



PEARSON EDEXCEL INTERNATIONAL GCSE (9–1)

COMPUTER SCIENCE

Student Book

David Waller



PEARSON EDEXCEL INTERNATIONAL
GCSE (9–1)

COMPUTER SCIENCE

Student Book

David Waller
Chris Charles
Pete Dring
Alex Hadwen-Bennett
Jason Welch
Shaun Whorton

SERIES EDITOR

Ann Weidmann

Published by Pearson Education Limited, 80 Strand, London, WC2R 0RL.

<https://www.pearson.com/international-schools>

Copies of official specifications for all Pearson Edexcel qualifications may be found on the website: <https://qualifications.pearson.com>

Text © Pearson Education Limited 2020
Designed by Pearson Education Limited 2020
Typeset by © SPi Global
Edited by Hillary Coaster and Sarah Wright
Original illustrations © Pearson Education Limited 2020
Picture research by Integra
Cover design © Pearson Education Limited 2020
With thanks to Sam Hartburn

The right of David Waller, Chris Charles, Pete Dring, Alex Hadwen-Bennett, Jason Welch, Shaun Whorton to be identified as the authors of this work has been asserted by them in accordance with the Copyright, Designs and Patents Act 1988.

First published 2020

22 21 20
10 9 8 7 6 5 4 3 2 1

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN 978 1 292 31022 0

Copyright notice

All rights reserved. No part of this publication may be reproduced in any form or by any means (including photocopying or storing it in any medium by electronic means and whether or not transiently or incidentally to some other use of this publication) without the written permission of the copyright owner, except in accordance with the provisions of the Copyright, Designs and Patents Act 1988 or under the terms of a licence issued by the Copyright Licensing Agency, 5th Floor, Shackleton House, 4 Battlebridge Lane, London, SE1 2HX (www.cla.co.uk). Applications for the copyright owner's written permission should be addressed to the publisher.

Printed in Slovakia by Neografia

Acknowledgements

Text Credits

Loon LLC: Extracts from Loon LLC, © Loon LLC, 256; **McKinsey & Company:** Stefan Heck, Sri Kasa, Dickon Pinner, Creating Value in the Semiconductor industry, © 2011, McKinsey & Company, 264.

Image Credits

(key: b-bottom; c-center; l-left; r-right; t-top)

Cover Image: Shutterstock/PopTika

123RF: Anton Starikov/123RF 160bc, Jan Mikš/123RF 54, Kubuntu/123RF 214, Lajo_2/123RF 71, Scyther5/123RF 123, Scanrail/123RF 238, Stockyimages/123RF 69; **Alamy Stock Photo:** Jonathan Ball/Alamy Stock Photo 168r, Martin Moxter/imageBROKER/Alamy Stock Photo 248, Paul Broadbent/Alamy Stock Photo 168l, Richard Levine/Alamy Stock Photo 52t, Stuart Kinlough/Alamy Stock Photo 2, 32, 108, 158, 200, 242; **Python Software Foundation:** Copyright ©2001-2019. Python Software Foundation 51, 94, 96, 101; **Shutterstock:** Adisa/Shutterstock 160tl, Asharkyu/Shutterstock 245, Anton Gvozdkov/Shutterstock 265, Burlingham/Shutterstock 34, Baloncici/Shutterstock 106, vii, Bloomicon/Shutterstock 160br, Brian A Jackson/Shutterstock 175, 226, Brainpencil/Shutterstock 181, Cristi180884/Shutterstock 125, Chrisdorney/Shutterstock 166, Crisp/Shutterstock 185, Cybrain/Shutterstock 267t, Cobalt88/Shutterstock 234, DavidTB/Shutterstock 24, Dja65/Shutterstock 160tc, D-VISIONS/Shutterstock 163, Dry3d/Shutterstock 234, ESB Professional/Shutterstock 85, Evgeny Karadaev/Shutterstock 211, Gaby Kooijman/Shutterstock 52b, Gorodenkoff/Shutterstock 172, GraphicNmotion/Shutterstock 268t, Hxdbzxy/Shutterstock 258, Jerry Zitterman/Shutterstock 37, Joshua Davenport/Shutterstock 7, Kavione/Shutterstock 160tr, Luisa Leal Photography/Shutterstock 160bl, Maksym Kapliuk/Shutterstock 25, Michaeljung/Shutterstock 65, Mmaxer/Shutterstock 219, Moomchak V. Design/Shutterstock 268b, Nito/Shutterstock 17, Nikita G. Bernadsky/Shutterstock 266b, Nobeastsofierce/Shutterstock 267b, OlegDoroshin/Shutterstock 5, vii, Platslee/Shutterstock 129, 142, Pakhnyushchy/Shutterstock 160cl, Phonlamai Photo/Shutterstock 224, Paul Fleet/Shutterstock 266t, Silverblackstock/Shutterstock 184, Shutterstock 93, Thinglass/Shutterstock 248, Trong Nguyen/Shutterstock 238, TY Lim/Shutterstock 4, vii, Vlad Kochelaevskiy/Shutterstock 177, 187, Volodymyr Krasnyuk/Shutterstock 235, Wright Studio/Shutterstock 144, Xfilephotos/Shutterstock 237.

Endorsement statement

In order to ensure that this resource offers high-quality support for the associated Pearson qualification, it has been through a review process by the awarding body. This process confirmed that this resource fully covers the teaching and learning content of the specification at which it is aimed. It also confirms that it demonstrates an appropriate balance between the development of subject skills, knowledge and understanding, in addition to preparation for assessment.

Endorsement does not cover any guidance on assessment activities or processes (e.g. practice questions or advice on how to answer assessment questions) included in the resource, nor does it prescribe any particular approach to the teaching or delivery of a related course.

While the publishers have made every attempt to ensure that advice on the qualification and its assessment is accurate, the official specification and associated assessment guidance materials are the only authoritative source of information and should always be referred to for definitive guidance.

Pearson examiners have not contributed to any sections in this resource relevant to examination papers for which they have responsibility.

Examiners will not use endorsed resources as a source of material for any assessment set by Pearson. Endorsement of a resource does not mean that the resource is required to achieve this Pearson qualification, nor does it mean that it is the only suitable material available to support the qualification, and any resource lists produced by the awarding body shall include this and other appropriate resources.

COURSE STRUCTURE	iv
ABOUT THIS BOOK	vi
ASSESSMENT OVERVIEW	viii
UNIT 1: PROBLEM SOLVING	2
UNIT 2: PROGRAMMING	32
UNIT 3: DATA	108
UNIT 4: COMPUTERS	158
UNIT 5: COMMUNICATION AND THE INTERNET	200
UNIT 6: THE BIGGER PICTURE	242
EXAM PREPARATION	272
APPENDICES	291
GLOSSARY	299
INDEX	306

**PROBLEM
SOLVING****2**

1. UNDERSTANDING ALGORITHMS	4
2. CREATING ALGORITHMS	12
3. SORTING AND SEARCHING ALGORITHMS	15
4. DECOMPOSITION AND ABSTRACTION	23
UNIT QUESTIONS	29

PROGRAMMING**32**

5. DEVELOP CODE	34
6. MAKING PROGRAMS EASY TO READ	51
7. STRINGS	54
8. DATA STRUCTURES	62
9. INPUT/OUTPUT	70
10. SUBPROGRAMS	82
11. TESTING AND EVALUATION	92
UNIT QUESTIONS	106

DATA**108**

12. BINARY	110
13. DATA REPRESENTATION	125
14. DATA STORAGE AND COMPRESSION	136
15. ENCRYPTION	144
UNIT QUESTIONS	155

COMPUTERS 158

16. MACHINES AND COMPUTATIONAL MODELS	160
17. HARDWARE	165
18. LOGIC	181
19. SOFTWARE	187
20. PROGRAMMING LANGUAGES	194
UNIT QUESTIONS	198

COMMUNICATION AND THE INTERNET 200

21. NETWORKS	202
22. NETWORK SECURITY	221
23. THE INTERNET AND THE WORLD WIDE WEB	233
UNIT QUESTIONS	240

THE BIGGER PICTURE 242

24. COMPUTING AND THE ENVIRONMENTAL IMPACT OF TECHNOLOGY	244
25. PRIVACY	250
26. DIGITAL INCLUSION	255
27. PROFESSIONALISM	258
28. COMPUTING AND THE LEGAL IMPACT OF TECHNOLOGY	260
29. CURRENT AND EMERGING TRENDS	264
UNIT QUESTIONS	271

EXAM PREPARATION 272

PAPER 1: PRINCIPLES OF COMPUTER SCIENCE	272
PAPER 2: APPLICATION OF COMPUTATIONAL THINKING	283

APPENDICES 291

APPENDIX 1 COMMAND WORDS	291
APPENDIX 2 FLOWCHART SYMBOLS	293
APPENDIX 3 PSEUDOCODE COMMAND SET	294
GLOSSARY	299
INDEX	306

ABOUT THIS BOOK

This book is written for students following the Pearson Edexcel International GCSE (9–1) Computer Science specification and covers both years of the course.

The course has been structured so that teaching and learning can take place in any order, both in the classroom and in any independent learning. The book contains six units that match the six areas of content in the specification: Problem Solving, Programming, Data, Computers, Communication and the Internet, The Bigger Picture.

Each unit is split into multiple sections to break down content into manageable chunks and to ensure full coverage of the specification.

Each unit features a mix of learning and activities. Summary questions at the end of each chapter help you to put learning into practice and prepare for the exam.

Paper 1 is Principles of Computer Science and Paper 2 is Application of Computational Thinking. Knowing how to apply your learning to both of these will be critical for your success in the exam. There is a real applied focus to the book. You will be encouraged to put the theory you are learning into context and apply what you have learned to your own practical activities.

Learning objectives

Each section starts with a list of what you will learn from it. They are carefully tailored to address key assessment objectives central to the course.

Activity

Each chapter includes activities to embed understanding through practical tasks and questions.

110 UNIT 3 DATA 12 BINARY

12 BINARY

GENERAL VOCABULARY
Binary information represented by only two values (e.g. a voltage or no voltage; on or off). There are no communication errors or misunderstandings because there are no small differences.
digital information represented by certain fixed values (e.g. high, medium or low). Any signal between these values would be meaningless and not used. Sending and receiving systems do not have to be as accurate as for analogue communication.
analogue using signals or information represented by a quantity (e.g. an electric voltage or current) that is continuously variable. Changes in the information being represented are indicated by changes in voltage. This method requires very accurate sending and receiving systems.

LEARNING OBJECTIVES

- Understand that computers use binary to represent data (numbers, text, sound, graphics) and program instructions
- Understand how computers represent and manipulate numbers (unsigned integers, signed integers [sign and magnitude, two's complement])
- Be able to convert between binary and denary whole numbers (0–255)
- Understand how to perform binary arithmetic (adds, shifts [logical and arithmetic]) and understand the concept of overflow
- Understand why hexadecimal notation is used and be able to convert between hexadecimal and binary
- Understand that file storage is measured in bytes and be able to calculate file sizes

WHY BINARY?
Binary is needed to represent data and program instructions because of the way in which computers work.

GENERAL VOCABULARY
manipulate to handle or control something in a skilful manner
compressed pressed into a smaller space
transistor a device that controls electric current
intensity the strength of something that can be measured, e.g. light, sound, heat
frequency the number per time unit, e.g. number per second
transmit cause something to move from one place to another

The processor, which processes all of the data and instructions, contains billions of transistors which are connected together to form circuits. The transistors act as switches, similar to light switches. They have only two states: on or off; they either transmit an electric current or they do not. A system with separate states is said to be digital. If there are two states, the system is binary. There are no in-between states with different levels of current as there would be in a dimmer switch, which produces different levels of brightness in a bulb. A system such as this, where there is a continuous range between two values, is said to be analogue.

As there are only two states, on or off, the states are represented by the digits of the binary number system, 1 and 0. All of the data and program instructions processed by a computer are nothing more than streams of millions of 1s and 0s.

Numbers, text, graphics and sound are all represented in the same way, as a series of 1s and 0s. The program instructions that the processor is following allow it to interpret them in different ways.

UNIT 3 DATA 13 DATA REPRESENTATION 131

This standard allows lifelike images, so it is often referred to as true colour. This shape is filled with a colour named 'Alice blue'. In binary you would have to enter 1111000011110001111111 every time you wanted to use it. However, as mentioned on page 120, hexadecimal comes to our rescue. Using hexadecimal, you would only have to enter #F0F8FF.

FILE SIZES
The file size for a bitmap image is calculated by finding the total number of pixels and multiplying that by the number of bits used to represent each pixel, or: Width × Height × Colour depth
The file size of the left-hand image on page 125 is: 4288 (width) × 2848 (height) × 24 (bit colour depth) = 293 093 376 bits
That is, 36 636 672 bytes.

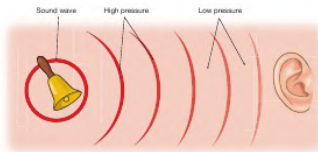
SKILLS REASONING PROBLEM SOLVING

KEY POINT
Calculators are not allowed in the exam so you will be asked to just create an expression to calculate file sizes without showing a final value.

REPRESENTATION OF SOUND
GENERAL VOCABULARY
vibrations quickly moving backwards and forwards about a fixed point

All sounds are caused by vibrations. As objects such as our vocal cords or guitar strings vibrate backwards and forwards, they push the air molecules alongside them, sending a wave of compressed molecules through the air. When these compression waves, or sound waves, reach our ears they set up vibrations in tiny sensory hairs in the inner ear. This sends nerve impulses to the brain, which interprets them as the sounds we hear.

▶ Figure 3.5 Sound waves travelling through the air from a vibrating bell



Extend your knowledge

Push yourself further by looking beyond the content of the course.

Key point

Easy to understand, core information to take away from sections.

Worked example

Key concepts can be demonstrated using step-by-step walkthroughs.

Did you know?
Interesting facts to encourage wider thought and stimulate discussion.

Subject vocabulary and General vocabulary
Useful words and phrases are colour coded within the main text and picked out in the margin with concise and simple definitions. These help understanding of key subject terms and support students whose first language is not English.

4 UNIT 1 PROBLEM SOLVING

1 UNDERSTANDING ALGORITHMS

1 UNDERSTANDING ALGORITHMS

GENERAL VOCABULARY
construct a command to control the order/flow in which instructions are executed (e.g. sequences, selection, repetition)

It is important to be able to **construct** algorithms and be able to read them and follow their logic in solving particular problems.

LEARNING OBJECTIVES

- Understand what an algorithm is
- Understand what algorithms are used for
- Interpret algorithms as flowcharts, pseudocode and written descriptions
- Use and describe the purpose of arithmetic operators
- Understand how to code an algorithm in a high-level language

AN EXAMPLE OF AN ALGORITHM

An interactive map is a useful way to find a route between two locations. Figure 1.1 shows a route between two cities that was calculated by a mapping program.

The route on this interactive map has been calculated using an algorithm.

SUBJECT VOCABULARY
unambiguous this means that the instructions cannot be misunderstood. Simply saying 'turn' would be ambiguous (i.e. unclear) because you could turn left or right. All instructions given to a computer must be unambiguous or it won't know what to do.
sequence an ordered set of instructions
algorithm a precise method for solving a problem

DID YOU KNOW?
The computer program that created the algorithm to map travel from Beijing to Shanghai was following an algorithm of its own – an algorithm ordering it how to create another algorithm!

► Figure 1.1 A route calculated by a mapping program

5

UNIT 1 PROBLEM SOLVING

1 UNDERSTANDING ALGORITHMS

SUCCESSFUL ALGORITHMS

There are three points to consider when deciding whether an algorithm is successful or not.

- Accuracy** – it must lead to the expected **outcome** (e.g. create a route from Beijing to Shanghai).
- Consistency** – it must produce the same result each time it is run.
- Efficiency** – it must solve the problem in the shortest possible time, using as few computer resources as possible. In this example, the mapping software is replacing a manual method. If it were no faster than looking in an atlas, then it would not be an improvement on the older method. Later in the unit there is a section on algorithms that are used to sort and search data. Some of these algorithms are more efficient than others and will sort the data far more quickly.

GENERAL VOCABULARY
outcome the final result of an action
consistency not changing; always the same
atlas a book of maps

THE RELATIONSHIP BETWEEN ALGORITHMS AND PROGRAMS

Algorithms and programs are closely related, but they are not the same. An algorithm is a detailed design for a solution; a program is when that design is implemented.

SUBJECT VOCABULARY
high-level programming language a programming language that is similar to natural human language

This unit is all about algorithms. We look at how algorithms are implemented in **high-level programming languages** in Unit 2.

DISPLAYING AN ALGORITHM

We carry out many everyday tasks using algorithms because we are following a set of instructions to achieve an expected result, for example, making a cup of coffee. If we have performed the task many times before, we usually carry out the instructions without thinking. But if we are doing something unfamiliar, such as putting together a flat-pack chest of drawers, then we follow the instructions very carefully.

An algorithm can be expressed in different ways.

WRITTEN DESCRIPTIONS

A written description is the simplest way of expressing an algorithm. Here is an algorithm describing the everyday task of making a cup of instant coffee:

ALGORITHM FOR MAKING A CUP OF COFFEE

Fill kettle with water.
Turn on kettle.
Place coffee in cup.
Wait for water to boil.
Pour water into cup.
Add milk and sugar.
Stir.

Summary
Quickly recap the core content of each section.

Skills
Relevant exam questions have been assigned the key skills that you will gain from undertaking them, allowing for a strong focus on particular academic qualities. These transferable skills are highly valued in further study and the workplace.

Assessment objectives
Questions are tagged with the relevant assessment objectives that are being examined.

UNIT 3 DATA

UNIT QUESTIONS

156

UNIT QUESTIONS

SKILLS REASONING, PROBLEM SOLVING **A02**

1 a Add together the following 8-bit numbers. (1)
01011001
11100111
b Identify the problem that this addition has created. (1)

SKILLS REASONING, CRITICAL THINKING **A02**

2 a Carry out a three-place logical right shift on the following binary number: 10010011 (2)
b Explain the effect of performing a right shift on a binary number. (2)
c Describe the steps needed to convert the binary number 11101110 into a hexadecimal one and show the result. (2)

HINT

1 a This is a straightforward question. Carry out the calculation and write the result in the space provided. Remember to carry over if the addition of each pair of digits is greater than 1.
b This just requires a one- or two-word answer.

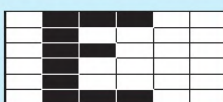
2 a Again, carry out the shift and write the result.
b Here you have to explain what effect the shift will have. You could say that it is equivalent to multiplying the number by...
c A longer answer is required. You should show stage by stage how the conversion is carried out. For example, you could start by saying what the 8-bit binary number is divided into. You should set out the explanation clearly and it could be in the form of a diagram.

SKILLS CRITICAL THINKING **A01**

SKILLS CRITICAL THINKING, PROBLEM SOLVING **A02**

3 Explain what is meant by a 'pixel'. (2)

4 The following diagram shows a black and white image consisting of 36 pixels.
a Explain why 36 bits are needed to represent the pixels in the image. (2)
b Write the bit pattern needed to represent these pixels. (4)



► Figure 3.19 A 36-pixel image of the letter 'E'

c State the number of bits per pixel that would be needed if the image was 16 colours rather than 2. (1)

Checkpoint
Checkpoints help you to check and reflect on your learning at the end of each section. Strengthen questions help you to consolidate basic knowledge and understanding. Challenge questions are more demanding and ask you to apply your learning.

Unit questions
These exam-style questions are found at the end of each unit. They are tailored to the Pearson Edexcel specification to allow for the practice and development of exam writing technique. They also allow for practice responding to the command words used in the exams.

ASSESSMENT OVERVIEW

The following tables give an overview of the assessment for this course. You should study this information closely to help ensure that you are fully prepared and know exactly what to expect in each part of the assessment.

