Hill Gate to Victoria on the yellow or green line.
ii Through 4 stations. For example, Chancery Lane – Holborn – Tottenham Court Road – Oxford Circus – Green Park – Victoria.

b i Marylebone to Waterloo on the brown line.
ii Victoria to Green park (light blue line), green park to baker street (grey line).

3 \( 7 \times 80 = 9 \text{ min } 20 \text{ sec} \)
ii Victoria to Green park (light blue line), Green park to Baker street (grey line).

3 \( 3 \times 80 + 180 = 7 \text{ min.} \)
iii Holborn to Oxford Circus (red line), Oxford Circus to Victoria (light blue line), Victoria to St James Park (yellow or green line)

5 \( 5 \times 80 + 2 \times 180 = 12 \text{ min } 40 \text{ sec} \)

4 a 40 \text{ min}
b Aberdeen/Cork
c Dublin – it is the airport with the most connections

5 a \( PTV \)
b Not correct. \( PTQRSV \) is 25 km

Challenge
a 6 routes
b 18 routes

Exercise 2B
1 a For example:
2 a is not simple.
There are two edges connecting C with D.
b and c are simple.
d is not simple. There is a loop attached to U.
3 a and c are connected.
b is not connected, there is no path from J to G, f or example.
d is not connected, there is no path from W to Z, for example.
4 a Any four of these:
\[ F A B D \quad F E D \]
\[ F A C B D \quad F E C B D \]
\[ F A B C E D \quad F E C A B D \]
\[ F A C E D \]
b e.g. \[ F A C B D E, D B A F E D \]
c
<table>
<thead>
<tr>
<th>Vertex</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

Here are examples. (There are many others.)
i A

\[ B \quad C \quad E \quad D \]

\[ A \quad B \quad C \quad D \]

e Sum of degrees = \( 3 + 3 + 3 + 2 + 3 + 2 = 16 \)
number of edges = 8
sum of degrees = \( 2 \times \) number of edges for this graph
5

<table>
<thead>
<tr>
<th>Vertex</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Here are some possible subgraphs (there are many others).
i L

\[ P \quad Q \quad R \]

\[ S \quad P \quad L \]

\[ J \quad S \]

Sum of degrees = \( 1 + 2 + 3 + 1 + 1 + 4 + 2 + 1 + 1 = 16 \)
number of edges = 8
sum of degrees = \( 2 \times \) number of edges for this graph

Exercise 2C
1 a and b are trees.
c is not a tree, it is not a connected graph.
d is not a tree, it contains a cycle.
2 i

\[ L \quad P \quad Q \quad R \]

\[ S \]

\[ \]

\[ \]

\[ \]

\[ \]

3 vertices all even, all of degree 2

c The sum of degrees = \( 2 \times \) number of edges, so the sum of degrees must be even. Any vertices of odd degree must therefore ‘pair up’. So there must be an even number of vertices of odd degree.
A tree is a connected graph with no cycles.

A spanning tree is a subgraph which includes all vertices and is a tree.

The graph is not connected.

Each vertex will have a degree of \( n - 1 \)

190

Exercise 2D

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

a

b

c

d

a

b

c

d
4

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>—</td>
<td>14</td>
<td>11</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B</td>
<td>—</td>
<td>—</td>
<td>10</td>
<td>13</td>
<td>11</td>
</tr>
<tr>
<td>C</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>12</td>
<td>—</td>
</tr>
<tr>
<td>D</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>10</td>
</tr>
<tr>
<td>E</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>7</td>
</tr>
<tr>
<td>F</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>9</td>
</tr>
</tbody>
</table>

**Worked solutions are available in SolutionBank.**

4  a. A distance matrix gives the weights of edges between pairs of vertices, whereas an adjacency matrix gives the number of edges between pairs of vertices.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>—</td>
<td>20</td>
<td>18</td>
<td>16</td>
<td>—</td>
</tr>
<tr>
<td>B</td>
<td>20</td>
<td>—</td>
<td>15</td>
<td>—</td>
<td>50</td>
</tr>
<tr>
<td>C</td>
<td>18</td>
<td>15</td>
<td>—</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>D</td>
<td>16</td>
<td>—</td>
<td>10</td>
<td>—</td>
<td>23</td>
</tr>
<tr>
<td>E</td>
<td>—</td>
<td>—</td>
<td>20</td>
<td>23</td>
<td>—</td>
</tr>
<tr>
<td>F</td>
<td>—</td>
<td>50</td>
<td>30</td>
<td>—</td>
<td>25</td>
</tr>
</tbody>
</table>

b. e.g.