PAPER 1	PERCENTAGE	MARK	TIME	AVAILABILITY
PRINCIPLES OF COMPUTER SCIENCE Written exam paper Paper code 4CP0/01 Externally set and assessed by Pearson Edexcel Single tier of entry	50%	80	2 hours	June exam series First assessment June 2019
PAPER 2	PERCENTAGE	MARK	TIME	AVAILABILITY
APPLICATION OF COMPUTATIONAL THINKING Practical and written exam paper Paper code 4CP0/02 Externally set and assessed by Pearson Edexcel Single tier of entry	50%	80	3 hours	June exam series First assessment June 2019

ASSESSMENT OBJECTIVES AND WEIGHTINGS

ASSESSMENT OBJECTIVE	DESCRIPTION	% IN INTERNATIONAL GCSE
A01	Demonstrate knowledge and understanding of the key principles of computer science	27.5
A02	Apply knowledge and understanding of key concepts and principles of computer science	42.5
A03	Analyse problems in computational terms: • to make reasoned judgements • to design, program, test, evaluate and refine solutions	30

RELATIONSHIP OF ASSESSMENT OBJECTIVES TO UNITS

UNIT NUMBER	ASSESSMENT OBJECTIVE		
	A01	A02	A03
PAPER 1	21.5%	21%	7.5%
PAPER 2	6%	21.5%	22.5%
TOTAL FOR INTERNATIONAL GCSE	27.5%	42.5%	30%

ASSESSMENT SUMMARY

PAPER 1	DESCRIPTION	MARKS	ASSESSMENT OBJECTIVES
PRINCIPLES OF COMPUTER SCIENCE PAPER CODE 4CP0/01	Structure Paper 1 contributes 50% of the total marks for the Computer Science qualification. Students must answer all questions. The paper consists of multiple-choice, short open-response, open-response and extended open-response answer questions.	The total number of marks available is 80	Questions will test the following Assessment Objectives: AO1 – 21.5% AO2 – 21% AO3 – 7.5%
	Content summary This paper will primarily assess knowledge and understanding of the basic principles of computer science, including some coverage of how these principles are applied when solving problems that relate to a particular situation.		
	Assessment This is a single-tier exam paper and all questions cover the full ranges of grades from 9–1. The assessment duration is 2 hours.		
PAPER 2	DESCRIPTION	MARKS	ASSESSMENT OBJECTIVES
APPLICATION OF COMPUTATIONAL THINKING PAPER CODE 4CP0/02	Structure Paper 2 contributes 50% of the total marks for the Computer Science qualification. Students must answer all questions. The paper consists of multiple-choice, short open-response, open-response, extended open-response answer and task-based questions. The task-based questions will be carried out using a computer system under supervision. All other questions requiring a written response will be answered in the paper.	The total number of marks available is 80	Questions will test the following Assessment Objectives: AO1 – 6% AO2 – 21.5% AO3 – 22.5%
	Content summary This paper will primarily assess the practical application of computational thinking, whereby learners will create, use and adapt existing algorithms to solve problems in a particular situation. This paper will also test students' knowledge and understanding of the topics.		
	Assessment This is a single-tier exam paper and all questions cover the full ranges of grades from 9–1. The assessment duration is 3 hours. A choice of three programming languages will be available (Python, C# or Java). A pseudocode reference document will be available for learners to reference during the assessment.		



UNIT 1

PROBLEM SOLVING

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

In this unit you will learn about algorithms, which are the basis of computer programming, and how they can be presented as flowcharts and pseudocode. You will learn about the basic constructs of an algorithm such as sequence, selection and iteration and how these are used to solve problems using computational thinking. You will also look at completing and correcting algorithms in addition to algorithms to sort and search data. In the next unit you will learn how to code these algorithms using pseudocode and high-level programming languages.

1 UNDERSTANDING ALGORITHMS

GENERAL VOCABULARY

construct a command to control the order/flow in which instructions are executed (e.g. sequences, selection, repetition)

It is important to be able to **construct** algorithms and be able to read them and follow their logic in solving particular problems.

LEARNING OBJECTIVES

- Understand what an algorithm is
- Understand what algorithms are used for
- Interpret algorithms as flowcharts, pseudocode and written descriptions
- Use and describe the purpose of arithmetic operators
- Understand how to code an algorithm in a high-level language

AN EXAMPLE OF AN ALGORITHM

SUBJECT VOCABULARY

unambiguous this means that the instructions cannot be misunderstood. Simply saying 'turn' would be ambiguous (i.e. unclear) because you could turn left or right. All instructions given to a computer must be unambiguous or it won't know what to do

sequence an ordered set of instructions

algorithm a precise method for solving a problem

An interactive map is a useful way to find a route between two locations. Figure 1.1 shows a route between two cities that was calculated by a mapping program.

The route on this interactive map has been calculated using an algorithm.

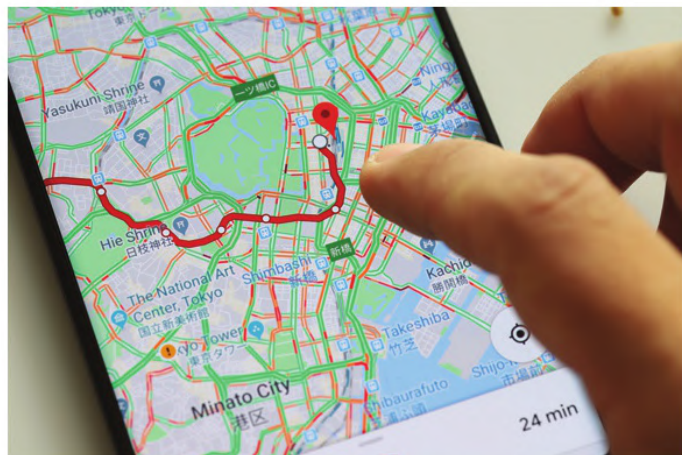
- It is **unambiguous** in telling the driver exactly what to do, like 'turn left', 'turn right' or 'go straight'.
- It is a **sequence** of steps.
- It can be used again and will always provide the same result.
- It provides a solution to a problem, in this case, how to get from Beijing to Shanghai.

A solution to a problem with these characteristics is called an **algorithm**. Most problems have more than one solution, so different algorithms can be created for the same problem.

DID YOU KNOW?

The computer program that created the algorithm to map travel from Beijing to Shanghai was following an algorithm of its own – an algorithm ordering it how to create another algorithm!

► **Figure 1.1** A route calculated by a mapping program



SUCCESSFUL ALGORITHMS

GENERAL VOCABULARY

outcome the final result of an action

consistency not changing; always the same

atlas a book of maps

There are three points to consider when deciding whether an algorithm is successful or not.

- Accuracy – it must lead to the expected **outcome** (e.g. create a route from Beijing to Shanghai).
- **Consistency** – it must produce the same result each time it is run.
- Efficiency – it must solve the problem in the shortest possible time, using as few computer resources as possible. In this example, the mapping software is replacing a manual method. If it were no faster than looking in an **atlas**, then it would not be an improvement on the older method. Later in the unit there is a section on algorithms that are used to sort and search data. Some of these algorithms are more efficient than others and will sort the data far more quickly.

THE RELATIONSHIP BETWEEN ALGORITHMS AND PROGRAMS

SUBJECT VOCABULARY

high-level programming language
a programming language that is similar to natural human language

Algorithms and programs are closely related, but they are not the same. An algorithm is a detailed design for a solution; a program is when that design is implemented.

This unit is all about algorithms. We look at how algorithms are implemented in **high-level programming languages** in Unit 2.

DISPLAYING AN ALGORITHM

We carry out many everyday tasks using algorithms because we are following a set of instructions to achieve an expected result, for example, making a cup of coffee. If we have performed the task many times before, we usually carry out the instructions without thinking. But if we are doing something unfamiliar, such as putting together a flat-pack chest of drawers, then we follow the instructions very carefully.

An algorithm can be expressed in different ways.

WRITTEN DESCRIPTIONS

A written description is the simplest way of expressing an algorithm. Here is an algorithm describing the everyday task of making a cup of instant coffee:



ALGORITHM FOR MAKING A CUP OF COFFEE

Fill kettle with water.
Turn on kettle.
Place coffee in cup.
Wait for water to boil.
Pour water into cup.
Add milk and sugar.
Stir.

SKILLS

REASONING,
PROBLEM SOLVING

ACTIVITY 1

GETTING TO SCHOOL

Produce a written description of an algorithm for getting to school. It should start with leaving home and end with arriving at school. For example, the algorithm could start with 'Walk to bus stop'.

Check your algorithm with other members of the group. Would your algorithm work for others? Are there any general statements that are common to all algorithms?

SUBJECT VOCABULARY

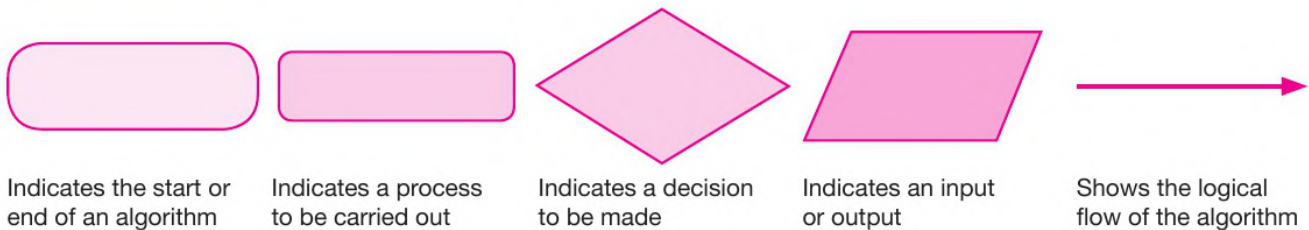
flowchart shows an algorithm as a diagram. Each step in the algorithm is represented by a symbol. Symbols are linked together with arrows showing the order in which steps are completed

FLOWCHARTS

Flowcharts can be used to show an algorithm as a diagram. They provide a more visual display.

There are special symbols that have to be used in a flowchart. You can't just make up your own, because nobody else would be able to follow your algorithm.

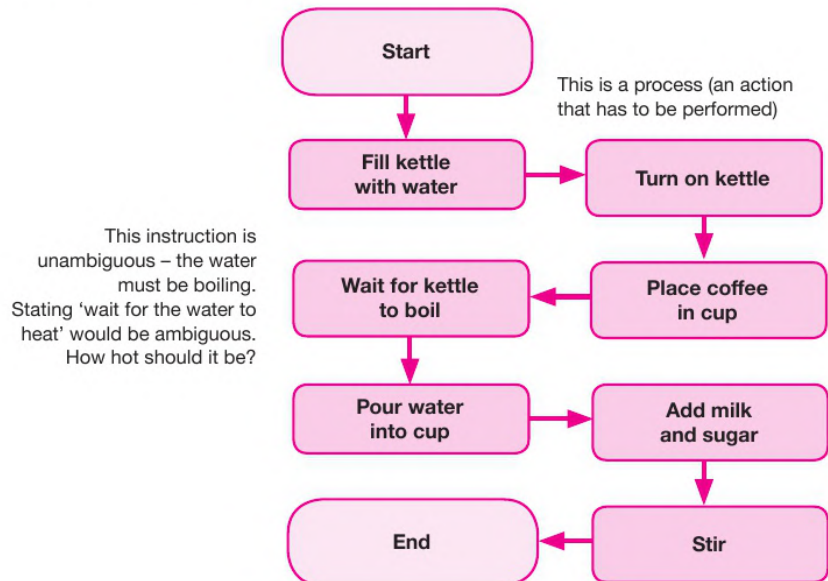
Figure 1.2 shows the flowchart symbols that should be used.



▲ Figure 1.2 Flowchart symbols

The flowchart in Figure 1.3 is an alternative way of showing the algorithm for making a cup of coffee as a written description.

► Figure 1.3 Flowchart of an algorithm to make a cup of coffee



SKILLS → PROBLEM SOLVING

ACTIVITY 2

SCHOOL JOURNEY FLOWCHART

Display the 'journey to school' algorithm that you created in Activity 1 as a flowchart.

SKILLS → PROBLEM SOLVING

ACTIVITY 3

BATH FLOWCHART

A student has created a written algorithm for preparing a bath. Working with a partner, display the following as a flowchart. You may need to change the order or add actions.

- Put in the plug.
- Fill the bath to the correct level.
- Check the temperature is OK.

GENERAL VOCABULARY

basis an important idea or fact that something is based on

SUBJECT VOCABULARY

pseudocode a structured, code-like language that can be used to describe an algorithm

developer a person whose job it is to create new software

logic the principles and reasoning underlying the constructs and elements to be applied in solving problems

The algorithms you have looked at so far are designed for humans to follow. Algorithms also form the **basis** of computer programs. Computers are mindless machines that simply do exactly what they are told. They follow a set of instructions, but they can carry out these instructions far more quickly than humans. That is why they are so useful.

PSEUDOCODE

In addition to flowcharts and written descriptions, algorithms can also be expressed in **pseudocode**. The pseudocode can be used to code the solution in an actual programming language.

It allows the **developer** to concentrate on the **logic** and efficiency of the algorithm without having to bother about the rules of any particular programming language. It is relatively straightforward to translate an algorithm written in pseudocode into any high-level programming language.

SKILLS → RESEARCH

ACTIVITY 4

INVESTIGATING PSEUDOCODE

Different organisations or examination boards have their own unique versions of pseudocode. Investigate the Pearson Edexcel pseudocode that you will need for your International GCSE course and which will be used in this book.

EXAMPLE OF A SIMPLE ALGORITHM

To introduce the Pearson Edexcel pseudocode, here is a simple written algorithm that asks the user to input two numbers and then outputs the result of adding them together.

WRITTEN DESCRIPTION

ALGORITHM FOR ADDING TWO NUMBERS

Enter first number.
Enter second number.
Calculate total by adding first and second numbers.
Output total.

FLOWCHART



▲ Figure 1.4 Flowchart showing the adding of two numbers

SUBJECT VOCABULARY

variable a 'container' used to store data. The data stored in a variable is referred to as a value. The value stored in a variable is not fixed. The same variable can store different values during the course of a program and each time a program is run

identifier a unique name given to a variable or a constant. Using **descriptive** names for variables makes code much easier to read

arithmetic operator an **operator** that performs a calculation on two numbers

GENERAL VOCABULARY

operator a character that represents an action, e.g. 'x' represents a multiplication and '/' a division

descriptive describing something clearly

PSEUDOCODE

ALGORITHM FOR ADDING TWO NUMBERS

```

SEND 'Please enter the first number' TO DISPLAY
RECEIVE firstNumber FROM KEYBOARD
SEND 'Please enter the second number' TO DISPLAY
RECEIVE secondNumber FROM KEYBOARD
SET total TO firstNumber + secondNumber
SEND total TO DISPLAY
  
```

The pseudocode gives clear step-by-step instructions that the computer will be expected to carry out. It also introduces some important programming concepts.

- The numbers entered by the user are stored in two **variables** with the **identifiers** `firstNumber` and `secondNumber`.
- The result of adding the numbers together is stored in the variable `total`.
- Text has to be placed in quotation marks (single or double) if it is to be displayed. For example, 'Please enter the first number' (or "Please enter the first number").
- Quotation marks are not used if a variable is to be displayed. If they were, the word 'total' in the last instruction would be displayed instead of the number it represents.
- **Arithmetic operators** are used to perform calculations. Table 1.1 shows the arithmetic operators.

ARITHMETIC OPERATORS

OPERATOR	FUNCTION	EXAMPLE
+	Addition: add the values together.	$8 + 5 = 13$ myScore1 + myScore2
-	Subtraction: subtract the second value from the first.	$17 - 4 = 13$ myScore1 - myScore2
*	Multiplication: multiply the values together.	$6 * 9 = 54$ numberBought * price
/	Real division: divide the first value by the second value and return the result including decimal places.	$13 / 4 = 3.25$ totalMarks/numberTests
DIV	Quotient : like division, but it only returns the whole number or <i>integer</i> .	$13 \text{ DIV } 4 = 3$ totalMarks DIV numberTests
MOD	Modulus /modulo: this will return the remainder of a division.	$13 / 4 = 3 \text{ remainder } 1$ Therefore $13 \text{ MOD } 4 = 1$
^	Exponentiation: this is for 'to the power of'.	$3 ^ 3 = 27$ It is the same as writing 3^3

▲ Table 1.1 Arithmetic operators

VARIABLES AND CONSTANTS

KEY POINT

When you chose a variable name, it should be descriptive of the data it will hold, e.g. distance or length. Long variable names are easier to misspell!!

SUBJECT VOCABULARY

constant a 'container' that holds a value that never changes; like variables, constants have unique identifiers

GENERAL VOCABULARY

quotient a result found by dividing one quantity by another

modulus the remainder after the division of one number by another

power the small number written to the right and above another number to show how many times it should be multiplied by itself

Variables play an important role in algorithms and programming. The value stored by a variable can change as a program is running. Variables are extremely useful in programming because they make it possible for the same program to process different sets of data.

A **constant** is the opposite of a variable. It is a 'container' that holds a value that always stays the same. Constants are useful for storing fixed information, such as the value of pi, the number of litres in a gallon or the number of months in a year.

Each variable and constant in an algorithm has to have a unique identifier. It is important to choose descriptive names for identifiers. This will make your code much easier to read. For example, a variable to hold a user's first name could be given the identifier `firstName` to show the data it contains. If it were given the identifier `X` instead, it wouldn't clearly show what data it contained.

NAMING CONVENTIONS FOR VARIABLES AND CONSTANTS

It is sensible to write identifiers in the same way throughout an algorithm. A common method is to use 'camel case' for compound words (e.g. `firstName`, `secondName`) with no space between words and the second word starting with a capital letter. Alternatively, you could capitalise the first letter of both words, e.g. `FirstName`, `SecondName`, or separate the words with an underscore, e.g. `first_name`, `second_name`, known as 'snake case'.

SKILLS PROBLEM SOLVING, ANALYSIS

KEY POINT

Whichever method you use for naming conventions, you must use it consistently and not keep switching between the different methods.

ACTIVITY 5
WRITING ALGORITHMS IN PSEUDOCODE

Here is a written description of an algorithm:

Enter the first number.

Enter the second number.

The third number is equal to the first number multiplied by the second number.

Display the third number.

Express this algorithm in pseudocode.

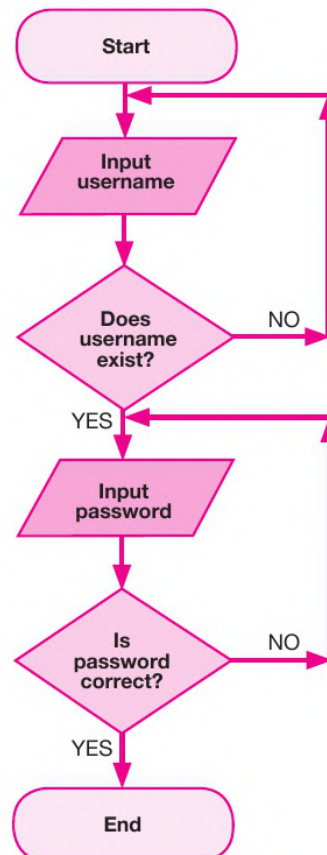
SKILLS PROBLEM SOLVING

EXTEND YOUR KNOWLEDGE

When you enter a search term into Google®, a list of links to websites is returned. But why are they presented in that particular order? With a partner, research the **PageRank** algorithm that Google uses to rate the importance of websites and write a short report about your findings.

ACTIVITY 6
WRITTEN DESCRIPTIONS OF ALGORITHMS

This algorithm is displayed as a flowchart.



▲ Figure 1.5 Flowchart of an algorithm

Produce a written description of this algorithm.

SKILLS CRITICAL THINKING

SKILLS DECISION MAKING

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING,
PROBLEM SOLVING

SKILLS PROBLEM SOLVING

CHECKPOINT

Strengthen

S1 Produce a written description of an algorithm for borrowing a book from the library.

S2 What is the function of each of the seven arithmetic operators?

S3 What is a variable? Why are they useful?

S4 What is the difference between a variable and a constant?

Challenge

C1 Produce a flowchart describing an algorithm for making a cheese sandwich.

C2 Write an algorithm expressed in pseudocode that receives three numbers from the keyboard, then calculates and displays the average.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again.

- For S1 re-read 'The relationship between algorithms and programs'.
- For S2 study Table 1.1.
- For S3 and S4 look again at 'Variables and constants'.

GENERAL VOCABULARY

structured organised so that the parts work well together

SUMMARY

- An algorithm is a precise method for solving a problem.
- Algorithms can be displayed as written descriptions, flowcharts and in pseudocode.
- Pseudocode is a **structured**, code-like language.
- Pseudocode is translated into program code.
- Arithmetic operators are used in calculations.
- Variables and constants are 'containers' for storing data. The value stored in a variable can change, whereas the value of a constant never changes.
- Selecting descriptive names for identifiers makes code easier to read.

2 CREATING ALGORITHMS

In this section you will learn how to create algorithms to solve particular problems using the constructs of sequence, selection and iteration. You will also practise displaying algorithms in flowcharts and pseudocode.

LEARNING OBJECTIVES

- Understand how to create an algorithm to solve a particular problem
- Make use of programming constructs (sequence, selection and iteration) and use appropriate conventions (flowchart, pseudocode, written description, draft program code)

ALGORITHMS FOR COMPUTERS

SUBJECT VOCABULARY

construct a smaller part from which something is built. Letters and numbers (i.e. a to z and 0 to 9) are the constructs we use to build our language and convey meaning. Bricks and cement are the basic constructs of a building

selection a construct that allows a choice to be made between different alternatives

iteration a construct that means a process is repeated. An action is repeated until a **condition** is met or a particular outcome is reached. It is often referred to as a 'loop'

GENERAL VOCABULARY

ambiguous when a statement or command does not have one obvious meaning but can be interpreted in different ways

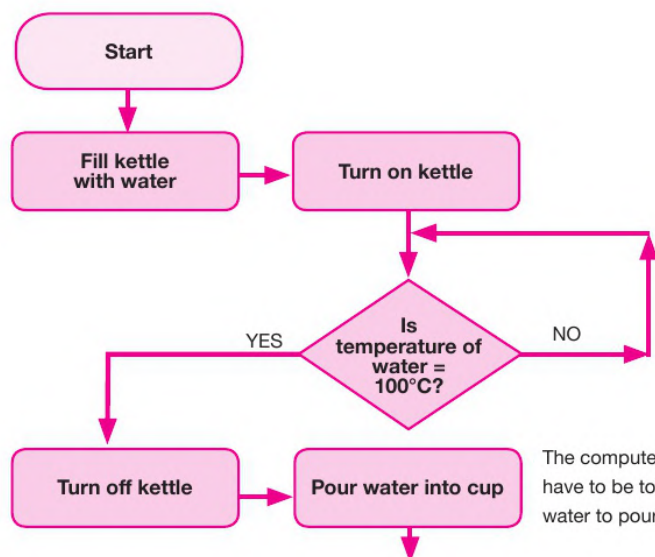
condition something that must happen before something else can happen

There was an **ambiguous** statement in the algorithm for making a cup of coffee. After filling the kettle with water and adding coffee to the cup, the next instruction was 'Wait for water to boil'.

A human can understand that this instruction means they have to keep checking the kettle over and over again until the water is boiling, but a computer is unable to understand an instruction like this in the same way. It would just wait. And wait. Forever.

Even worse, the algorithm didn't state clearly how to tell the water was boiling. Through experience, we humans assume the water is boiling when there is lots of steam, sound and bubbles; or, even better, when the kettle turns itself off. An algorithm for a computer would have to state that it must wait until the water reached 100°C.

A version of this part of the algorithm, suitable for a computer, is shown in Figure 1.6. This example introduces two new **constructs** from which algorithms are created. We have already met the construct sequence – step-by-step instructions in the correct order. To add to this, we now have **selection** and **iteration**.



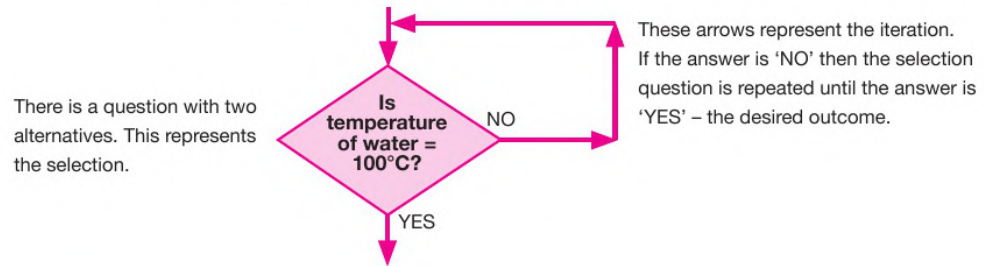
The computer would also have to be told how much water to pour into the cup!

► Figure 1.6 Part of an algorithm suitable for a computer for making coffee

REPRESENTING SELECTION AND ITERATION IN A FLOWCHART

Selection and iteration are represented in a flowchart as shown in Figure 1.7.

► **Figure 1.7** Selection and iteration in a flowchart



SKILLS

REASONING,
PROBLEM SOLVING

DID YOU KNOW?

We use iteration in our daily lives whenever we carry out an action over and over again. For example, at mealtimes we keep on eating until our plate is empty or we have had enough to eat.

When we're travelling by car and the traffic lights are red, we have to keep waiting until they change to green.

An actor repeats their lines over and over again, learning more each time until they know them all.

SKILLS

PROBLEM SOLVING,
ANALYSIS

ACTIVITY 7

GUESSING GAMES

A student is creating a guessing game. A player has to enter a number no greater than 10. If it is too high, they are informed that they have made an error. But if it is within the range 1 to 10, they are told whether or not they have guessed the correct number. (Assume that the correct number is 3.)

Can you make an algorithm to solve this problem and express it as a written description and a flowchart?

Compare your solution to others in your group.

Check that they are correct and would produce the correct outcome.

Are some of the algorithms more efficient than others? Do they use fewer commands?

ACTIVITY 8

CALCULATING GRADES

A school uses this algorithm to calculate the grade that students achieve in end-of-topic tests.

```

RECEIVE testScore FROM KEYBOARD
IF testScore >= 80 THEN
    SEND 'A' TO DISPLAY
ELSE
    IF testScore >= 70 THEN
        SEND 'B' TO DISPLAY
    ELSE
        IF testScore >= 60 THEN
            SEND 'C' TO DISPLAY
        ELSE
            IF testScore > 0 THEN
                SEND 'D' TO DISPLAY
            ELSE
                SEND 'FAIL' TO DISPLAY
  
```



```

END IF
END IF
END IF
END IF

```

What would be the output of this algorithm for these test scores: 91, 56 and 78?

ITERATION

When writing programs, it is often necessary to repeat the same set of statements several times. Instead of making multiple copies of the statements, you can use iteration to repeat them. The algorithm for making a cup of coffee includes an instruction to keep waiting until the water in the kettle boils.

SKILLS CRITICAL THINKING,
DECISION MAKING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

CHECKPOINT

Strengthen

S1 How are sequence, selection and iteration used in algorithms? Give examples to justify your answer.

Challenge

C1 Develop an algorithm using a flowchart that asks the user to enter their height (in metres) and weight (in kilograms) and displays their body mass index (BMI). The formula for calculating BMI is $\text{weight}/\text{height}^2$.

C2 Develop an algorithm expressed as a flowchart to control the heating in a house. A thermostat monitors the temperature within the house. During the week the temperature should be 20°C between 06.00 and 08.30 in the morning and between 17.30 and 22.00 at night. At weekends it should be 22°C between 08.00 and 23.00. If the temperature in the house falls below 10°C at any time the boiler is switched on.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again.

- For S1 have a look at the Subject vocabulary sections in 'Understanding algorithms' and 'Creating algorithms'.

SUMMARY

- The constructs sequence, selection and iteration are the basic building blocks of algorithms.

3 SORTING AND SEARCHING ALGORITHMS

Whenever we want to find information, we carry out a search. Just imagine the number of people around the world who are using the same search engine at the same time. Without efficient searching algorithms, we would have to wait a long time to be shown the results.

We are on all sorts of lists (for example, school and college, clubs and groups, voting registers) and all these must be searched to find relevant information. Sorting information is also important to facilitate efficient searches.

LEARNING OBJECTIVES

- Understand how standard algorithms work (bubble sort, merge sort, linear search, binary search)
- Understand how the choice of algorithm is influenced by the data structures and data values that need to be manipulated
- Evaluate the fitness for purpose of algorithms in meeting specified requirements efficiently, using logical reasoning and test data

Two of the most common tasks in computer programs are sorting data into a particular order and searching for particular items of information.

There might be millions of items of stored data and searching for information wouldn't be efficient if the data was not sorted. Imagine the confusion and difficulty of having to find something in a dictionary that wasn't in alphabetical order. Or planning a trip with train timetables that weren't sorted into time order. Even small lists such as football league tables or the Top 20 music charts are much more useful if they are sorted into order.

SORTING ALGORITHMS

SUBJECT VOCABULARY

ascending order this is arranging items from smallest to largest (e.g. 1, 2, 3)

descending order this is arranging items from largest to smallest (e.g. 3, 2, 1)

traversal travel across or through something.

As sorting is such a widely used procedure, many algorithms have been created to carry it out. As with all algorithms, some are more efficient than others.

BUBBLE SORT

When data is sorted, different items must be compared with each other and moved so that they are in either **ascending order** or **descending order**.

The bubble sort algorithm starts at one end of the list and compares pairs of data items. If they are in the wrong order, they are swapped. The comparison of pairs continues to the end of the list, each complete **traversal** of the list being called a 'pass'. This process is repeated until there have been no swaps during a pass. This indicates that the items must all be in the correct order.

The algorithm can be described as follows.

BUBBLE SORT (ASCENDING ORDER)

- 1 Start at the beginning of the list.
- 2 Compare the values in position 1 and position 2 in the list – if they are not in ascending order then swap them.
- 3 Compare the values in position 2 and position 3 in the list and swap if necessary.
- 4 Continue to the end of the list.
- 5 If there have been any swaps, repeat steps 1 to 4.

WORKED EXAMPLE

Here is an example of a bubble sort in action.

Pass 1

4	2	6	1	3	Items 1 and 2 must be swapped.
2	4	6	1	3	Items 1 and 2 are swapped.
2	4	6	1	3	Items 2 and 3 are already in ascending order.
2	4	6	1	3	Items 3 and 4 must be swapped.
2	4	1	6	3	Items 3 and 4 have been swapped.
2	4	1	6	3	Items 4 and 5 must now be swapped.
2	4	1	3	6	Items 4 and 5 have been swapped.

Pass 2

2	4	1	3	6	Items 1 and 2 are in correct order.
2	4	1	3	6	Items 2 and 3 must be swapped.
2	1	4	3	6	Items 2 and 3 have been swapped.
2	1	4	3	6	Items 3 and 4 must be swapped.
2	1	3	4	6	Items 3 and 4 have been swapped.
2	1	3	4	6	Items 4 and 5 do not need to be swapped.

Pass 3

2	1	3	4	6	Items 1 and 2 must be swapped.
1	2	3	4	6	Items 1 and 2 have been swapped.
1	2	3	4	6	All items are now in the correct order.

▲ Figure 1.8 A bubble sort

DID YOU KNOW?

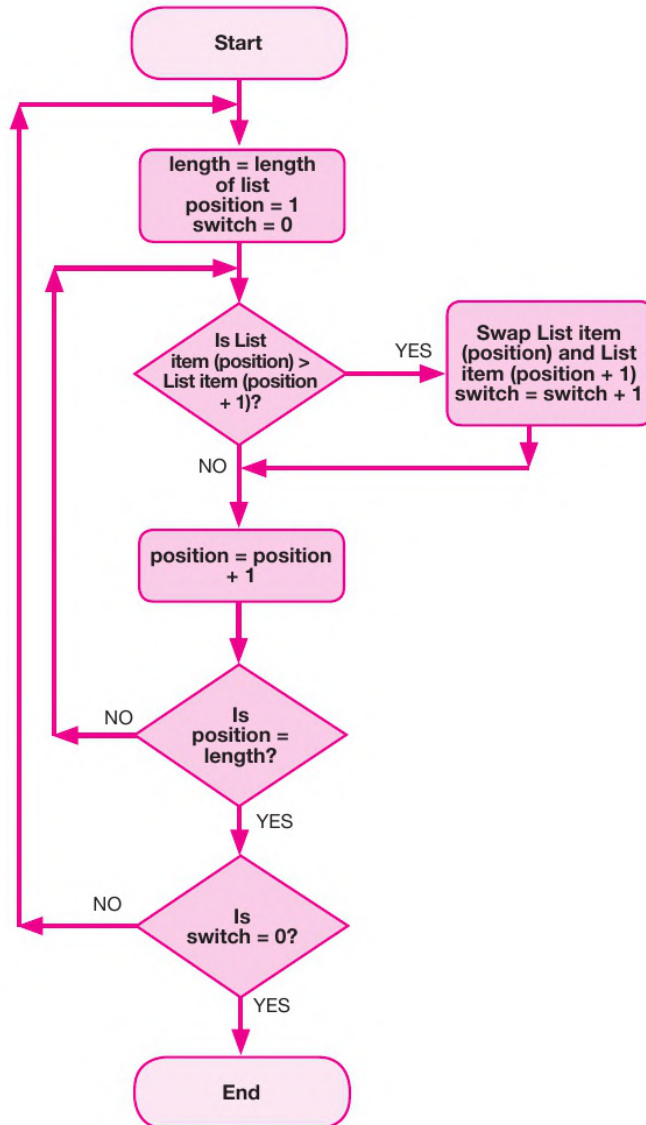
Do you know why it is called ‘bubble sort’? If you look carefully, you can see that the largest items gradually move to the end, like bubbles rising in water. After the first pass, the largest number is in its correct position. Then after the second pass, the next largest is in its correct position. This happens on each pass and so if the algorithm is to be made more efficient the last set of comparisons can be left out.



It would take a human three passes to carry out this bubble sort. A computer would need four passes because it must continue until there have been no swaps; it cannot just look at all of the numbers at once and see that they are all in order.

The bubble sort algorithm can be represented as a flowchart as shown in Figure 1.9.

► **Figure 1.9** Bubble sort algorithm written as a flowchart



SKILLS CRITICAL THINKING

ACTIVITY 9

HOW DOES BUBBLE SORT WORK?

Study the flowchart of the bubble sort algorithm.

Using the variables declared, can you explain the logic behind the algorithm? How does it function to sort a list?

GENERAL VOCABULARY

repeatedly many times

SUBJECT VOCABULARY

recursion a process that is repeated. For example, a document can be checked and edited, checked and edited and so on until it is perfect

MERGE SORT

Merge sort is a sorting algorithm that divides a list into two smaller lists and then divides these until the size of each list is one. **Repeatedly** applying a method to the results of a previous application of the method is called **recursion**.

In computing, a problem is solved by repeatedly solving smaller parts of the problem. A part of a program can be run and rerun on the results of the previous run (e.g. repeatedly dividing a number by 2).

WORKED EXAMPLE

Here is an example of a merge sort which will sort the following list into ascending order.



The list is split into half with recursion to produce a left list and a right list each time.



This continues until there is only one item in each list. Therefore, each list is sorted into order.



The left and right lists are now merged through recursion with the items in the correct order.



The leftmost items in each list are the lowest items of those lists and the algorithm compares them – in this case 4 with 2. The 2 is inserted in the new list and the 4 is then compared with the second number of the right list – 6. The 4 is inserted and the 6 is compared with the second number of the left list.



The algorithm now merges these two lists in the same way to produce the final sorted list. 1 is compared with 2 and then 2 with 3, 3 with 4, etc.



KEY POINT

In the exam, you will be expected to show the intermediate stages when the algorithms are applied to data.

SKILLS PROBLEM SOLVING

ACTIVITY 10

USING MERGE SORT

Using a table like the one in the worked example on page 18, show how the following list would be sorted into descending order using merge sort.

48, 20, 9, 17, 13, 21, 28, 60

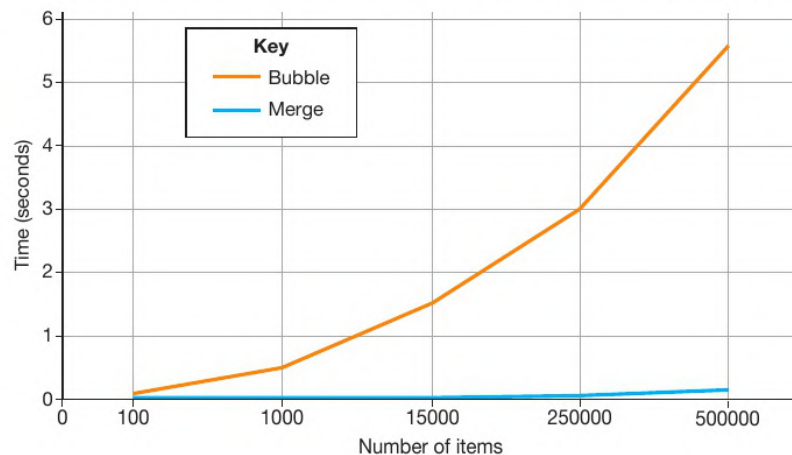
EXTEND YOUR KNOWLEDGE

Only two sorting algorithms are required for the specification: bubble sort (the slowest) and merge sort (one of the most efficient). There are far more, and many of them are relatively easy to code. Research the insertion and selection sorts.

► Figure 1.10 A graph comparing the performance of bubble and merge sort algorithms

EFFICIENCY OF SORTING ALGORITHMS

This graph compares the performance of the bubble and merge sort algorithms.



SUBJECT VOCABULARY

brute force an algorithm design that does not include any techniques to improve performance, but instead relies on computing power to try all possibilities until the solution to a problem is found

divide and conquer an algorithm design that works by dividing a problem into smaller and smaller sub-problems, until they are easy to solve. The solutions to these are then combined to give a solution to the complete problem

The bubble and merge sort algorithms demonstrate two alternative approaches to algorithm design.

The bubble sort algorithm is said to be using **brute force** because it starts at the beginning and completes the same task over and over again until it has found a solution.

The merge sort uses the **divide and conquer** method because it repeatedly breaks down the problem into smaller sub-problems, solves those and then combines the solutions.

The graph shows that a bubble sort is far slower at sorting lists of more than 1000 items, but for smaller lists the time difference is too small to be of importance.

As the bubble sort algorithm is easier to code, it could be beneficial to use it for smaller lists of less than 1000 items.

SEARCHING ALGORITHMS

To find a specific item in a list involves carrying out a search. Like sorting, some methods of searching are more efficient than others.

LINEAR SEARCH

A linear search is a simple algorithm and not very sophisticated. It simply starts at the beginning of the list and goes through it, item by item, until it finds the item it is looking for or reaches the end of the list without finding it.

A linear search is **sequential** because it moves through the list item by item.

GENERAL VOCABULARY

sequential following in order, one after the other

LINEAR SEARCH

- 1 Start at the first item in the list.
- 2 Compare the item with the search item.
- 3 If they are the same, then stop.
- 4 If they are not, then move to the next item.
- 5 Repeat 2 to 4 until the end of the list is reached.

BINARY SEARCH

Like a merge sort, a binary search uses a 'divide and conquer' method.

In a binary search the middle or **median** item in a list is repeatedly selected to reduce the size of the list to be searched – another example of recursion. If the selected item is too high or too low, then the items below or above that selected item can be searched.

To use this method, the list must be sorted into ascending or descending order. It will not work on an unsorted list.

SUBJECT VOCABULARY

median the middle number when the numbers are put in ascending or descending order (e.g. if there are 13 numbers, then the 7th number is the median). If there are an even number of items in a list, the median is the mean of the middle two numbers (e.g. if there are 10 numbers, add the 5th and 6th numbers together and divide the result by 2). In a binary search, the higher of the two numbers would be chosen

DID YOU KNOW?

You have probably used a binary search method when trying to guess a number between two limits. If you are asked to guess a number between 1 and 20 you will probably start at 10, the middle number. If you are told this is too high, you will then guess 5, the middle number between 1 and 10, and then repeat this method until you find the correct one.

BINARY SEARCH (ITEMS IN ASCENDING ORDER)

- 1 Select the median item of the list.
- 2 If the median item is equal to the search item, then stop.
- 3 If the median is too high, then repeat 1 and 2 with the sub-list to the left.
- 4 If the median is too low, then repeat 1 and 2 with the sub-list to the right.
- 5 Repeat steps 3 and 4 until the item has been found or all of the items have been checked.

WORKED EXAMPLE

In this list, the search item is the number 13.

3	13	24	27	31	39	45	60	69	SELECT THE MEDIAN NUMBER.
---	----	----	----	----	----	----	----	----	------------------------------

As this is too high, the sub-list to the left of the median must be searched.

3	13	24	27	THE MEDIAN NUMBER OF THIS SUB-LIST IS NOW SELECTED.
---	----	----	----	--

This is again too high and so the sub-list to the left must be searched.

3	13	THE MEDIAN NUMBER IS NOW THE SEARCH ITEM.
---	----	--

▲ Figure 1.11 Binary search including sub-lists

In this example, it took three attempts to find the search item. A linear search would have accomplished this with only two attempts.

SKILLS → PROBLEM SOLVING

ACTIVITY 11

USING BINARY SEARCH

Display the stages of a binary search, as in the worked example above, to find the number 13 in this list.

3 9 13 15 21 24 27 30 36 39 42 54 69

Compare your results with those of others in your group. Are all your answers the same?

EFFICIENCY OF SEARCHING ALGORITHMS

In the example on page 20, the linear search was more efficient because it only had to carry out two comparisons instead of the three for a binary search. But is this always the case?