5  e – 1

6  PQR, PQTR, PTR, PSR

7  a. 3

b. i. e.g. BEAD  
    ii. e.g. ACEA

c. i. e.g.

ii. e.g.

8  a. The sum of the degrees of the vertices must be even. However, 3 + 1 + 2 = 7 is odd, so there is no 4-vertex graph whose vertices have these degrees.

b. 5

9  a. AGBECFDSA

b. e.g. ACEFDSA

**Challenge**

a. \( V = 7, E = 13, R = 7; 7 + 7 - 13 = 1 \)

b. \( V = 4, E = 10, R = 4; 1 + 4 - 4 = 1 \)

c. \( E = R, \) so \( V + R - E = V - 1 \)
d) \( V' = V - 1, E' = E - 1, R' = R \)

So \( V' + R' - E' = (V - 1) + R - (E - 1) = V + R - E = 1 \)

e) We will prove the following statement by induction:

The relationship \( V + R - E = 1 \) holds for all connected graphs with \( n \) vertices'. Basis step: part c shows that this statement holds for \( n = 1 \). Induction step: Assume true for any graph with \( n - 1 \) vertices. Then, given a graph \( G \) of \( n \) vertices, contract one edge to obtain a graph \( G_9 \) of \( n - 1 \) vertices. The induction hypothesis implies that \( G_9 \) satisfies the relation, and then part d implies that \( G \) also satisfies the relation.

CHAPTER 3

Prior knowledge check

1 a) 5

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>–</td>
<td>12</td>
<td>15</td>
<td>–</td>
<td>–</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>12</td>
<td>–</td>
<td>7</td>
<td>14</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>C</td>
<td>15</td>
<td>7</td>
<td>–</td>
<td>9</td>
<td>5</td>
<td>11</td>
<td>–</td>
</tr>
<tr>
<td>D</td>
<td>–</td>
<td>14</td>
<td>9</td>
<td>–</td>
<td>16</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>E</td>
<td>–</td>
<td>–</td>
<td>5</td>
<td>16</td>
<td>–</td>
<td>–</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>4</td>
<td>–</td>
<td>11</td>
<td>–</td>
<td>–</td>
<td>3</td>
<td>–</td>
</tr>
<tr>
<td>G</td>
<td>9</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>2</td>
<td>3</td>
<td>–</td>
</tr>
</tbody>
</table>

c) e.g. \( AF \) (4), \( FG \) (3), \( GE \) (2), \( EC \) (5), \( BC \) (7), \( CD \) (9).

Total weight 30.

Exercise 3A

1 a) \( EF, BD, CD, AH, DF, AC, GH \); weight 98

b) \( BF, FG, AB, CE, BC, CD \); weight 27

c) \( DE, EF, CD, EG, GJ, AB, GH, BH \); weight 22.4

2 a) i) A tree is a connected graph with no cycles.

ii) A minimum spanning tree is a tree of minimum total weight that connects all of the nodes.

b) Writing the arcs in order of their weights: \( YZ \) (15), \( VW \) (16), \( XY \) (17), \( UV \) (17), \( UX \) (18), \( WX \) (18), \( SU \) (18), \( WZ \) (18), \( UV \) (19), \( FU \) (20), \( ST \) (22), \( TV \) (23)

Underlined arcs are in the minimum spanning tree. Total weight = 121

c) The minimum spanning tree is not unique. For example, \( UX \) can be replaced with \( WX \).

3 a) The three shortest edges do not form a cycle.

b) The three shortest edges form a cycle.