Searching algorithms can be compared by looking at the 'worst case' and the 'best case' for each one.

WORKED EXAMPLE

If you wanted to find a particular item in a list of 1000 items, these are the best- and worst-case scenarios for the linear search and binary search algorithms.

Linear search

A linear search starts at the first item and then works through sequentially.

The best case would be if the item is first in the list.

The worst case would be if it is last in the list.

Therefore, in this example the average would be 500 comparisons.

Binary search

The best case would be if the item is in the median position in the list. The search would require only one comparison.

For the worst case it would have to choose the following medians until it finally hit the target.

(This assumes that the target is always smaller than the median.)

Attempt	Median
1	500
2	250
3	125
4	63
5	32
6	16
7	8
8	4
9	2
10	1

Therefore, the worst case for the binary search is ten comparisons.

The binary search is therefore far more efficient than the linear search.

So, should a binary search be used every time? That depends on the circumstances. The binary search has one great disadvantage. The list must be already sorted into ascending or descending order. Therefore, a sorting algorithm must be applied before the search.

If the list is to be searched just once then a linear search would be better, but if there is a large list that will be searched many times then sorting the list and using a binary search would be better. Once the list has been sorted, new items can be inserted into the correct places.

SKILLS CRITICAL THINKING

SKILLS DECISION MAKING

SKILLS CRITICAL THINKING

CHECKPOINT

Strengthen

S1 What are the differences between the 'bubble sort' and 'merge sort' algorithms?

S2 How does a binary search algorithm find the search item?

Challenge

C1 When might a linear search be preferable to a binary search, even if the binary search algorithm is more efficient?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again.

- For S1 have a look at the 'Sorting algorithms' section.
- For S2 have a look at the 'Binary search' section.

SUMMARY

- There are many algorithms for sorting and searching data.
- The choice of algorithm depends on the data that is to be processed.
- If only a small amount of data needs to be processed, then a simpler, but less efficient search algorithm may be the best choice. The time difference of the search or sort time will be negligible.

4 DECOMPOSITION AND ABSTRACTION

GENERAL VOCABULARY

thought process the act of using your mind to consider or think about something

When computer scientists attempt to solve problems by producing algorithms and coding them into programs, they approach the problems in a particular way. This method has been given the name 'computational thinking'. Before they create a structured solution or algorithm and code it into a program, they must define and analyse the problems. This section will introduce two of the **thought processes** they use – decomposition and abstraction.

LEARNING OBJECTIVES

- Analyse a problem, investigate requirements (inputs, outputs, processing, initialisation) and design solutions
- Decompose a problem into smaller sub-problems
- Understand how abstraction can be used effectively to model aspects of the real world
- Program abstractions of real-world examples

PROBLEM SOLVING

SUBJECT VOCABULARY

computational thinking the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by a computer

decomposition breaking a problem down into smaller, more manageable parts, which are then easier to solve

abstraction the process of removing or hiding unnecessary detail so that only the important points remain

The tasks of a computer scientist include defining and analysing problems; creating structured solutions – algorithms; and coding the solutions into a form that can be implemented by a computer. These tasks are part of what is known as **computational thinking**.

One of the skills required for computational thinking is algorithm design (which we've covered in detail in this unit). If there is a fault in the algorithm design, then the program will not work, however good a coder you are. Two other skills are **decomposition** and **abstraction**.

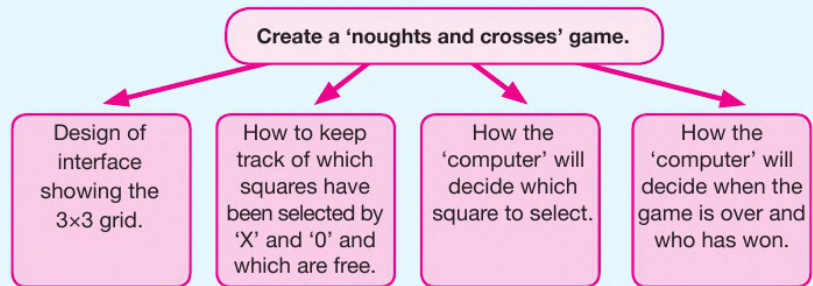
DECOMPOSITION

Decomposition is usually the first step in the problem-solving process. Once a problem has been broken down and the sub-problems have been identified, algorithms can be developed to solve each of them.

Decomposition means that sub-problems can be worked on by different teams at the same time. As smaller algorithms are developed for each sub-problem, it is easier to spot and correct errors. When the algorithm is developed into a program, code can be used again.

WORKED EXAMPLE

A student has been set the task of creating a computer version of the game 'noughts and crosses' (also known as 'tic-tac-toe') where a user plays against the computer.



► Figure 1.12 Sub-problems to be solved to create a noughts and crosses computer program

The diagram shows some of the sub-problems that must be solved in order to solve the complete problem and create a version of the game.

ABSTRACTION

GENERAL VOCABULARY

abstract to remove something from somewhere

We use abstraction all the time in our daily lives. We **abstract** the essential features of something so that we can understand what people are trying to communicate.

Somebody might say, 'I was walking down the street when I saw a cat'. You immediately understand what they mean by 'street' – probably a road with a pavement and houses or shops along the side of it. Similarly, you can picture the cat – a small animal with fur, four legs and a tail. An animal that is basically 'cattish'. You have extracted the basic properties of animals called cats so that you can recognise one when you see one, or imagine one when somebody talks about a cat.



▲ Is this the street and cat you imagined?

What you imagine is very unlikely to match the actual street and cat that the person experienced. But, because of our ability to abstract, the person did not

have to go into unnecessary detail about exactly where they were and what they saw. They wouldn't get very far with the story if they did.

When we create algorithms, we abstract the basic details of the problem and represent them in a way that a computer is able to process.

GENERAL VOCABULARY

model to make a simple version of something to show how it works

WORKED EXAMPLE

Yasmin is designing a computer version of a game in which users have to throw a die to find out their number of moves.

In the computer game, the users can't have an actual die, so she will have to design a 'pretend' or virtual die that behaves in exactly the same way as a real-life die.

Yasmin will have to use her powers of abstraction to work out the essential features of a die and then represent them in computer code.

To represent the die, she will have to create a routine that will select a random number from 1 to 6 because that's what a die does.

Yasmin has used abstraction to **model** a real-life event.

LEVELS OF ABSTRACTION

There are different levels or types of abstraction. The higher the level of abstraction, the less detail is required. We use abstraction all the time in accomplishing everyday tasks.

When programmers write the 'print' command they do not have to bother about all of the details of how this will be accomplished. They are removed from them. They are at a certain level of abstraction.

A driver turning the ignition key to start a car does not have to understand how the engine works or how the spark to ignite the petrol is generated. It just happens and they can simply drive the car. That is abstraction.



▲ A game of noughts and crosses

AN EXAMPLE – NOUGHTS AND CROSSES

Figure 1.12 showed some of the sub-problems that the problem of creating a noughts and crosses game could be divided into. The following could be written at a high level of abstraction.

- The computer goes first. Then the user. This continues until either one wins, or all of the squares have been used.

Immediately a pattern can be recognised – a loop will be needed.

Inputs and outputs

The following inputs from the user will be needed.

- Start the game.
- Entries for the user.
- Select a new game or finish.

The following outputs will be needed.

- A message to inform the user when it is their turn.
- A message to inform the user if they try to select a square that has already been used.

- A message to inform the user if the game is a draw.
- A message to inform the user if they or the computer has won.
- A message to ask the user if they want to play another game or want to finish.

Processing and initialisation

The following processing will be needed.

- Set up the grid with the nine squares.
- Initialise all variables to a start value.
- Decide which square the computer will select.
- Allow the user to select a square.
- Check if the user has selected an already used square.
- Check if the computer or the user has won.
- Check if all squares have been used and the game is a draw.
- Allow the user to select a new game or finish.

The solution is still at a high level of abstraction and more details will need to be added.

For example, the programmer will need to decide how the game will record which player has selected each square; how the computer will decide which square to select; how the game will decide if the computer or the user has won.

The programmer will have to go into more and more detail or move to lower levels of abstraction.

Eventually, the programmer will be able to design an algorithm for the game and code it using a high-level programming language such as Python or Java. Even before they start to implement the game, they will need to plan how they will test the finished program to make sure that it works correctly, what test data they will use and what outcomes it should produce.

CODING AN ALGORITHM

High-level programming languages make it easier for a programmer to write code. Unfortunately, the processor that has to execute the program cannot understand the language it is written in. It therefore needs a translator to translate the code into the only language it does understand – a stream of 1s and 0s.

These high-level languages are therefore at a high level of abstraction – very far removed from the actual language of a computer.

The processing can be split into parts. For example, in the example of the noughts and crosses game there could be separate algorithms for:

- deciding where the computer should make its next selection – it could be called ‘computer entry’
- checking if the computer or the player has won – it could be called ‘check if won’
- checking if there are any empty squares left – it could be called ‘check draw’.

These separate algorithms could be used when they are needed. It is efficient because it means that the same code doesn’t have to be rewritten whenever it is needed.

SUBJECT VOCABULARY

subprogram a self-contained module of code that performs a specific task. It can be 'called' by the main program when it is needed

SKILLS

REASONING,
PROBLEM SOLVING

GENERAL VOCABULARY

decompose to divide something into smaller parts

EXTEND YOUR KNOWLEDGE

Complete the noughts and crosses game by coding it in the language you are studying. See if you can end up with a working game.

SKILLS

CRITICAL THINKING

SKILLS

CRITICAL THINKING

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING

SKILLS

CRITICAL THINKING

SKILLS

PROBLEM SOLVING

SKILLS

DECISION MAKING

These items of code are called **subprograms**.

In Unit 2, we'll look in detail at how subprograms are used to reduce the complexity of programs and to make them easier to understand.

In the die example above, the designer could write a subprogram called 'die' that generates a random number from 1 to 6. In the main program the designer could just call the 'die' subprogram without having to think about how to implement it each time.

ACTIVITY 12

CREATING A QUIZ GAME

In a game, each player spins a wheel that is divided into four colours: red, blue, green and yellow. Each player has to answer a question on a particular topic depending on the colour next to a pointer when the wheel stops. Red is for science, blue for history, green for general knowledge and yellow for geography. A player scores two points if they answer correctly on the first attempt and one point for being correct on the second attempt. The first player to reach 30 points is the winner.

Your task is to design a computer version of the game for up to four players. You must analyse the problem and list all of the requirements; **decompose** the problem, list all the sub-problems and write a brief description of each; list all of the input, output and processing requirements.

One of the requirements that will have to be modelled is the spinning of the wheel. Using a written description and pseudocode shows how this could be done.

CHECKPOINT

Strengthen

S1 What is meant by 'decomposition'? What are the benefits it provides for programmers?

S2 What is meant by 'abstraction'?

S3 A student is creating a model of the cost of a car journey, a real-world problem. Write down the important items she will have to include in her model and how they interact to calculate the cost of the journey.

Challenge

C1 Can you think of some examples when 'decomposition' and 'abstraction' are used when solving a problem?

C2 In your own words, can you explain what is meant by 'computational thinking'?

C3 Explain how we use abstraction in our daily lives when we are communicating with others.

DID YOU KNOW?

Computer models or 'simulations' of real life are widely used. It is far cheaper and safer to train pilots on flight simulators than on real aircraft. They are also used in weather forecasting, designing and testing new cars and bridges and even teaching people to drive. Computer models are used by all governments around the world to experiment with the short and long-term effects of changing variables such as tax rates on the economy.

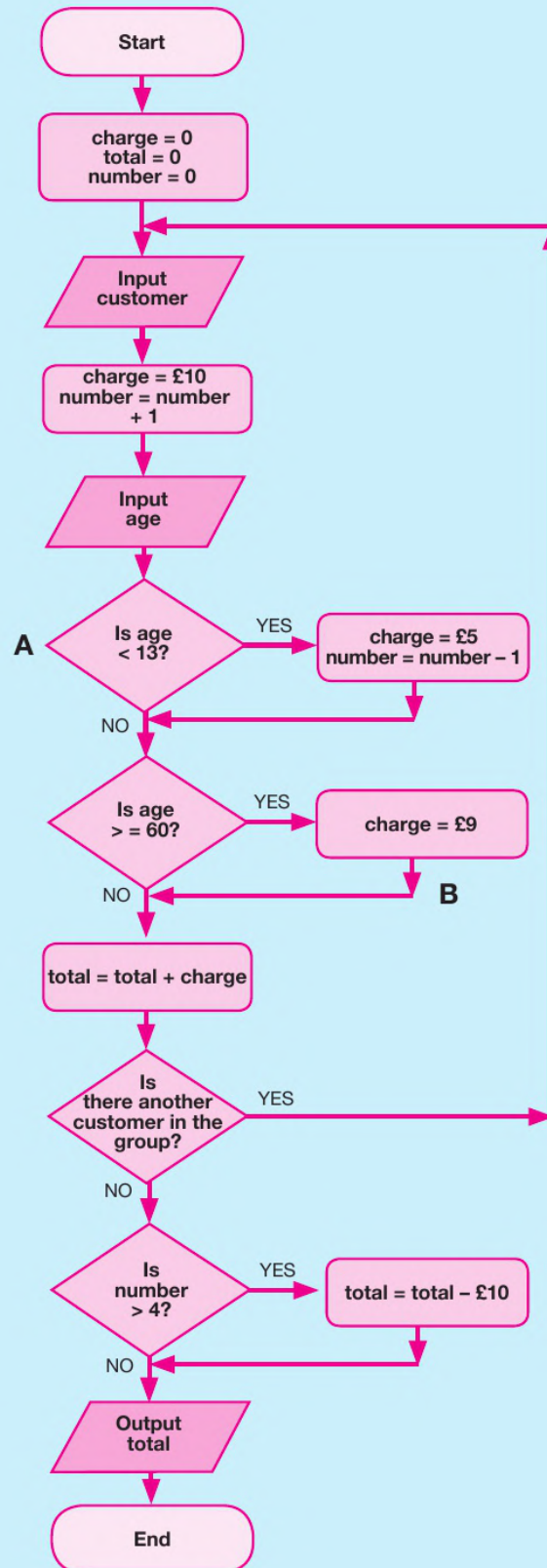
How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read the sections about 'Decomposition' and 'Abstraction'.

SUMMARY

- Computational thinking is an approach to solving problems, such as traffic flow in a city, or how many products a business needs to make and sell to produce a profit. It includes techniques such as decomposition and abstraction.
- Problems are easier to solve if they are decomposed into smaller sub-problems.
- Abstraction is used to remove unnecessary detail to make a problem easier to understand and solve. For example, when modelling traffic flow in a city, unnecessary details could include the colours of the vehicles or the ages of the drivers.
- When designing a solution to a problem the inputs, outputs and processing requirements should be identified at the outset.

UNIT QUESTIONS

- Figure 1.13 Flowchart of an algorithm showing charges in a theme park



The flowchart in Figure 1.13 displays an algorithm used by Holiday Theme Parks Limited.

SKILLS PROBLEM SOLVING A02

- 1 Explain how the algorithm calculates the total amount that should be paid. (4)

SKILLS PROBLEM SOLVING A02

- 2 Give two variables that are used in the algorithm. (2)

SKILLS PROBLEM SOLVING A02

- 3 In the flowchart, two of the constructs are labelled A and B. State the type of each construct. (2)

SKILLS PROBLEM SOLVING A02

- 4 The Lim family is visiting the park. The family consists of two children, one aged 8 and one aged 10, their two parents and their grandfather, who is aged 65. Use the algorithm to calculate how much the family should have to pay for entry. (4)

HINT

Spend some time studying the algorithm to ensure that you fully understand it.

In **1** you are asked to 'explain' how the algorithm works. A longer answer is required, which includes all of the stages of the algorithm. Use the correct terms to explain the constructs.

In **2** and **3** short answers are sufficient.

In **4** calculate the charge for each person using the rules of the algorithm. Then calculate the overall charge and check to see if the family qualifies for a group discount.

SKILLS PROBLEM SOLVING A02

- 5 A teacher has stored learner surnames as shown below.

Marek	Jackson	Bachchan	Wilson	Abraham	French	Smith
-------	---------	----------	--------	---------	--------	-------

Identify the stages of a bubble sort when applied to this data. (5)

SKILLS PROBLEM SOLVING A02

- 6 The teacher has a sorted list of names from another class as shown below.

Azikiwe	Bloom	Byrne	Davidson	Gateri	Hinton	Jackson	Linton	Smith	Wall
---------	-------	-------	----------	--------	--------	---------	--------	-------	------

Identify the stages of a binary search to find the name 'Jackson' when applied to this list. (4)

HINT

These questions are testing knowledge of the sort and search algorithms.

In question **5**, the answer should be set out to show how the data is progressively sorted using the bubble sort and the result of each pass should be shown.

Question **6** is to check that you know that this is a **recursive method** where the median is repeatedly selected.

SUBJECT VOCABULARY

recursive method a recursive method calls a function over and over until a required goal is met

SKILLS PROBLEM SOLVING A03

- 7 Create an algorithm to calculate the cost of sending a parcel.

If the weight of the parcel is 2 kg or under then the standard charge is \$3. There is then a charge of \$2 for each extra kilogram up to 10 kg. After 10 kg the charge per extra kilogram is \$3.

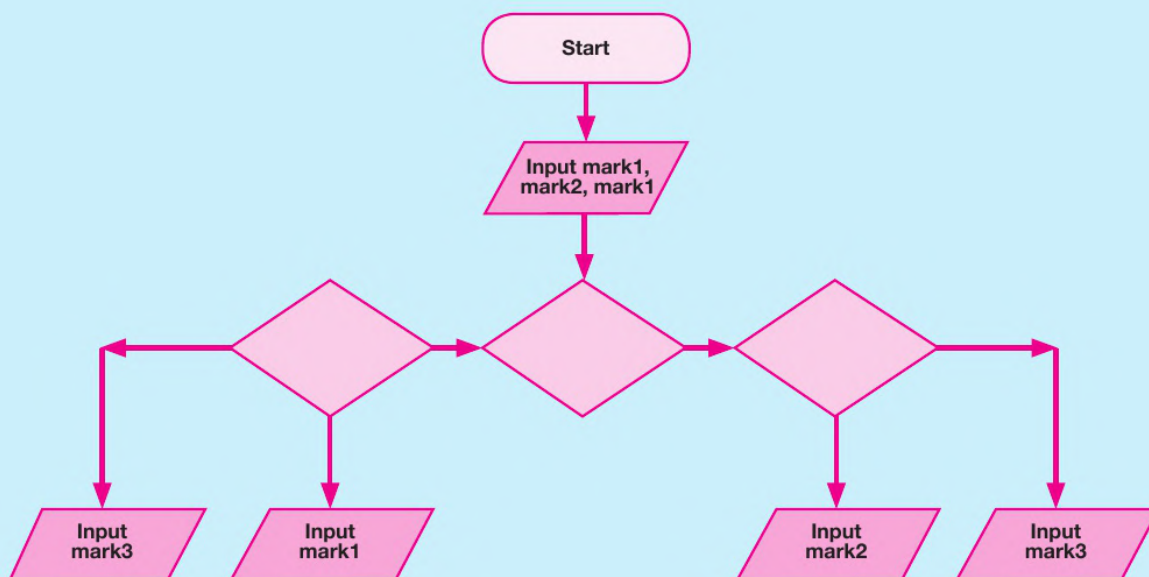
- a Display your algorithm as a flowchart. (5)
b Construct your algorithm as pseudocode. (5)

SKILLS PROBLEM SOLVING A03

- 8 A learner hands in three homework assignments, which were each given a mark out of 10. All of the marks were different. The following is part of an algorithm to find the highest mark but some of the decision symbols are empty.

Complete the decision symbols and add 'YES' and 'NO' labels where required.

(6)

**SKILLS** PROBLEM SOLVING A02

- 9 A list is made up of the numbers 4, 1, 2, 6, 3, 5. Identify the steps involved when sorting this list using a bubble sort algorithm.

(2)



UNIT 2

PROGRAMMING

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

In this unit you will learn about translating algorithms into programs written in a high-level language using constructs, such as variables and arrays. You will also look at how programs can be structured using subprograms and how data input can be validated to ensure that it is reasonable. Humans are not perfect. Most programs have one or two errors and you will look at how programs can be tested, and errors corrected.

5 DEVELOP CODE

PROGRAMMING LANGUAGES

Throughout the book, we have coloured the different programming languages in order for you to easily find the one you are looking for.

The colours are:

Pearson Edexcel pseudocode

Python

Java

C#

Once an algorithm has been developed to solve a particular problem, it has to be coded into the programming language that the developer is using. This usually means that the written descriptions, flowcharts and pseudocode have to be converted into actual programming code.

As you will be expected to understand, use and edit Pearson Edexcel pseudocode in the examination, examples will be given in the pseudocode as well as in Python, Java and C#.

LEARNING OBJECTIVES

- Explain the difference between algorithms and programs
- Code an algorithm in pseudocode and a high-level programming language
- Describe the characteristics of data types and select appropriate data types for variables
- Use sequencing, selection and iteration constructs in your programs

ALGORITHMS AND PROGRAMS

SUBJECT VOCABULARY

execution the process by which a computer carries out the instructions of a computer program

As you learnt in Unit 1, an algorithm is a precise method of solving a problem. It consists of a sequence of unambiguous, step-by-step instructions. A program is an algorithm that has been converted into program code so that it can be **executed** by a computer. A well-written algorithm should be free of logic errors and easy to code in any high-level language.

As part of this course you will learn to write programs in a high-level programming language. All high-level programming languages are like natural human languages, which makes them easier for humans to read and write but impossible for computers to understand without the help of a translator. You will learn more about how a program written in a high-level language is translated into machine code – the language of computers – in Unit 4. The aim of this unit is to develop your programming skills.

► A computer programmer at work writing code



DATA TYPES

GENERAL VOCABULARY

determine to discover facts about something

As you learnt in Unit 1, algorithms use variables (named memory locations) to store values. Variables have a variety of uses. For example, controlling the number of times a loop is executed, **determining** which branch of an IF statement is taken, keeping running totals and holding user input.

When algorithms are converted into programs, the computer needs to be told what type of data is stored in each variable. Every programming language has a number of built-in data types.

DATA TYPE	DESCRIPTION	EXAMPLE	EXAMPLES OF USE
integer	Used to store whole numbers without a fractional part	30	age = 30 number = 5
real or float	Used to store numbers with a fractional part (decimal place). Real numbers are sometimes referred to as floats (short for floating point)	25.5	weight = 25.5 price = 12.55
Boolean	Only has two possible values: True or False	False	correct = False lightOn = True
character*	A character can be a single letter, a symbol, a number or even a space. It is one of the four basic data types	'm'	gender = 'm' char = ':'
string	A set of characters which can include spaces and numbers and are treated as text rather than numbers	'the computer'	name = 'Catherine' type = 'liquid'

*Python does not have a character data type.

▲ Table 2.1 Common data types

GENERAL VOCABULARY

assign to give somebody a particular task

When writing pseudocode, you don't have to specify the data types of variables. However, data types become much more important once you start programming in a high-level language. This is because the data type of a variable determines the operations that can be performed on it.

For example, the result of multiplying a value by 5 differs according to its data type.

integer	$8 * 5 = 40$
real	$8.0 * 5 = 40.0$
character	$'8' * 5 = '88888'$

The method of declaring variables differs between programming languages. Some languages, such as Python, automatically select the appropriate data type for a variable based on the data **assigned** to it. Others, such as Java and C#, require the data type of variables to be declared before the variables can be used.

VARIABLE INITIALISATION

When a variable is declared, the computer gives it a location in its memory. Initially, this location is empty, so before a variable can be used it has to be given a value.

HINT

Although you don't have to declare the data types of variables in pseudocode, it is a good idea to do so. You can simply list the variables and their data types at the start of an algorithm, for example:

```
INTEGER age
REAL weight
BOOLEAN correct
CHARACTER gender
```

You can also specify the data type of a variable in the RECEIVE statement, for example:

```
RECEIVE age FROM (INTEGER)
KEYBOARD
RECEIVE price FROM (REAL)
KEYBOARD
```


SUBJECT VOCABULARY

initialise to set variables to their starting values at the beginning of a program or subprogram

initialisation the process of assigning an initial value to a variable

assignment statement the SET...TO command is used to initialise variables in pseudocode, for example:

```
SET anotherGo TO 0
SET correct TO False
```

You can put an initial value into a variable by:

- initialising it when the program is run (e.g. SET total TO 0, in pseudocode)
- reading a value from a keyboard or other device (e.g. RECEIVE admissionCharge FROM (INTEGER) KEYBOARD).

Once a variable has been **initialised** an **assignment statement** is used to change its value (e.g. SET total TO total + admissionCharge).

In Python this would be:

```
total = 0
total = total + admissionCharge
```

In Java this would be:

```
Scanner scan = new Scanner(System.in);
int admissionCharge = scan.nextInt();
int total = 0;
total = total + admissionCharge;
```

In C#, variables must be declared before use. When you declare a variable, you need to state the data type that the variable will store, for example:

```
// declare a variable called total that will be used to
// store floating point numbers and assign the value 0.0 to it
float total = 0.0;
// add the value stored in the variable admissionsCharge
// to the total
total = total + admissionsCharge;
```

GENERAL VOCABULARY

intended designed for a certain purpose

If a variable, such as a loop counter, is **intended** to hold a running total, then it should always be initialised to a starting value. Some programming languages won't execute if the programmer fails to do this; others will do so but may well produce some unexpected results.

SKILLS

CRITICAL THINKING,
DECISION MAKING

ACTIVITY 1

DATA TYPES

- 1 Investigate what data types are available in the high-level language you are studying. Produce a table similar to Table 2.1 to give a summary of your findings.
- 2 What do you think is an appropriate data type for each of these items?
 - a the test score of an individual learner
 - b the average score for a group of learners
 - c whether or not the pass mark for the test has been achieved.
- 3 Look back over the algorithms you wrote in Unit 1 and find instances of variable initialisation.

SUBJECT VOCABULARY

type coercion the process of converting the value stored in a variable from one data type to another

TYPE COERCION

Sometimes the data type of a variable gets changed during program execution. This is known as **type coercion**. For example, if an integer value and a real value are used in an arithmetic operation, the result will always be a real.

In Python, type coercion is done automatically, for example, the output from the following program is 3.25.

```
x = 1
y = 2.25
z = x + y
print(z)
```

Type coercion is also automatic in Java:

```
int x = 1;
double y = 2.25;
double z = x + y;
System.out.print(z);
```

In C# this is generally referred to as casting. As with Python this can be automatic for many data type conversions and is known as implicit casting. For more complicated conversions you may need to research explicit casting.

One frequently used type of explicit casting is converting from a string to a numeric value. This is used when getting input from a user as `Console.ReadLine`. For example, the following C# code will output 3.25 and demonstrates automatic type coercion.

```
int x;           // declares a variable called x that will store integers
double y, z;     // declares two variables called y and z that will store
                 // floating point numbers

x = 1;
y = 2.25;
z = x + y;
Console.WriteLine(z);
```

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
DECISION MAKING



ACTIVITY 2

MONITORING VISITOR NUMBERS

A theme park uses a program to monitor the number of people entering and exiting the park. The maximum number of visitors at any one time must not exceed 10 000. When the number of people in the park reaches the maximum, a 'Park Full' message is displayed at the entrance gate. Children can visit the park free of charge. Adults must pay AED 125 admission. The program records the amount of money collected at the gate.

- 1 What are the variables needed in the program?
- 2 Select an appropriate data type for each variable and constant.

**COMMAND SEQUENCE,
SELECTION AND ITERATION**

In Unit 1 you learnt that the three key building blocks of algorithms are command sequence, selection and iteration. In this unit, you will have the opportunity to implement these constructs in the high-level programming language you are studying.

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
CREATIVITY

ACTIVITY 3**UNDERSTANDING ALGORITHMS**

Read the following algorithm written in pseudocode and then answer the questions below.

```
RECEIVE number1 FROM (INTEGER) KEYBOARD
RECEIVE number2 FROM (INTEGER) KEYBOARD
SET result1 TO number1 / number2
SEND result1 TO DISPLAY
SET result2 TO number1 MOD number2
SEND result2 TO DISPLAY
SET result3 TO number1 DIV number2
SEND result3 TO DISPLAY
```

- 1 What does this algorithm do?
- 2 What is the output of the algorithm, given the following inputs:
 - a 4, 2
 - b 10, 3
 - c 20, 6?

Implement this algorithm in the high-level language you are studying.

SELECTION

The selection construct is used to create a branch in a program. The computer selects which branch to follow based on the outcome of a condition, using an **IF...THEN...ELSE** statement. For example, in pseudocode:

```
IF day = 'Saturday' OR day = 'Sunday' THEN
    SET alarm TO 11
ELSE
    SET alarm TO 8
END IF
```

SUBJECT VOCABULARY

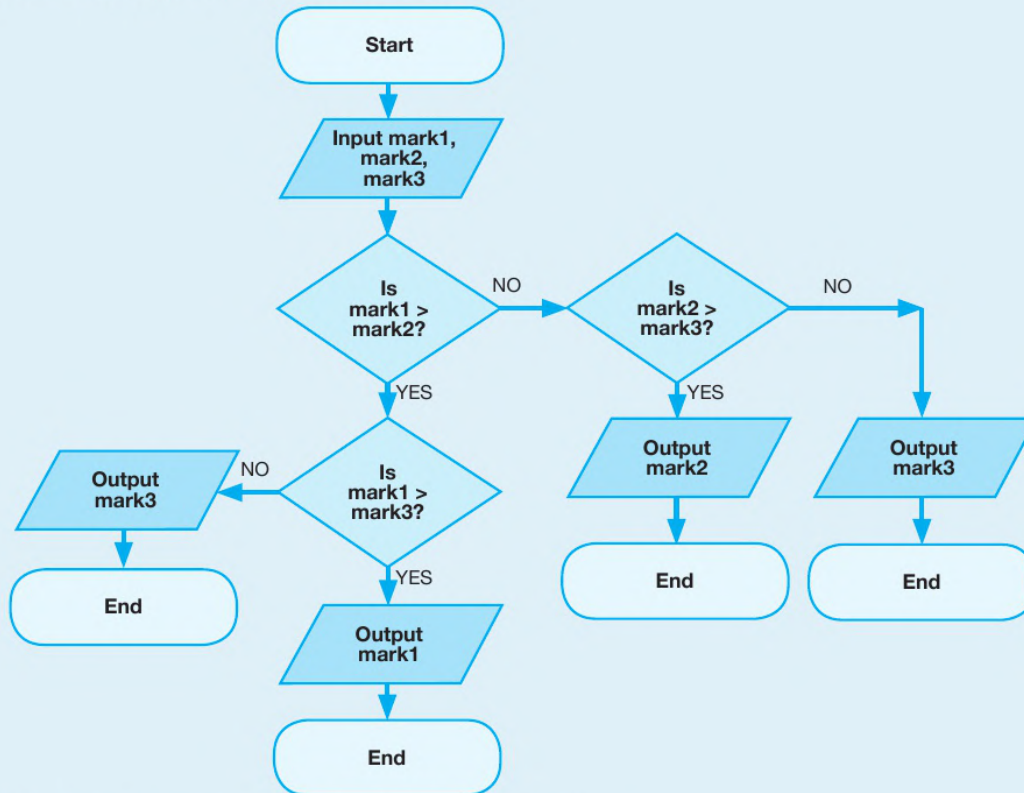
nested IF statement a nested IF statement consists of one or more IF statements placed inside each other. A nested IF is used where there are more than two possible courses of action

A standard **IF...THEN...ELSE** statement provides two alternatives. If there are more than two, then in Pearson Edexcel pseudocode a nested **IF** must be used. However, many high-level programming languages have an additional built-in selection construct that does away with the need for a **nested IF statement**.

WORKED EXAMPLE

A learner handed in three homework assignments, which were each given a mark out of 10. All the marks were different. Write an algorithm that would print out the highest mark.

Figure 2.1 shows the algorithm expressed as a flowchart.



▲ Figure 2.1 Flowchart of an algorithm to print out the highest homework mark

HINT

When you are creating nested **IF** statements, you have to ensure that each one is completed with an **END IF** statement at the correct indentation level. Some programming languages do not need an **END IF** statement and just use the indentation levels to indicate when statements are grouped.

In Pearson Edexcel pseudocode, this algorithm could be expressed as:

```

RECEIVE mark1 FROM KEYBOARD
RECEIVE mark2 FROM KEYBOARD
RECEIVE mark3 FROM KEYBOARD
IF mark1 > mark2 AND mark1 > mark3 THEN
    SEND mark1 TO DISPLAY
ELSE
    IF mark2 > mark1 AND mark2 > mark3 THEN
        SEND mark2 TO DISPLAY
    ELSE
        IF mark3 > mark1 AND mark3 > mark2 THEN
            Send mark3 TO DISPLAY
        END IF
    END IF
END IF
END IF
  
```


In Python, Java and C# this does not have to be done as they have an 'else if' statement.

In Python the 'else if' statement is `elif` and the algorithm above could be:

```
mark1 = input('Please enter the first mark')
mark2 = input('Please enter the second mark')
mark3 = input('Please enter the third mark')

if mark1 > mark2 and mark1 > mark3:
    print(mark1)
elif mark2 > mark1 and mark2 > mark3:
    print(mark2)
elif mark3 > mark1 and mark3 > mark2:
    print(mark3)
```

Here is the algorithm in Java:

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Please enter the first mark: ");
        int mark1 = scanner.nextInt();
        System.out.print("Please enter the second mark: ");
        int mark2 = scanner.nextInt();
        System.out.print("Please enter the third mark: ");
        int mark3 = scanner.nextInt();
        scanner.close();

        if (mark1 > mark2 && mark1 > mark3) {
            System.out.print(mark1);
        } else if (mark2 > mark1 && mark2 > mark3) {
            System.out.print(mark2);
        } else if (mark3 > mark1 && mark3 > mark2) {
            System.out.print(mark3);
        }
    }
}
```

And in C#:

```
string mark1String, mark2String, mark3String;
int mark1, mark2, mark3;

Console.WriteLine("Please enter the first mark");
mark1String = Console.ReadLine(); // read input into string
mark1 = int.Parse(mark1String); // convert string to integer

Console.WriteLine("Please enter the second mark");
mark2String = Console.ReadLine(); // read input into string
mark2 = int.Parse(mark2String); // convert string to integer
```

```

Console.WriteLine("Please enter the third mark");
mark3String = Console.ReadLine(); // read input into string
mark3 = int.Parse(mark3String); // convert string to integer

if (mark1 > mark2 && mark1 > mark3)
{
    Console.WriteLine(mark1);
}
else if (mark2 > mark1 && mark2 > mark3)
{
    Console.WriteLine(mark2);
}
else if (mark3 > mark1 && mark3 > mark2)
{
    Console.WriteLine(mark3);
}

```

These examples use **relational** and **logical operators**.

SUBJECT VOCABULARY

relational operator an operator that tests the relationship between two entities

logical operator a Boolean operator using AND, OR and NOT

RELATIONAL OPERATORS

The relational operators are used to compare two values and in Python, Java and C# are all the same.

RELATIONAL OPERATOR	PYTHON, JAVA, C#
Equal to	==
Greater than	>
Greater than or equal to	>=
Less than	<
Less than or equal to	<=
Not equal to	!=

ACTIVITY 4

Look at the following algorithm and answer the questions.

```

IF score <= highScore THEN
    SEND 'You haven't beaten your high score.' TO DISPLAY
ELSE
    SEND 'You've exceeded your high score!' TO DISPLAY
END IF

```

What is the output of the algorithm when

- score = 5 and highScore = 10?
- score = 20 and highScore = 10?
- score = 15 and highScore = 15?

LOGICAL OPERATORS

AND

If two conditions are joined by the **AND** operator, then they must both be true for the whole statement to be true.

OR

If two conditions are joined by the **OR** operator, then either one must be true for the whole statement to be true.

NOT

The **NOT** operator reverses the logic of the **AND** and **OR** statements. The statement **IF A = 3 AND B = 6** will be true only if the conditions are met, i.e. A and B are both equal to the values stated.

The statement **IF NOT (A = 3 AND B = 6)** will be true whenever both A and B are NOT equal to the values stated, i.e. either or both are not equal to those values.

Logical operators in high-level languages.

LOGICAL OPERATOR	PYTHON	JAVA	C#
AND	and	&	&&
OR	or		
NOT	not	!	!

ACTIVITY 5

A driving school uses this rule to estimate how many lessons a learner will require.

- Every learner requires at least 20 lessons.
- Learners over the age of 18 require more lessons (two additional lessons for each year over 18).

Create a program in a high-level language that inputs a learner's age and calculates the number of driving lessons they will need.

LOOPS

A loop is another name for an iteration. Loops are used to make a computer repeat a set of instructions more than once. There are two types of loop: definite and indefinite.

A definite loop is used when you know in advance how often the instructions in the body of the loop are to be repeated. For example, if you want the computer to display a character on the screen for a fixed amount of time and then remove it.

An indefinite loop is used when the number of times a loop will need to be repeated is not known in advance. For example, if you want to give a user the option of playing a game as often as they want. Indefinite loops are repeated until a specified condition is reached.

Every programming language has a number of built-in loop constructs. You will need to explore the ones provided in the language you are studying.

DEFINITE ITERATION

This is used when the number of iterations, or turns of the loop, is known in advance.

In the Pearson Edexcel pseudocode there are two ways of doing this using **REPEAT...END REPEAT** and **FOR...END FOR**.

An example of a **REPEAT** loop is shown below.

```
REPEAT 50 TIMES
    SEND '*' TO DISPLAY
END REPEAT
```

Using a **FOR** loop, this would be:

```
FOR times FROM 1 TO 100 DO
    SEND '*' TO DISPLAY
END FOR
```

FOR loops can also include a step so that the counting is not consecutive. A step is included in the following pseudocode example.

```
FOR times FROM 0 TO 100 STEP 25 DO
    SEND times TO DISPLAY
END FOR
```

The output would be 0, 25, 50, 75.

USE	EXAMPLE	RESULT
Using the 'range' command	<pre>for x in range(6): print(x)</pre>	0 1 2 3 4 5
Stipulating a start number so that it does not begin at 0	<pre>for x in range(2,6): print(x)</pre>	2 3 4 5
Using a step by adding a third number into the brackets	<pre>for x in range(0,100,25): print(x)</pre> To also print 100 you would use: <pre>for x in range(0,101,25):</pre>	0 25 50 75
Printing items from a list	<pre>aList = ['red', 'blue', 'green', 'yellow', 'purple', 'orange'] for colour in aList: print (colour)</pre> The number of iterations does not have to be given as the length of the list is used.	red blue green yellow purple orange

(continued)

(continued)

USE	EXAMPLE	RESULT
Leaving out items from the list	<pre>aList = ['red', 'blue', 'green', 'yellow', 'purple', 'orange'] for colour in aList: if colour == 'yellow': continue print (colour)</pre>	red blue green purple orange

▲ Table 2.2 Python

USE	EXAMPLE	RESULT
Using a for loop	<pre>for(int x = 0; x < 6; x++) { System.out.println(x); }</pre>	0 1 2 3 4 5
Stipulating a start number so that it does not begin at 0	<pre>for(int x = 2; x < 6; x++) { System.out.println(x); }</pre>	2 3 4 5
Using a step by adding an amount to increase by	<pre>for(int x = 0; x < 100; x+=25) { System.out.println(x); }</pre> To also print 100 you would use: <pre>for(int x = 0; x <= 100; x+=25) { System.out.println(x); }</pre>	0 25 50 75
Printing items from a list	<pre>String[] aList = {"red", "blue", "green", "yellow", "purple", "orange"}; for(String colour: aList) { System.out.println(colour); }</pre>	red blue green yellow purple orange
Leaving out items from the list	<pre>String[] aList = {"red", "blue", "green", "yellow", "purple", "orange"}; for(String colour: aList) { if(colour == "yellow") { continue; } System.out.println(colour); }</pre>	red blue green purple orange

▲ Table 2.3 Java

USE	EXAMPLE	RESULT
Using a for loop	<pre>for (int i = 0; i <= 5; i++) { Console.WriteLine(i); }</pre>	0 1 2 3 4 5
Stipulating a start number so that it does not begin at 0	<pre>for (int i = 2; i <= 5; i++) { Console.WriteLine(i); }</pre>	2 3 4 5
Using a step by altering the increment value of a for loop	<pre>for (int i = 0; i < 100; i+= 25) { Console.WriteLine(i); }</pre>	0 25 50 75
Printing items from an array*	<pre>string[] colours = {"red", "blue", "green", "yellow", "purple", "orange"}; foreach (string i in colours) { Console.WriteLine(i); }</pre> The number of iterations does not have to be given as the length of the array is used.	red blue green yellow purple orange
Leaving out items from the array*	<pre>string[] colours = {"red", "blue", "green", "yellow", "purple", "orange"}; foreach (string i in colours) { if (i == "yellow") { continue; } else { Console.WriteLine(i); } }</pre>	red blue green purple orange

*See pages 62–68 for more information on arrays.

▲ Table 2.4 C#

SKILLS PROBLEM SOLVING

ACTIVITY 6

Produce a program in a high-level language that asks a user to enter a start number and an end number and then outputs the total of all the numbers in the range. For example, if the start number was 1 and the end number was 10, the total would be 55 ($10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1$).

NESTED LOOPS

A **nested loop** is made of a loop within a loop. When one loop is nested within another, each iteration of the outer loop causes the inner loop to be executed until completion.

HINT

You should initialise the variable total to zero before the start of the loop.

ACTIVITY 7**1 Python**

Look at the following program and then answer the questions below.