4 a) Arcs were added in the order \( MCl \) (0.8), \( ShR \) (1.1), \( CIS, CeR, CeH, SR, ME, PE \)

b) 15 km
Exercise 3B

1

![Graph](image1)

weight: 98

b

![Graph](image2)

weight: 27

c

![Graph](image3)

weight: 22.4

2 In Prim’s algorithm, the starting point can be any node, whereas Kruskal’s algorithm starts from the arc of least weight.

In Prim’s algorithm, each new node and arc are added to the existing tree as it builds, whereas in applying Kruskal’s algorithm, the arcs are selected according to their weight and may not be connected until the end.

3 a

![Graph](image4)

Selection order: AH, AG, AB, BC, CD, BF, EF

b 17 340

4 a

![Graph](image5)

b $17 340

c Replace WX with VX. This is the cheapest way to link X to the spanning tree. The total cost is now $231000 + $280000 − $190000 = $240000

5 a Prim’s algorithm identifies the next node to link to the existing tree. Linking a new node cannot form a cycle.

b One minimum connector is shown below.

c The minimum connector is not unique; there are three minimum connectors with total weight 87.

Exercise 3C

1 a Arcs in order:
   AF (9)
   FB (14)
   AC (20)
   AE (25)
   DE (26)
   weight = 94

b Arcs in order:
   RS (28)
   ST (16)
   SU (19)
   UV (37)
   weight = 100

2 Arcs in order:
   BS (78.4)
   SM (70.4)
   SN (89.6)
   NL (59.2)
   weight = 297.6

3 a cost = €1014

b D

c i It is cheaper to translate from E to H then from H to G at a cost of 48 + 52 = 100 euro rather than 159 euro per 1000 words.

ii A direct translation is likely to be more accurate then a translation via another language.
4 a order of arcs
XE (26)
EG (18)
EH (23)
HA (25)
AF (20)
FB (16)
AD (22)
FC (24)
CI (26)

b

X
E
H
A
D

I

C
G

Exercise 3D
1 a

C
A
F
B
E
D

weight = 97

b

S
T
U
R
V

weight = 110

2 AEFBCD weight = 69
BCDEFA weight = 76
CBFAED weight = 73
DCBEAF weight = 68
EAFBDC weight = 68
FAEDCB weight = 72
shortest route is DCBFAE weight = 68
or EAFBDC weight = 68

3 ACBDE weight = 437
BCDAE weight = 395
CBDAE weight = 444
DCBEA weight = 420
ECBDA weight = 432
shortest route is BCDAE weight = 395

Exercise 3E
1 a Shortest route: S − B − C − F − E − G − T
Length of shortest route: 1660

b Shortest route: S − A − B − D − F − E − H
Length 94

c Shortest route: A to H via G: A − D − G − H
Length 96

d Shortest route: A to H not using C: E: A − D − F − E − H
Length 95

2 Shortest route: S − B − C − T
Length of shortest route: 10

3 a Quickest route is SCEHT. Shortest time = 31 min.
b i Route changes to SABDGT. New time = 32 min
ii The driver should change the route to HEGT to save 1 minute. Total travel time 38 min.

Chapter review 3
1 a i Arcs are labelled with initial letters of the nodes.
CK add to tree
SH add to tree
CE add to tree
EK reject
CH add to tree
HW add to tree
CS reject
HQ add to tree
QS reject
QB add to tree
KS reject
DW reject
EW reject

ii EC

K
S
WD
Q
C
H

538
300
740
423
378

340

R

T

S

2 a i LT (3.8) add to tree
MT (3.7) add to tree
MQ (3.8) add to tree
NQ (4.7) add to tree
ST (5.3) reject

ii MQ (3.7) add to tree
LT (3.8) add to tree
MT (4.1) add to tree
NQ (4.7) add to tree
MN (5.3) reject
ST (6.6) add to tree
QR (6.6) add to tree
NP (6.8) add to tree
reject remaining arcs
b Start off the tree with QT and PR then apply Kruskal’s algorithm. Prim’s algorithm requires the ‘growing’ tree to be connected at all times. When using Kruskal’s algorithm the tree can be built from non-connected sub-trees.