```
for student in range(1, 21):
    sum = 0
    for mark in range(1, 6):
        nextMark = int(input('Please enter a mark'))
        sum = sum + nextMark
    averageMark = sum/5
    print(averageMark)
```

- a What is the purpose of this program?
- b Why is `int` used in the line `nextMark = int(input('Please enter a mark'))`?

2 Java

Look at the following program and then answer the questions below.

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        for(int student = 1; student < 21; student++) {
            int sum = 0;
            for(int mark = 1; mark < 6; mark++) {
                System.out.print("Please enter a mark: ");
                int nextMark = scanner.nextInt();
                sum = sum + nextMark;
            }
            double averageMark = sum / 5;
            System.out.println(averageMark);
        }
        scanner.close();
    }
}
```

- a What is the purpose of this program?

3 C#

```

int nextMark, sum, averageMark;
string markString;
sum = 0;
for (int mark = 1; mark <= 5; mark++)
{
    Console.WriteLine("Please enter a mark");
    markString = Console.ReadLine();
    nextMark = int.Parse(markString);
    sum = sum + nextMark;
}

averageMark = sum / 5;
Console.WriteLine(averageMark);

```

- a What is the purpose of the program?
- b Why is `int.Parse` used in the line `nextMark = int.Parse(markString);`?

SKILLS **PROBLEM SOLVING****ACTIVITY 8**

Produce an algorithm that will print out the times tables (up to 12 times) for the numbers 2 to 12.

INDEFINITE ITERATION

An indefinite loop is used when the number of times a loop will need to be repeated is not known in advance. For example, if you want to give a user the option of playing a game as often as they want. Indefinite loops are repeated until a specified condition is reached.

Python

For indefinite iteration, Python uses the 'while' loop – something is done while a condition is met.

The following program asks a user to enter a number while the number is less than 20.

```

number = 1
while number <= 20:
    number = int(input('Please enter a number'))
print('You entered a number greater than 20')

```

As soon as the number is greater than 20, the program breaks out of the loop and prints a message for the user.

Java

For indefinite iteration, Java uses a 'while' loop – instructions are repeated while a condition is met.

The following program asks a user to enter a number while the number is less than 20.

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int number = 1;
        while(number <= 20) {
            System.out.print("Please enter a number: ")
            number = scanner.nextInt();
        }
        System.out.println("You entered a number greater than 20")
        scanner.close();
    }
}
```

C#

```
string numberString;
int number = 1;
while (number <=20)
{
    Console.WriteLine("Please enter a number");
    numberString = Console.ReadLine();
    number = int.Parse(numberString);
}

Console.WriteLine("You entered a number greater than 20");
```

As soon as the number is greater than 20, the program breaks out of the loop and prints a message for the user.

ACTIVITY 9

What do these two algorithms do? Implement them in the high-level programming language you are studying.

a Algorithm A

```
FOR index FROM 1 TO 10 DO
    SEND index * index * index TO DISPLAY
END FOR
```

b Algorithm B

```
SET counter TO 10
WHILE counter > 0 DO
```

```
SEND counter TO DISPLAY
SET counter TO counter - 1
END WHILE
```

RANDOM NUMBERS

Random numbers are commonly used in games of chance such as flipping a coin or rolling a dice. The aim is to make an event random.

All high-level programming languages have functions to create random numbers.

Python

Python has a random module. The following code will generate a random number between 1 and 10.

```
import random
x = random.randint(1, 11)
print(x)
```

The 'import' command is necessary so that the 'random' module can be used.

Java

Java can generate a random integer using `System.util.Random`. The following code will generate a random number between 1 and 10.

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Random r = new Random();
        int x = r.nextInt(10) + 1;
        System.out.println(x);
    }
}
```

Alternatively, you can generate a random real value using `Math.random()`.

```
class Main {
    public static void main(String[] args) {
        int x = (int)(Math.random() * 10) + 1;
        System.out.println(x);
    }
}
```

C#

C# can generate random numbers using the Random Class. The following code will generate a random number between 1 and 10 and display on screen.

```
Random rand = new Random();
randomNumber = rand.Next(1, 11);
Console.WriteLine(randomNumber);
```


ACTIVITY 10

Create a guessing-game program with the following specification.

- The computer generates a random number between 1 and 20.
- The user is asked to enter a number until they enter this random number.
- If their guess is too low or too high they are told.
- They are told when their guess is correct.
- They are asked if they want to play another game until their answer is 'NO'.

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS PROBLEM SOLVING

CHECKPOINT

Strengthen

S1 Why are variables needed?

S2 Provide examples of the four data types.

S3 How are selection and iteration implemented in the high-level language you are studying?

Challenge

C1 Outline the following structural components of a program: variable and type declarations, command sequences, selection and iteration constructs.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try redoing the activities in this section.

SUMMARY

- A program is an algorithm that has been converted into program code.
- Pseudocode is far more forgiving than program code.
- The four basic data types are integer, float/real, Boolean and character.
- The data type of a variable determines the operations that can be performed on it.
- Data types don't have to be declared in pseudocode but it's a good idea to do so.
- Variable and type declarations, command sequences, selection and iteration are four of the structural components of a program.

6 MAKING PROGRAMS EASY TO READ

Developers usually work in teams and it is important that they understand how each other's code and programs work, especially if there are errors that need correcting. Code should be written in standard ways and be explained using comments.

LEARNING OBJECTIVES

- Explain the benefit of producing programs that are easy to read
- Use techniques to improve the readability of code and describe how code works

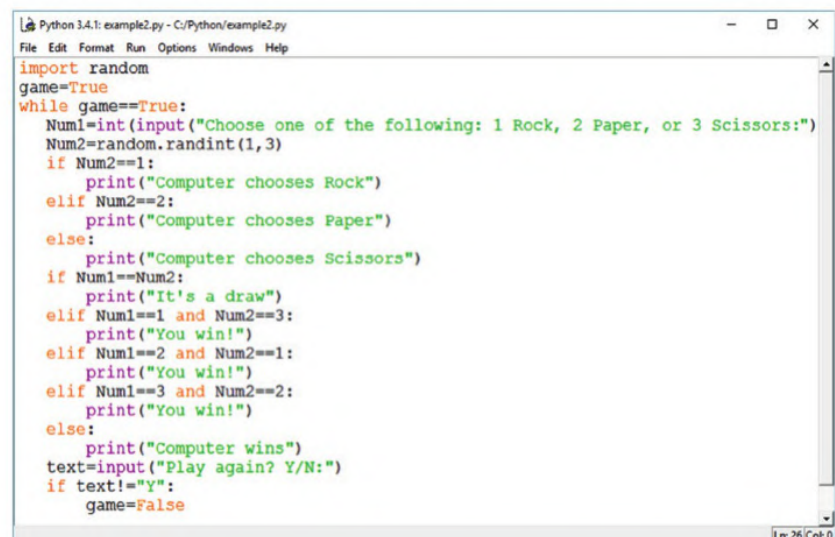
CODE READABILITY

You should always try to ensure that any code you write is easy to read and understand. We refer to this as 'readability'. This benefits you and anyone else who needs to understand how your programs work.

It is surprising how quickly you forget. Try revisiting the programs you have already written in this unit and make sure it is still clear to you what they do and how they work. Imagine how much more difficult it would be to make sense of a complex program with lots of variables, subprograms, nested loops and multiple selection statements.

Programming is not a solo activity. Programmers usually work in teams, with each programmer developing a different part of the program. This only works if they all adopt a standard approach to writing readable code.

► Figure 2.2 An example of Python code



```
Python 3.4.1: example2.py - C:/Python/example2.py
File Edit Format Run Options Windows Help
import random
game=True
while game==True:
    Num1=int(input("Choose one of the following: 1 Rock, 2 Paper, or 3 Scissors:"))
    Num2=random.randint(1,3)
    if Num2==1:
        print("Computer chooses Rock")
    elif Num2==2:
        print("Computer chooses Paper")
    else:
        print("Computer chooses Scissors")
    if Num1==Num2:
        print("It's a draw")
    elif Num1==1 and Num2==3:
        print("You win!")
    elif Num1==2 and Num2==1:
        print("You win!")
    elif Num1==3 and Num2==2:
        print("You win!")
    else:
        print("Computer wins")
    text=input("Play again? Y/N:")
    if text!="Y":
        game=False
```

GENERAL VOCABULARY

practice the way that you do something

SUBJECT VOCABULARY

block of code a grouping of two or more code statements

The programmer who produced the program in Figure 2.2 did not follow good **practice**. There are a number of ways that the readability of this code could be improved.

- Use descriptive names for variables (e.g. `userChoice` instead of `Num1`).
- Add blank lines between different **blocks of code** to make them stand out.
- Add comments that explain what each part of the code does.

HINT

Here's how to write comments in Python, Java and C#:

- **Python:** start each line of a comment with the # symbol
- **Java:** start single line comment with // or start a multi-line comment with /* and end it with */
- **C#:** Same as for Java.



▲ Programmers often work in teams; it is vital that any coding they share is clear and error free

Table 2.5 lists the techniques you should use to make your programs easy to read and understand.

DID YOU KNOW?

Microsoft Windows® uses roughly 50 million lines of code – code readability obviously becomes more and more important as programs get larger.

TECHNIQUE	DESCRIPTION
Comments	Comments should be used to explain what each part of the program does.
Descriptive names	Using descriptive identifiers for variables, constants and subprograms helps to make their purpose clear.
Indentation	Indentation makes it easier to see where each block of code starts and finishes. Getting the indentation wrong in Python will result in the program not running or not producing the expected outcomes.
White space	Adding blank lines between different blocks of code makes them stand out.

▲ Table 2.5 Techniques for program clarity

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
CREATIVITY

ACTIVITY 11

REWRITING AND IMPLEMENTING ALGORITHMS

- 1 Program the following algorithm in a high-level language and make it readable by adding comments and indentation.

```

SET x TO 10
WHILE x >= 0 DO
    IF x > 0
        SEND x TO DISPLAY
    ELSE
        SEND 'Blast Off' TO DISPLAY
    END IF
    SET x TO x - 1
END WHILE
  
```



▲ Calculators have a number of useful functions

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS PROBLEM SOLVING

- 2 Develop an algorithm for a simple calculator and code it in a high-level language that:
 - a allows the user to choose from these options: addition, subtraction, division and multiplication
 - b prompts the user to input two numbers
 - c performs the calculation and displays the result
 - d offers the user the option of performing another calculation.

CHECKPOINT

Strengthen

S1 Why it is important to make your code easy to read?

S2 Outline the four techniques that a programmer should use to make code easy to read.

Challenge

C1 Revisit the programs you have already written. Do you still understand what they do and how they work? If not, try to improve their readability.

KEY POINT

Readability of code is important. You should always make your code easy to read.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, look again at Table 2.5.

SUMMARY

- Using comments, descriptive names, indentation and white space makes code easier to read.
- Producing readable code makes it easier to understand what a program does and how it does it.

7 STRINGS

A string is a data type that is used to represent a sequence of characters, such as numbers, text, spaces and punctuation. Strings must be enclosed in quotation marks to distinguish them from variable names. A string could be a user name, a whole sentence or paragraph of data.

LEARNING OBJECTIVES

- Describe what a string is and explain what strings are used for
- Use iteration to traverse a string
- Concatenate and split strings

SUBJECT VOCABULARY

string a sequence of characters. They can be letters, numbers, symbols, punctuation marks or spaces

A character is one of the four basic data types. It can be a single letter, a symbol, a number or even a space. A sequence of characters is called a **string**. Although strings can contain different sorts of characters, including numbers, they are all treated as if they were text.

When a computer executes a program, it needs a way of telling the difference between a string and an instruction. In most programming languages, this is achieved by enclosing strings in quotation marks (e.g. 'johnsmith@mail.com', '10/04/15' or '123'). Both single ' ' and double " " quotation marks are acceptable, as long as they are used in the same way each time. In the example, "It's her book", the apostrophe is treated as part of the string because double quotes have been used to enclose the string.

Strings are very useful when communicating with users. For example, asking them to enter some information into a program or displaying the output of a program in a format that humans can read and understand.

STRING INDEXING

GENERAL VOCABULARY

reference to refer to something

Each character in a string has an index number, with the first character at position 0. You can use the index to **reference** individual characters in a string.

Therefore the index position of the letter 'm' is 2, even though it is the third character, as shown in Table 2.6.

INDEX	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
String	C	o	m	p	u	t	e	r		S	c	i	e	n	c	e

▲ Table 2.6 An example string index

KEY POINT

Computers start counting at 0. Therefore, although the length of the string 'Computer Science' is 16, the indexes of the characters range from 0 to 15.

LENGTH

SUBJECT VOCABULARY

function a subprogram that performs a specific task and returns a value to the main program. High-level programming languages have a number of useful built-in functions. You can also create your own or use functions available in online libraries

The Pearson Edexcel pseudocode has a built-in **LENGTH function**, which you can use to find the number of characters in a string. Therefore:

```
SET numChars TO LENGTH(myText)
SEND numChars TO DISPLAY
```

would print '16'. Table 2.7 gives the method for Python, Java and C#.

LANGUAGE	METHOD	EXAMPLE	OUTPUT
Python	The function to find the length of a string is <code>len()</code> .	<pre>string = 'Computer Science' length = len(string) print(length)</pre>	16
Java	Each string object has a <code>.length()</code> method.	<pre>String s = "Computer Science"; int length = s.length(); System.out.println(length);</pre>	16
C#	The length of a string can be found by accessing the length property.	<pre>string subject = "Computer Science"; Console.WriteLine(subject.Length);</pre>	16

▲ Table 2.7 Finding the length of a string in Python, Java and C#.

SKILLS

PROBLEM SOLVING,
CREATIVITY,
DECISION MAKING

ACTIVITY 12

CHECKING PASSWORD LENGTH

Create and write a program to check the length of a password. If the password entered is less than six characters, the program should output 'The password you have entered is not long enough'; otherwise it should output 'Length of password OK'.

STRING TRAVERSAL

SUBJECT VOCABULARY

string traversal using a loop to cycle through each character in a string

You can use a **FOR** loop to cycle through each of the characters in a string. This is known as **string traversal**.

The following algorithm prints out the word 'monkey' letter by letter, displaying each letter on a separate line.

```
SET animalName TO 'monkey'
FOR index = 0 TO LENGTH(animalName) - 1
    SEND animalName[index] TO DISPLAY
END FOR
```

The loop runs until `LENGTH(animalName) - 1` because string indexing starts at 0 and therefore has to print characters with indexes of 0 to 5.

PYTHON

```
animalName = 'monkey'
for index in range(0, len(animalName)):
    print(animalName[index])
```


Using Python you should not use `len(animals) - 1` as the range command does not include the end number of the loop.

JAVA

```
class Main {
    public static void main(String[] args) {
        String animalName = "monkey";
        for(int index = 0; index < animalName.length(); index++) {
            System.out.println(animalName.charAt(index));
        }
    }
}
```

C#

```
string animalName = "monkey";

for (int index = 0; index < animalName.Length; index++)
{
    Console.WriteLine(animalName[index]);
}
```

SKILLS

PROBLEM SOLVING,
CREATIVITY,
DECISION MAKING

ACTIVITY 13

WHICH CAR?

Write a program that will check if a make of car entered by the user is in the string 'The cars present included Ford, Mercedes, Toyota, BMW, Audi and Renault.'

If the car entered by the user is present, then 'It is present' should be returned or 'It is not present', if not.

It should not matter which case the car name is entered by the user.

OTHER WAYS TO MANIPULATE STRINGS

FINDING A CHARACTER WITH A PARTICULAR INDEX

The index of a character is the position of that character within a string. Note that indexing starts at 0 not 1. This means the first character will have an index of 0 and the second character will have an index of 1.

PYTHON

Python uses square brackets to access elements in a string:

```
string = 'Computer Science'
print(string[11])
```

This code would return the character 'c'.

JAVA

Java allows you to get a character at a particular index using the `charAt` method.

```
class Main {  
    public static void main(String[] args) {  
        String string = "Computer Science";  
        System.out.println(string.charAt(11));  
    }  
}
```

This code would display the character 'i' (the 12th character at index 11).

C#

```
string text;  
text = "Computer Science";  
Console.WriteLine(text[3]);
```

This code would return the character 'p'.

CHANGING ALL CHARACTERS TO LOWER CASE**PYTHON**

```
string = 'Computer Science'  
string = string.lower()  
print(string)
```

This code would return 'computer science'.

JAVA

```
class Main {  
    public static void main(String[] args) {  
        String string = "Computer Science";  
        string = string.toLowerCase();  
        System.out.println(string);  
    }  
}
```

C#

```
string text;  
text = "Computer Science";  
text = text.ToLower();  
Console.WriteLine(text);
```

Would output 'computer science'.

CHANGING ALL CHARACTERS TO UPPER CASE**PYTHON**

```
string = 'Computer Science'  
string = string.upper()  
print(string)
```

This would return 'COMPUTER SCIENCE'.

JAVA

```
class Main {
    public static void main(String[] args) {
        String string = "Computer Science";
        string = string.toUpperCase();
        System.out.println(string);
    }
}
```

C#

```
string text;
text = "Computer Science";
text = text.ToUpper();
Console.WriteLine(text);
```

This would output 'COMPUTER SCIENCE'.

HINT

Changing strings to upper or lower case is useful when you are comparing two strings and the case does not matter, only the characters.

EXTRACTING CHARACTERS FROM A STRING**PYTHON**

```
string = 'Computer Science'
substring = string[3:6]
print(substring)
```

This code would return 'put' as it selects the characters from index 3 to index 5. Index 6 is not included.

JAVA

```
class Main {
    public static void main(String[] args) {
        String string = "Computer Science";
        String substring = string.substring(3,6);
        System.out.println(substring);
    }
}
```

C#

```
string text;
text = "Computer Science";
text = text.Substring(3,4);
Console.WriteLine(text);
```

This would display 'pute' as it selects 4 characters starting at index 3 (remember that the first character is at index 0).

CHECKING A PHRASE IN THE STRING

These examples check to see if the string 'Computer Science' contains the strings 'put' and 'PUT': 'put' is a substring of 'Computer Science' but 'PUT' is not, so these examples will display 'True' then 'False'.

PYTHON

```
string = 'Computer Science'
present = 'put' in string
print(present)
present = 'PUT' in string
print(present)
```

JAVA

```
class Main {
    public static void main(String[] args) {
        String string = "Computer Science";
        boolean present = string.contains("put");
        System.out.println(present);
        present = string.contains("PUT");
        System.out.println(present);
    }
}
```

C#

You can check if a substring is present in a string using the contains method. If the substring is present in the string then 'True' is returned, otherwise 'False' is returned.

```
string text;
bool found = false;
text = "Computer Science";
found = text.Contains("put");
Console.WriteLine(found);
```

This would display 'True', as 'put' is in 'Computer Science'.

CONCATENATION

Concatenation involves joining two or more items of information together. Concatenating two strings produces a new string object. It is very useful when displaying text on screen.

In Pearson Edexcel pseudocode concatenation is done in the following way.

```
RECEIVE userName FROM (STRING) KEYBOARD
SEND 'Hello' & userName TO DISPLAY
```

Note that literal text is enclosed in speech marks but the variable name is not. In Python, this would be:

```
userName = input('Please enter your username')
print('Hello' + userName)
```

The '+' character is used for concatenation.

In Python, you cannot concatenate strings with numbers. For example, the following would produce an error message.

```
length = 13
print('The length is' + length)
```

To overcome this, the number must be converted to a string:

```
length = 13
print('The length is' + str(length))
```

In Java, you can concatenate strings with most other data types:

```
Scanner scanner = new Scanner(System.in);
System.out.println("Please enter your username:");
String username = scanner.nextLine();
System.out.println("Hello" + username);
```

```
int length = 13;
System.out.print("The length is" + length);
```

In C# this would be:

```
string userName;
Console.WriteLine("Please enter your username:");
userName = Console.ReadLine();
Console.WriteLine("Hello" + userName);
```

In C#, you can concatenate with numbers.