3 a

Selection order: AC, AE, ED, AF, FB

b length 332 mm

4 a Arcs in order: Entrance 2–Office; Entrance 2–Entrance 4; Entrance 4–Entrance 3; Office–Entrance 1

Entrance 1

Entrance 2

Entrance 3

Entrance 4

b length 3112 m

5 a 29 30 19 26 17 18 12 24 20 14 12 22 26

b 8 29 27 19 9 26 17 18 12 24 14 22 26

6 a weight = 45 so 45000 m needed

b weight = 47 so 4700 m

7 a Possible paths are A – H – G – E – I – K

and A – H – J – I – K

and A – B – C – K

i A, B, F, H, D, G, J, E, C, I, K

ii 44 – 9 = 35

JK 44 – 9 = 35

DK 35 – 10 = 25

EI 35 – 17 = 18

KL 25 – 10 = 15

GE or 18 – 8 = 10

HJ 15 – 5 = 10

HG 10 – 10 = 0

AH

or

44 – 12 = 32

CK

32 – 25 = 7

BC

7 – 7 = 0

AB


and A – B – C – K

c The arcs could be roads.
The nodes could be junctions.
The number on each arc could be distance in km.
The network, together with Dijkstra’s algorithm, could be used to find the shortest route from A to K.

8 Order of vertex labelling:

S C B A E D G F T

Route: S – C – E – F – T

141 – 101 = 310 FT

310 = 123 = 187

187 – 85 = 102 CE

102 – 102 = 0 SC

9 a

b Students’ own answer

c Prim’s algorithm may be quicker on a graph with a large number of arcs, such as a complete network, as Kruskal’s algorithm would require arcs to be sorted by weight.

10 a

weight of MST = 7930 km

b C – G – B – E – H – A – D – F weight = 9620 km

c MST × 2 = 15860 , nearest neighbour and return = 13080. The method using the nearest neighbour will give the shortest way.
Review exercise 1

1 a

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>f = 0?</th>
</tr>
</thead>
<tbody>
<tr>
<td>645</td>
<td>255</td>
<td>2.53</td>
<td>2</td>
<td>510</td>
<td>135</td>
<td>No</td>
</tr>
<tr>
<td>255</td>
<td>135</td>
<td>1.89</td>
<td>1</td>
<td>135</td>
<td>120</td>
<td>No</td>
</tr>
<tr>
<td>135</td>
<td>120</td>
<td>1.13</td>
<td>1</td>
<td>120</td>
<td>15</td>
<td>No</td>
</tr>
<tr>
<td>120</td>
<td>15</td>
<td>8</td>
<td>8</td>
<td>120</td>
<td>0</td>
<td>Yes</td>
</tr>
</tbody>
</table>

b The first row would be

255 645 0.40 0 0 255

but the second row would then be the same as the first row in the table above. So in effect this new first line would just be an additional row at the start of the solution.

c Finds the highest common factor of a and b.

2 a Total length = 390 cm, so 4 planks are needed.

b 6 planks

c 5 planks

d There are 5 lengths over 50 cm, so none of these can be paired together. Therefore minimum of 5 lengths are required.

3 a 80 55 84 25 34 25 75 17 5 3

b 403 ÷ 100 = 4.03 so 5 bins are needed.

c Bin 1: 84 + 5 + 3

Bin 2: 80 + 17

Bin 3: 75 + 25

Bin 4: 55 + 34

Bin 5: 25

4 a For example, 45 37 18 56 79 90 81 51

b For example, 56 45 79 46 37 90 81 51 18

5 For example,

R P B Y T K M H W G
B H G K R P Y T M W
B G H R P M T Y W
B G H K M P R T W Y

list in order

6 a Since the graph is simple, there are no loops, so each of the degree-5 vertices must be joined to each of the other vertices. This means that each of the other vertices has degree at least 2.

b e.g.