COMPARING STRINGS

Comparing strings in Python and C# can be done using the == operator but in Java you need to use a string object's equals method:

```
String food = "pie";
if(food.equals("pie")) {
    System.out.println("Yum! I like pie");
}
```

SKILLS

PROBLEM SOLVING,
CREATIVITY,
DECISION MAKING

ACTIVITY 14

CONCATENATING AND SLICING STRINGS

A company wants a program to generate usernames for new employees. Each username consists of the first four letters of the employee's last name and the first letter of their first name joined together. If the employee's last name is less than four characters in length a letter 'X' is used to fill in for each of the missing characters. Develop a program that asks the user to input their first and last names and outputs their username.

SKILLS CRITICAL THINKING**SKILLS** PROBLEM SOLVING,
CREATIVITY**SKILLS** CRITICAL THINKING**SKILLS** PROBLEM SOLVING,
CREATIVITY**CHECKPOINT****Strengthen****S1** How are individual characters in a string referenced?**S2** Develop an algorithm that uses a loop to traverse a string.**S3** How are a string and a non-string concatenated?**Challenge****C1** Develop a program that asks the user to input a sentence and then splits it up wherever a space occurs. Each word should then be displayed on a separate line.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, have another go at the activities in this section.

SUMMARY

- A string is a sequence of characters.
- Each character in a string has a unique index value representing its position in the string. The first character in a string has the index value 0.
- High-level programming languages have a built-in length function that finds the length of a string.
- A loop is used to traverse a string, character by character.
- Concatenation is the process of joining two or more strings together.
- Slicing is the process of extracting part of a string.
- String formatting is used to control the way text is displayed on screen.

8 DATA STRUCTURES

Data should be classified and organised into similar types so it can be easily searched and analysed. They are stored together in data structures, such as records and arrays.

SUBJECT VOCABULARY

data structure an organised collection of related elements. Arrays and records are two common data structures used in programming

LEARNING OBJECTIVES

- Describe the structure of one- and two-dimensional arrays and give examples of their use
- Create and use one- and two-dimensional arrays in programs
- Describe the record data structure and explain what it is used for
- Design record structures

A **data structure** is an organised collection of related elements. There are many different data structures that can store multiple data items used in programming. You have already encountered one of them – strings. In this section we will investigate two more: arrays and records.

ARRAYS

SUBJECT VOCABULARY

array a structure that contains many items of data of the same type. The data is indexed so that a particular item of data can be easily found

index a number that identifies each element of an array in Python and Java

An **array** is an organised collection of related values with a single shared identifier. All the elements in an array are the same data type. Each has a unique **index** value denoting its position in the array.

In the Pearson Edexcel pseudocode, an array is initialised using the **SET** command. For example, this statement initialises an array called `firstNames` with four elements, all of string type. The square brackets denote the start and end of the array.

```
SET firstNames TO ['Ashura', 'Bryn', 'Eloise', 'Mei']
```

In most programming languages, arrays are static. A static array has a fixed size and when it is declared the number of items it can hold must be stated.

For example, `array friends [5]`.

As with strings, arrays have a length indicating the number of items. Here are the functions to find the length of an array declared as `myArray`.

PYTHON	JAVA	C#
<code>len(myArray)</code>	<code>myArray.length</code>	<code>myArray.Length</code>

Loops can be used to traverse arrays.

ACTIVITY 15

Declare an array with the elements Ford, Mercedes, Toyota, BMW, Audi and Renault.

In a high-level language, create a program to find the length of the array and then traverse it, printing out each of the items.

PYTHON

In Python, arrays are not commonly used. Instead **lists** are used but they are very similar in the way that they operate. Just like an array, a list is created by adding items, separated by commas, inside square brackets.

Lists are easier to use as they are **dynamic** – they do not have a fixed size and can grow as new elements are added. When they are declared they do not have to be given a size, e.g.:

```
cars = []
```

When adding items to an array the `append` command is used, for example:

```
cars = []  
cars.append('Audi')
```

```
print(cars)
```

would return the following:

```
['Audi']
```

Another advantage of lists is that the items do not have to be of the same data type.

```
cars = []  
cars.append('Audi')  
cars.append(3)  
print(cars)
```

would return:

```
['Audi', 3]
```

There are many functions in Python for manipulating arrays.

Investigate the following:

```
max()  
min()  
slice()
```

JAVA

Arrays are static data structures, which means that you can't change the size of an array to make it bigger or smaller in order to add or remove values.

For example, you can have an array of 5 integers:

```
int[] integers = {1,2,3,4,5};
```

The following is not possible with an array:

```
integers.add(6); or integers.remove(5);
```

In order to add or remove values you need to use a list which is a dynamic data structure.

The most common type of list is an `ArrayList`:

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<Integer> integers = new ArrayList<Integer>();
        integers.add(1);
        integers.add(2);
        integers.add(3);
        integers.add(4);
        integers.add(5);
        integers.add(6);
        integers.remove(0); // remove the value at index 0
                           // (not the value 0)
        System.out.println(integers);
    }
}
```

This example will display [2,3,4,5,6] because the first value will be removed.

Arrays are supported for any data type in Java. For example, you can have an array of strings:

```
String[] colours = {"red", "green", "blue"};
```

Or an array of integers:

```
int[] numbers = {1,2,3,4};
```

However, you can't have an array that contains both strings and integers (or any other mix of data types).

For this, you need to use a vector:

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Vector allSorts = new Vector();
        allSorts.add(1);
        allSorts.add("Two");
        allSorts.add(true);
        System.out.println(allSorts);
    }
}
```

This example will output [1, Two true] because the vector `allSorts` contains an integer, a string and a Boolean value.

C#

C# supports lists and arrays. The code below uses an array to hold the items.

```
string[] make = { "Ford", "Mercedes", "Toyota", "BMW",
    "Audi", "Renault" };

foreach (string currentItem in make)
{
    Console.WriteLine(currentItem);
}
```


SKILLS

CRITICAL
THINKING,
PROBLEM
SOLVING,
CREATIVITY

ACTIVITY 16

DESCRIBING AND IMPLEMENTING A LINEAR SEARCH ALGORITHM

The following code uses a linear search algorithm.

```
SET firstNames TO ['Ashura', 'Bryn', 'Eloise', 'Mei', 'James', 'Irena']
RECEIVE searchName FROM (STRING) KEYBOARD

SET found TO False
SET index TO 0

WHILE found = False AND index <= (LENGTH(firstNames) - 1) DO
  IF searchName = firstNames[index] THEN
    SET found TO True
  END IF
  SET index to index + 1
END WHILE

IF found = True THEN
  SEND searchName & 'is at position' & index & 'in the list' TO DISPLAY
ELSE
  SEND searchName & 'is not in the list' TO DISPLAY
END IF
```

Implement this algorithm in the high-level programming language you are studying.



MULTIDIMENSIONAL ARRAYS

In the array `cars = ['Ford', 'Mercedes', 'Toyota', 'BMW', 'Audi', 'Renault']` there is only one item at each index position: the name of the manufacturer.

A multidimensional array is an 'array of arrays'; each item at an index is another array.

We declared the above array using this statement, `array cars[6]`.

If we wanted to create a two-dimensional array we could declare it as `array cars[3, 2]` so that there is another array at each index to store two items of information.

Here is an extract from an array named `examResults`. It has three rows, each of which stores a set of four exam results. The mark of 47 is located at `examResults [1, 2]` – second row, third element along – and the value 80 at `examResults [0, 0]` – first row, first column.

	0	1	2	3
0	80	59	34	89
1	31	11	47	64
2	29	56	13	91

▲ Table 2.8 `examResults` two-dimensional array

Each item of data has two indexes. An array to hold this data would be declared as `array[3, 4]`.

If the array was printed it would be `[[80, 59, 34, 89], [31, 11, 47, 64], [29, 56, 13, 91]]`. There are square brackets around each set of results and around the whole array.

In Python, the array would be declared and initialised as:

```
Scores = [[80, 59, 34, 89], [31, 11, 47, 64], [29, 56, 13, 91]]
```

In Java, it would be:

```
int[][] Scores = new int[][] {
    {80, 59, 34, 89},
    {31, 11, 47, 64},
    {29, 56, 13, 91}};
```

In C# the array would be declared and initialised as:

```
int[,] Scores= new int[,] { { 80, 59, 34, 89 }, { 31, 11,
47, 64 }, { 29, 56, 13, 91 } };
```

WORKED EXAMPLE

A teacher has stored the surnames and test scores of a class of students in a two-dimensional array, e.g. `results[['Smith', '69'], ['Jackson', '90']]`, etc. Create a program that would print out the names and test scores of all the students who have scored 50 or over in the test.

Python

```
results = [['Smith', 69], ['Jackson', 90], ['Dubois', 30]]
for index in range(0, len(results)):
    if results[index][1] >= 50:
        print(results[index][0] + str(results[index][1]))
```

In Python each element is written as `results[0][1]` and not `results[0, 1]`.

Java

```
class Main {
    public static void main(String[] args) {
        String[][] results = {
            {"Smith", "69"},
            {"Jackson", "90"},
            {"Dubois", "30"}};
```

```

        for(int index = 0; index < results.length; index++) {
            int score = Integer.parseInt(results[index][1]);
            String name = results[index][0];
            if(score >= 50) {
                System.out.println(name + score);
            }
        }
    }
}

```

In Java, a two-dimensional array has two indexing expressions (e.g., `results[0][1]`). The first index is the row number and the second index is the column number. Both index expressions count from 0. This means `results[0][1]` will get the value in the second column (index 1) of the first row (index 0), which in the above example would be '69'.

C#

```

string[,] results = new string[,] { { "Smith", "69" },
{ "Jackson", "90" }, { "Dubois", "30" } };

```

```

for (int index = 0; index <= results.GetUpperBound(0);
index++)
{
    if (int.Parse(results[index,1]) >= 50)
    {
        Console.WriteLine(results[index,0] + " " +
results[index,1]);
    }
}

```

In C# an array only consists of one data type, hence the student scores are in double quotes. To be able to check if the student scores are above 50, the score in the array must be cast as an integer using `int.Parse`.

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
CREATIVITY

HINT

All the data values stored in an array must be of the same data type. In this case they are all strings. You will have to convert the scores to integers to find the highest. But as Python uses lists this is not necessary. Strings and integers can be combined within the same list.

ACTIVITY 17

USING TWO-DIMENSIONAL ARRAYS

- 1 Develop a program that creates and initialises an array to hold these five sets of marks.
80, 59, 34, 89
31, 11, 47, 64
29, 56, 13, 91
55, 61, 48, 0
75, 78, 81, 91
- 2 Extend your program so that it calculates and displays the highest mark, the lowest mark and the average mark achieved.

ACTIVITY 18

This two-dimensional array holds the highest score achieved by each player in each of the three levels of an online game.

- 1 Develop a program that initialises the array and then searches through it to find the player with the highest score in each of the three levels.

PLAYER	LEVEL	SCORE
Alexis	1	19
Seema	1	29
Seema	2	44
Lois	1	10
Alexis	2	17
Alexis	3	36
Dion	1	23
Emma	1	27
Emma	2	48

▲ Table 2.9 Example of two-dimensional array

RECORDS

SUBJECT VOCABULARY

record a data structure that stores a set of related values of different data types

field an individual element in a record

We have already said that the elements of an array must all be the same data type. In contrast, the **record** data structure stores a set of related values of different data types.

Each element in a record is known as a **field** and is referenced using a field name.

Table 2.10 shows how the record data structure works. Each row of the table holds a set of information about a particular learner (these are the records). Each column stores one item of information about the learner – their learner number, their age, their form, etc. (these are the fields). All the values in a column have the same data type – `learnerNum` and 'age' are integers; `firstName`, `lastName` and 'form' are strings.

Programming languages vary in the way they handle the record data structure.

learnerNum	firstName	lastName	AGE	FORM
1	Isla	Smith	15	10H
2	Shinji	Fujita	14	10B
3	Anita	Khan	15	10A
4	Abdur	Rahman	15	10G

▲ Table 2.10 Example of record data structure

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
CREATIVITY

▲ It is essential to have a way to sort through the vast amounts of recorded music available

SKILLS

CRITICAL THINKING

SKILLS

REASONING

SKILLS

REASONING

SKILLS

REASONING

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING

ACTIVITY 19

USING RECORDS TO STORE MUSIC DETAILS

- 1 A record data structure is to be used to store the details of music albums. Provide the appropriate data type for these fields:
 - a the title of the album
 - b the name of the artist
 - c the year of release
 - d the genre.
- 2 Develop a program that uses a record structure for storing the details of music albums. It must:
 - a have fields for title, artist, year of release and genre
 - b allow the user to input the details of new albums
 - c allow the user to search for an album by name and display its details.

CHECKPOINT

Strengthen

S1 What is the index of the first element in a one-dimensional array?

S2 How does a linear search algorithm find an element in a one-dimensional array?

S3 How is an element stored in a two-dimensional array referenced?

S4 How is a `nested IF` used to traverse a two-dimensional array?

Challenge

C1 Develop a program for a simple address book that uses a two-dimensional array to store a set of names and email addresses, and allows the user to search for a person by name and returns their email address.

C2 Develop a program that uses a two-dimensional array to represent a treasure map consisting of a grid of 4 rows and 4 columns. A random number function should be used to establish the location of the treasure. The user must hunt for the treasure by repeatedly entering the coordinates of squares. The program should tell them when they have found the treasure and help them in their search by indicating how close they are.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try redoing activities 15–19.

SUMMARY

- A data structure is an organised collection of related elements. Arrays and records are common data structures.
- A one-dimensional array is a list of elements, each of which has a unique index value representing its position in the list.
- A two-dimensional array is a matrix of rows and columns. Each element in the array has a unique pair of indices, one to identify the row and one the column in which it is located.
- All the elements in an array have the same data type (Python uses lists).
- A record consists of a collection of fields. The values stored in a record can be of different data types.

9 INPUT/OUTPUT

All program data has to be entered (input) and information is output. The input could be automatic, e.g. from a sensor, but is often provided by human users. It is essential that this input is checked to ensure that it is what could reasonably be expected and falls within a certain range. Incorrect input could cause problems or even catastrophes.

LEARNING OBJECTIVES

- Explain the need for validation
- Write code that validates user input
- Write code that accepts and responds appropriately to user input
- Write code that reads to and writes from a text file

USER INPUT

Most programs require some form of input either from a user or from a file. You already know how to receive user input from a keyboard.

A program can be made much more 'user friendly' by displaying helpful messages informing users of what they are expected to enter and confirming successful input.

VALIDATION

It is important to ensure that data entered by the user is valid, as invalid data can cause a program to behave unexpectedly or even stop altogether. If the data entered into a program is incorrect, the output it produces will also be wrong. This is sometimes called the Garbage In, Garbage Out (GIGO) principle.

Any program that requires data entry should have appropriate forms of **validation** built in. But validation can't guarantee that the data entered is correct. It can only make sure that it is reasonable.

There are a number of different types of validation.

RANGE CHECK

A range check is used to ensure that the data entered is within a specified range. Study this algorithm, written in Pearson Edexcel pseudocode which checks that the number entered is between 1 and 10.

```

BOOLEAN valid
SET validNum TO False
WHILE validNum = False DO
    SEND 'Please enter a number between 1 and 10:' TO
    DISPLAY
    RECEIVE number FROM (INTEGER) KEYBOARD
    IF number >= 1 AND number <= 10 THEN
        SET validNum TO True
    END IF
END WHILE
SEND 'You have entered:' & number TO DISPLAY

```

SUBJECT VOCABULARY

validation to check that the data entered by a user or from a file meets specified requirements

The algorithm uses a Boolean variable named `validNum` as a status flag. It is initially set to False. The `WHILE` loop continues running until `validNum` is equal to True. An `IF` statement determines if the value of `validNum` should be set to True.

SKILLS CRITICAL THINKING, PROBLEM SOLVING



SKILLS CRITICAL THINKING, PROBLEM SOLVING

GENERAL VOCABULARY

predefined already defined before the start of something

ACTIVITY 20

IMPLEMENTING RANGE CHECKS

Implement the range check algorithm in the high-level programming language you are studying.

PRESENCE CHECK

Another type of validation is a presence check. This simply ensures that a value has been entered, preventing the user from leaving an input blank.

This algorithm asks the user to input their name and uses a presence check to ensure they have entered a value. Any value will cause the loop to finish. It will keep asking the user to input their name until they input a value.

```
SET userName TO ' '
WHILE userName = ' ' DO
    RECEIVE userName FROM (STRING) KEYBOARD
END WHILE
SEND 'Hello' & userName TO DISPLAY
```

ACTIVITY 21

IMPLEMENTING A PRESENCE CHECK

Implement the presence check algorithm in the high-level programming language you are studying.

LOOK-UP CHECK

A look-up check is used to test that a value is one of a **predefined** set of acceptable values. The list of acceptable values can be stored in a one-dimensional array.

This algorithm stores a list of valid form names in an array. It compares the form name entered by the user with the values in the array.

```
SET arrayForms TO ['7AXB', '7PDB', '7ARL', '7JEH']
RECEIVE form FROM (STRING) KEYBOARD
SET valid TO False
SET index TO 0
SET length TO LENGTH(arrayForms)
WHILE valid = False AND index < length DO
    IF form = arrayForms[index] THEN
        SET valid TO True
    END IF
    SET index TO index + 1
END WHILE
```

```

IF valid = True THEN
    SEND 'Valid form' TO DISPLAY
ELSE
    SEND 'The form you entered does not exist.' TO DISPLAY
END IF

```

SKILLS CRITICAL THINKING,
PROBLEM SOLVING

ACTIVITY 22

IMPLEMENTING A LOOK-UP CHECK

Implement the look-up check algorithm in the high-level programming language you are studying.

LENGTH CHECK

It is sometimes necessary to check that the length of a value entered falls within a specified range. For example, all UK postcodes are between six and eight characters long, so validation could be used to check that the length of a postcode entered is within this range. Needless to say, that doesn't necessarily mean it is correct.

```

RECEIVE postCode FROM (STRING) KEYBOARD
SET length TO LENGTH(postCode)
IF length >= 6 AND length <= 8 THEN
    SEND 'Valid' TO DISPLAY
ELSE
    SEND 'Invalid' TO DISPLAY
END IF

```

SKILLS CRITICAL THINKING

ACTIVITY 23

IMPLEMENTING A LENGTH CHECK

Implement the length check algorithm in the high-level programming language you are studying.

TESTING VALIDATION RULES

It is important to test your validation rules to ensure they work as expected. You should use:

Normal data – This is data that is within the limits of what should be accepted by the program. For example, a password with seven characters fulfils the validation rule that states that passwords must be between six and eight characters in length.

Boundary data – This is data that is at the outer limits of what should be accepted by the program. For example, if a validation rule specifies that the range of acceptable values is ≥ 75 AND ≤ 100 , then a value of 100 is at the upper limit and a value of 75 at the lower.

GENERAL VOCABULARY

erroneous not correct; based on incorrect information

Erroneous data – This is data that should not be accepted by the program. For example, a value of 0 should not be accepted by either of the validation rules given above.

WORKING WITH TEXT FILES

SUBJECT VOCABULARY

text file a sequence of lines, each of which consists of a sequence of characters

The programs you have created so far haven't required a huge amount of data entry, but imagine typing a set of test results for everyone in your computer science class. It would take a considerable amount of time to do and – even worse – when the program terminates, all of the data will be lost. Should you need to use it again you'd have to re-enter it. This is where storing data in an external file comes in really useful. If the data you enter is stored in an external **text file** you can access it as often as you like without having to do any further keying in.

Commas are used to separate individual values on a line and a special character is used to denote the end of a line.

Text files provide permanent storage for data. This means that the data can be reused without having to be retyped. Data can be read from, written to and appended to a file.

SUBJECT VOCABULARY

file handle a label that is assigned to a resource needed by the program. It can only access the file through the computer's operating system

overwritten if a file exists on the computer and a new file is created with the same name, the new file is kept and the old file is written over and lost

PYTHON

To access a text file, you must first give it a **file handle**, e.g. `myFile`.

Files can be opened to read from them, to write to them or to append data to them (Table 2.11).