7 a

b 4 c 5

8 a A Hamiltonian cycle is a cycle that includes every vertex.

b ABDCFGEA

9 a GC, FD, GF, reject CD, ED, reject EF, BC, AG, reject AB.

b A 54 G 30 F 25 D 35 E

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
<th>B</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>50</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

Cost = (20 + 25 + 30 + 35 + 50 + 54) × €1000 = €214,000

10 a i Method:

• Start at A and use this to start the tree.

• Choose the shortest edge that connects a vertex already in the tree to a vertex not yet in the tree. Add it to the tree.

• Continue adding edges until all vertices are in the tree. 

AF, FC, or FD, EB, or BC

ii The tree is not unique, there are 2 of them (see above).

b i number of edges = 7 − 1 = 6

ii number of vertices = n + 1

11 a

b BD, {AC} BC, reject CD, DE; length 18 km

c DB, DF, BC, CA, DE

12 a In Prim's algorithm, the starting point can be any node, whereas Kruskal's algorithm starts from the arc of least weight.

In Prim's algorithm, each new node and arc is added to the existing tree as it builds, whereas in applying Kruskal's algorithm, the arcs are selected according to their weight and may not be connected until the end.

b i GH, GI, HF, FD, DA, AB, AC, DE

ii GH, AB, AC, AD, reject BD, DF, GI, reject BC, FH, reject DG, DE

c weight is 76
13 a Route: $S - A - C - G - T$, length: 82

For example:
- 82 - 12 = 70 $GT$
- 70 - 16 = 54 $CG$
- 54 - 20 = 34 $AC$
- 34 - 34 = 0 $SA$

e Shortest route from $S$ to $H + HT$. $S - B - F - H - T$ length: 84

14 a

\[ \begin{array}{c|c|c|c|c|c|c} 
A & 1 & 0 & 4 & B & 3 & 4 \\
0 & 1 & 2 & 6 & 4 & 5 & 8 \\
& 10 & 8 & 16 & 2 & 5 & 4 \\
\end{array} \]

For example:
- $13 - 1 = 12$ $HL$ or $13 - 1 = 12$ $KL$
- $12 - 2 = 10$ $GH$ or $12 - 4 = 8$ $JK$
- $10 - 5 = 5$ $FG$ or $8 - 2 = 6$ $IJ$
- $5 - 4 = 1$ $EF$ or $6 - 5 = 1$ $EI$
- $1 - 1 = 0$ $AE$ or $1 - 1 = 0$ $AE$

b i Possible sets are:
Blue: $ABD, ACD$
Red: $BCF, DEF$

ii For $K_5$, any vertex will have a valency of 5, an edge to each of the other points.

Challenge
1 a Let $G$ be any simple graph with more than one vertex and with number of vertices $= n \geq 2$.

The maximum degree of any vertex in $G$ is $\leq n - 1$. Also, if our graph $G$ is not connected, then the maximum degree is $< n - 1$.

Case 1: Assume that $G$ is connected. We cannot have a vertex of degree 0 in $G$, so the set of vertex degrees is a subset of $S = \{1, 2, \ldots, n - 1\}$. Since the graph $G$ has $n$ vertices and there are $n - 1$ possible degree options, there must be two vertices of the same degree in $G$.

Case 2: Assume that $G$ is not connected. $G$ has no vertex of degree $n - 1$, so the set of vertex degrees is a subset of $S' = \{0, 1, 2, \ldots, n - 2\}$. Again, we have $n$ vertices and $n - 1$ possible degree options, so there must be two vertices of the same degree in $G$.

b i Possible sets are:
Blue: $ABD, ACD$
Red: $BCF, DEF$

ii For $K_5$, any vertex will have a valency of 5, an edge to each of the other points.

Online
Worked solutions are available in SolutionBank.

CHAPTER 4

Prior knowledge check
1 a 16
b Sum of orders $= 2 \times$ number of arcs
4
d From part b, the sum of the valencies must be even.

Exercise 4A
1 a

<table>
<thead>
<tr>
<th>Vertex</th>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
<th>$D$</th>
<th>$E$</th>
<th>$F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

There are 4 nodes with odd degree so the graph is neither Eulerian nor semi-Eulerian.