<code>myFile = open('names.txt', 'r')</code>	Open a file named <code>names.txt</code> so that the data can be read.
<code>myFile = open('names.txt', 'w')</code>	Open a file named <code>names.txt</code> to be written to and create a new file if it does not exist. Unfortunately, if the file did exist it would overwrite all of its data.
<code>myFile = open('names.txt', 'a')</code>	Open a file named <code>names.txt</code> to add data to it.

▲ Table 2.11 Using text files

After a file has been opened it must be closed. The data is not written to a file until this command is executed.

The following program would create a text file called 'names' and add two items of data.

```
myFile = open('names.txt', 'w')
myFile.write('First item')
myFile.write('Second item')
myFile.close()
```


The file it creates would be: 'First itemSecond item'.

There is nothing to separate the two items of data. Therefore, it is useful to add a character such as a ',' or a ';' between them.

```
myFile = open('names.txt', 'w')
myFile.write('First item' + ',')
myFile.write('Second item' + ',')
myFile.close()
```

The file created would be: 'First item,Second item'.

It is important to separate the data items so that they can be read back into a data structure.

The following program would write the contents of an array into a text file called `cars.txt` with a comma between them.

```
Alist = ['BMW', 'Toyota', 'Audi', 'Renault', 'Rover']

myFile = open('cars.txt', 'w')

for index in range(len(Alist)):
    myFile.write((Alist[index]) + ',')

myFile.close()
```

The file can be read back into an array in the following way.

```
Blist = []
myFile = open('cars.txt', 'r')

Blist = myFile.read().split(',')

MyFile.close()

Blist = myFile.read().split(',') reads the data and splits it into
separate items where there is a ','.
```

JAVA

Java programs that read and write to files often start with:

```
import system.io.*
```

This allows your Java code to access files such as `File` and `FileReader`, `FileWriter` from the `system.io` namespace.

```
import java.io.*;
class Main {
    public static void main(String[] args) {
        try {
            FileWriter fw = new FileWriter("names.txt");
            fw.write("First item");
            fw.write("Second item");
            fw.close();
```

```

    } catch (IOException e) {
        System.out.println("Could not write to file");
    }
}
}

```

This example will write two strings to the file `names.txt`, which would then contain: `First itemSecond item`.

It is useful to separate each data value in a file with a comma (,) or new line (`\n`) so that they can then be read back into an array or opened as a comma separated value (CSV) file to be edited in a spreadsheet program:

```

import java.io.*;
class Main {
    public static void main(String[] args) {
        try {
            FileWriter fw = new FileWriter("names.txt");
            fw.write("First item,");
            fw.write("Second item\n");
            fw.close();
        } catch (IOException e) {
            System.out.println("Could not write to file");
        }
    }
}

```

Note that the two examples above contain `try...catch` blocks of code. These are necessary in Java because file input or output can often cause runtime errors and Java needs to know what to do instead of crashing the program.

The following program will write the contents of an array of car brands into a file called `cars.txt` with each value separated by a comma:

```

import java.io.*;
class Main {
    public static void main(String[] args) {
        String[] cars = {"BMW", "Toyota", "Audi", "Renault", "Rover"};
        try {
            FileWriter fw = new FileWriter("cars.txt");
            for(int index = 0; index < cars.length; index++) {
                fw.write(cars[index] + ",");
            }
            fw.close();
        } catch (IOException e) {
            System.out.println("Could not write to file");
        }
    }
}

```

The file can be read back into an array in the following way:

```
import java.io.*;
import java.util.*;
class Main {
    public static void main(String[] args) {
        String[] cars;
        try {
            File file = new File("cars.txt");
            Scanner scanner = new Scanner(file);
            String line = scanner.nextLine();

            // split each line into an array of strings
            cars = line.split(",");

            // display each car brand
            for(int index = 0; index < cars.length; index++) {
                System.out.println(cars[index]);
            }

            scanner.close();
        } catch (IOException e) {
            System.out.println("Could not read from file");
        }
    }
}
```

C#

In C#, you're likely to use the `File` class for reading and writing files. To use the `File` class you will need to add the line `Using System.IO;` at the top of your C# program, usually under the existing `Using System;` line. The `File` class provides over 50 different methods for working with files, so this section only provides an introduction to some of the basic methods.

WRITING TO A FILE

C#

The following program creates a file called `"names.txt"` and adds two items of data.

```
StreamWriter sw = new StreamWriter("names.txt");
sw.Write("First Item");
sw.Write("Second Item");
sw.Close();
```

The file it creates would contain: `First itemSecond item`. There is nothing to separate the two items of data. Therefore, it is useful to add a character such as a comma or semicolon between them. The file created would then contain: `First item,Second item`. It is important to separate the data items so that they can be read back into a data structure.

The following program would write the contents of an array into a text file called `cars.csv` with a comma between each item.


```
string[] makes = {"BMW", "Toyota", "Audi", "Renault", "Rover"};
StreamWriter sw = new StreamWriter("cars.csv");

foreach (string currentItem in makes)
{
    sw.Write(currentItem + ",");
}

sw.Close();
```

The file can be read back into an array using:

```
string[] makes = File.ReadAllText("z:cars.txt").Split(',');
```

ACTIVITY 24

A student has coded a computer game which stores the five highest scores in an array. She now wants to save those scores to a file and load them back in when the game is run again.

Write a program that will save the array of scores to a suitable text file and then load them back in again. Run your program to check that it is working as intended.

PYTHON

A two-dimensional array can be stored in a text file in a similar way.

A teacher stores the names of students and their test scores in a two-dimensional array.

```
Slist = [['Faruq', 60], ['Jalila', 90]]
```

The names and scores can be saved in a text file:

```
Slist = [['Faruq', 60], ['Jalila', 90]]
myFile = open('results.txt', 'w')
```

```
for x in range(len(Slist)):
    myFile.write(Slist[x][0] + ',')
    new = str(Slist[x][1])
    myFile.write(new + ',')
```

```
myFile.close()
```

It is easier to copy the items from a text file into a two-dimensional array in two stages.

First, copy them into a one-dimensional array:

```
Blist = []
myFile = open("results.txt", "r")
Blist = myFile.read().split(',')
myFile.close()
```

Second, move them into a two-dimensional array:

```
newList = []
for index in range(0, len(Blist) - 1, 2):
    newList.append([Blist[index], Blist[index + 1]])
```

A step of 2 is used as each record in the two-dimensional array contains 2 items from the one-dimensional array.

JAVA

The following code writes the student test scores to a file:

```
import java.io.*;
import java.util.*;
class Main {
    public static void main(String[] args) {
        // store student scores in a 2d array
        String[][] studentScores = {
            {"Faruq", "60"},
            {"Jalila", "90"}
        };

        // write scores to a file
        try {
            FileWriter fw = new FileWriter("results.txt");
            for(int row = 0; row < studentScores.length; row++) {

                // write student's name
                fw.write(studentScores[row][0] + ",");

                // write student's score
                fw.write(studentScores[row][1] + "\n");
            }
            fw.close();
        } catch (IOException e) {
            System.out.println("Could not write to file");
        }
    }
}
```

The following code reads the test scores back into a two-dimensional array:

```
import java.io.*;
import java.util.*;
class Main {
    public static void main(String[] args) {
        // list of arrays of strings to store results
        ArrayList<String[]> results = new ArrayList<String[]>();
```

```

        // read scores from a file
        try {
            File file = new File("results.txt");
            Scanner scanner = new Scanner(file);

            // read each line from the file
            while(scanner.hasNextLine()) {
                String line = scanner.nextLine();

                // split line into name and score and add it to
                the list
                String[] singleStudent = line.split(",");
                results.add(singleStudent);
            }
            scanner.close();
        } catch (IOException e) {
            System.out.println("Could not read from file");
        }

        // display scores for each student
        for(int i = 0; i < results.size(); i++) {
            System.out.println(results.get(i)[0] + ": " +
            results.get(i)[1]);
        }
    }
}

```

In the above example, a list is used instead of an array because a list is a dynamic data structure. This allows us to add new items as we read them from the file, which we couldn't do with an array.

C#

```

string[,] grades = new string[,] { { "Faruq", "60" }, {
    "Jalila", "90" } };

StreamWriter sw = new StreamWriter("grades.csv");

for (int i = 0; i < grades.GetLength(0); i++)
{
    sw.WriteLine(grades[i, 0] + "," + grades[i, 1]);
}

sw.Close();
}

```


Reading back in to two-dimensional array:

```
// read all of file into array
string[] allLines = System.IO.File.ReadAllLines("grades.csv");
int length = allLines.Length;

string[,] grades;
grades = new string[length, 2];

// for each line in array containing all lines from the
// file, split using delimiter and assign to 2d array
for (int i = 0; i < length; i++)
{
    string[] temp = allLines[i].Split(',');
    grades[i, 0] = temp[0];
    grades[i, 1] = temp[1];
}
```

ACTIVITY 25

In Activity 24, only the high scores were saved. Change your program so that high scores and the names of the players who attained them are stored.

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS PROBLEM SOLVING

SKILLS REASONING

SKILLS PROBLEM SOLVING

CHECKPOINT

Strengthen

- S1** What might happen if a program doesn't include validation on user input?
- S2** Can you think of a data structure that is suitable for storing a list of values used in a look-up check?
- S3** Show how your name, date of birth and favourite colour would be stored in a text file.
- S4** Why is it beneficial to write data to a text file?

Challenge

C1 Develop a program that:

- writes a set of employee records consisting of employee number, name and department to a text file
- reads in the stored records from the text file
- allows the user to search for an employee's details.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try redoing the activities in this section.

SUMMARY

- Validation techniques should be used to ensure that data entered by a user or from a file is valid. They can't guarantee that the data is correct, only that it is reasonable.
- A range check is used to ensure that data is within a specified range.
- A length check is used to ensure that data has a length within a specified range.
- A presence check is used to ensure the user has entered some data.
- When users are required to choose from a list of options, their input should be validated to ensure that their choice is valid.
- Large sets of data are normally stored in text files. The advantage of writing data to a file is that the data is not lost when the program is terminated. It can be read in from the file whenever it is needed.

10 SUBPROGRAMS

When programs are being coded, they should be as structured as possible so that people reading them can quickly understand their logic. One way of doing this is to use self-contained modules, which can be reused where possible. This ensures that programs are shorter, as commands do not have to be repeated several times. These modules can then be called when they are needed.

LEARNING OBJECTIVES

- Describe what a subprogram is
- Explain the benefits of using subprograms
- Explain the difference between global and local variables
- Explain the concept of passing data into and out of subprograms
- Create subprograms that use parameters
- Write code that uses user-defined and pre-existing subprograms

A subprogram is a self-contained module of code that performs a specific task. For example, a high-level programming language could provide a predefined subprogram that calculates the average of a set of numbers. A programmer can use this subprogram in any program they write without needing to know how it works. In other words, subprograms support the process of abstraction.

FUNCTIONS

EXAMPLE FUNCTIONS

A function returns a value to the main program that called it.

```
FUNCTION dice ( )
    simDie = RANDOM(6)
```

This is the start of the function.

This statement uses the predefined function RANDOM to generate a random number between 1 and 6 which is stored in the variable `simDie`.

```
RETURN simDie
```

This returns the value of the variable `simDie` to the main program.

```
END FUNCTION
```

This ends the definition of the function.

This function is called by the main program like this.

```
dieThrow = dice()
```

The value returned by the function is stored in the variable `dieThrow`.

DID YOU KNOW?

In a flowchart, a subprogram is represented by this symbol.



▲ Figure 2.3 The symbol for a subprogram in a flowchart

In high-level programming languages, this would be written as:

PYTHON

```
def dice()
    simDie = random.randint(1, 7)
    return simDie
```

It would be called from the main program by:

```
dieThrow = dice()
```

JAVA

```
import java.util.*;
class Main {
    // define the function
    public static int dice() {
        Random r = new Random();
        return 1 + r.nextInt(6);
    }

    public static void main(String[] args) {
        // call the function
        int dieThrow = dice();
        System.out.println(dieThrow);
    }
}
```

C#

As C# is an **OOP language** it has methods rather than functions. There are a number of ways of implementing methods, but it could be written like this:

```
public static int dice()
{
    Random random = new Random();
    return random.Next(1, 7);
}
```

It would be called from the main program by: `int value = dice();`

SUBJECT VOCABULARY

OOP language an object-oriented programming language. Instead of data structures and separate program structures, both data and program elements are combined into one structure called an object

LOCAL AND GLOBAL VARIABLES

SUBJECT VOCABULARY

local variable a variable that is accessed only from within the subprogram in which it is created

global variable a variable that can be accessed from anywhere in the program, including inside subprograms

Notice that there are two variables that store the random number generated by the function. In the function itself, the variable `simDie` is used. This variable only exists within the function and is referred to as a **local variable**.

In the main program the value returned by the function is stored in the variable `dieThrow`. It can be used anywhere within the main program and is therefore referred to as a **global variable**.

PROCEDURES

Unlike a function, a procedure does not return a value to the main program.

EXAMPLE PROCEDURE

In the dice example, a procedure would be written in pseudocode as:

```
PROCEDURE averageScore(score1, score2, score3)
BEGIN PROCEDURE
    SET total TO score1 + score2 + score3
    SET average TO total / 3
    SEND average TO DISPLAY
END PROCEDURE
```

Python

In Python, it would be exactly the same without the return command.

```
PROCEDURE dice()
    simDie = RANDOM(6)
    SEND simDIE TO DISPLAY
END PROCEDURE
```

Java

Because a procedure doesn't return a value, Java uses the void keyword to indicate that a subprogram doesn't return a value.

```
import java.util.*;
class Main {
    // define the procedure
    public static void dice() {
        Random r = new Random();
        System.out.println(1 + r.nextInt(6));
    }

    public static void main(String[] args) {
        // call the procedure
        dice();
    }
}
```

C#

In C#, the equivalent to a procedure is a method that doesn't return a value. For example:

```
public static void dice()
{
    Random random = new Random();
    int value = random.Next(1, 7);
    Console.WriteLine(value);
}
```

ARGUMENTS AND PARAMETERS

SUBJECT VOCABULARY

parameter the names of the variables that are used in the subroutine to store the data passed from the main program as arguments

Data for the functions and procedures to work on can be passed from the main program as arguments. The function accepts them as **parameters**.

PYTHON

```
# Function rectangle
def rectangle(length, width):
    area = length * width
    return area

#Main program
rectangleLength = int(input('Please enter the length of
the rectangle'))
rectangleWidth = int(input('Please enter the width of the
rectangle'))
rectangleArea = rectangle(rectangleLength, rectangleWidth)
print(rectangleArea)
```

JAVA

```
import java.util.*;
class Main {
    // define the rectangle function
    public static int rectangle(int length, int width) {
        int area = length * width;
        return area;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Please enter the length of the
rectangle: ");
        int rectangleLength = scanner.nextInt();

        System.out.print("Please enter the width of the rectangle: ");
        int rectangleWidth = scanner.nextInt();
        scanner.close();

        int rectangleArea = rectangle(rectangleLength,
rectangleWidth);
        System.out.println(rectangleArea);
    }
}
```

C#

```
public static void Main()
{
    string lengthString, widthString;
    int length, width, rectangleArea;
```



```

Console.WriteLine("Please enter the length of the rectangle");
lengthString = Console.ReadLine();
length = int.Parse(lengthString);

Console.WriteLine("Please enter the width of the rectangle");
widthString = Console.ReadLine();
width = int.Parse(widthString);

rectangleArea = rectangle(length, width);
Console.WriteLine(rectangleArea);
}

public static int rectangle(int length, int width)
{
    int area;
    area = length * width;
    return area;
}

```

In this example, two data items are passed to the function – `rectangleLength` and `rectangleWidth`. These are called arguments.

The function receives them as parameters called `length` and `width` when the function is declared.

SUBJECT VOCABULARY

scope the region of code within which a variable is visible

In the function, a variable is declared – `area`. This is called a local variable and its **scope** is within the function. If you tried to use it in the main program you would get an error message.

Lots of arguments can be passed to the function and many values can be returned.

In the following example, two values are requested and returned.

PYTHON

```

# Function rectangle
def rectangle(length, width):
    area = length * width
    circumference = (2 * length) + (2 * width)
    return area, circumference

#Main program
rectangleLength = int(input('Please enter the length of
the rectangle'))
rectangleWidth = int(input('Please enter the width of the
rectangle'))
rectangleArea, rectangleCircumference = rectangle
(rectangleLength, rectangleWidth)
print(rectangleArea)
print(rectangleCircumference)

```

JAVA

In Java, functions can only return one value so if you need to return more than one value, you need to put them in a list or array.

```
import java.util.*;

class Main {

    // returns an array of the area and perimeter of a rectangle
    public static int[] rectangle(int length, int width) {
        int area = length * width;
        int perimeter = 2 * (length + width);
        return new int[]{area, perimeter};
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Please enter the length of the
rectangle: ");
        int rectangleLength = scanner.nextInt();

        System.out.print("Please enter the width of the
rectangle: ");
        int rectangleWidth = scanner.nextInt();
        scanner.close();

        int[] results = rectangle(rectangleLength,
rectangleWidth);
        int rectangleArea = results[0];
        int rectanglePerimeter = results[1];
        System.out.println("Area: " + rectangleArea);
        System.out.println("Perimeter: " + rectanglePerimeter);
    }
}
```

C#

Methods in C# can't return multiple values unless you use **By Reference**, which is beyond the scope of this book.

ACTIVITY 26

From the above example, list:

- a** the global variables
- b** the local variables
- c** the arguments
- d** the parameters.

SUBPROGRAMS AND MENUS

Functions and procedures are useful when using menus in a program. When a user selects a menu option, they can be sent to a particular function or procedure.

WORKED EXAMPLE

For a system login and password control, the main program could have a menu system like this:

- 1 Register as a new user
- 2 Login
- 3 Change your password
- 4 Exit.

Now that we are using structured programming using procedures, it is relatively easy to direct a user to the part of the program they need.

The user enters a number between 1 and 4 and is directed to the correct section:

Python

```
print ("1. Register as a new user")
print ("2. Login.")
print ("3. Change your password.")
print ("4. Exit.")

choice = int(input('Please select a menu option.'))
if choice == 1:
    newUser()
elif choice == 2:
    login()
elif choice == 3:
    changePassword()
elif choice == 4:
    exit()
else:
    print('Incorrect option. Try again.')
```

Java

```
import java.util.*;
class Main {
    // procedures can be implemented later
    // just shown here to illustrate the structure
    public static void newUser() {}
    public static void login() {}
    public static void changePassword() {}
    public static void exit() {}
    public static void main(String[] args) {
        // display main menu
        System.out.println("1. Register as a new user");
```



```
        System.out.println("2. Login");
        System.out.println("3. Change your password");
        System.out.println("4. Exit");
        // get user input
        Scanner scanner = new Scanner(System.in);
        int choice = scanner.nextInt();

        switch(choice) {
            // Register as a new user
            case 1:
                newUser();
                break;

            // Login
            case 2:
                login();
                break;

            // change password
            case 3:
                changePassword();
                break;

            // exit
            case 4:
                exit();
                break;

            // anything else
            default:
                System.out.println("Incorrect option, try again");
                break;
        }
        scanner.close();
    }
}
```

C#

```
string choiceString;
int choice;

Console.WriteLine("1. Register as a new user");
Console.WriteLine("2. Login");
Console.WriteLine("3. Change your password");
Console.WriteLine("4. Exit");
```

```
Console.WriteLine("Please select a menu option");
choiceString = Console.ReadLine();
choice = int.Parse(choiceString);
if (choice == 1)
{
    newUser();
}
else if (choice == 2)
{
    login();
}
else if (choice == 3)
{
    changePassword();
}
else if (choice == 4)
{
    exit();
}
else
{
    Console.WriteLine("Incorrect option. Try again");
}
```

ACTIVITY 27

- a Describe the purpose of the program in the worked example above and explain how it functions.
- b If the user inputs an incorrect option, they receive an error message and then the program terminates. Edit the program so that the program will run until a suitable option is input.

THE BENEFITS OF USING SUBPROGRAMS

Repeated sections of code need only be written once and called when necessary. This shortens the development time of a program and means that the finished program will occupy less memory space when it is run.

Subprograms also improve the structure of the code, making it easier to read through and follow what is happening. It's less complicated to check your code and debug your program if you use subprograms because each subprogram can be coded, inspected and tested independently. If changes have to be made at a later date it is easier to change a small module than having to work through the whole program.

In large development teams different members can be working independently on different subprograms. They can use and develop standard libraries of subroutines that can be reused in other programs.

BUILT-IN FUNCTIONS**SUBJECT VOCABULARY**

built-in functions functions that are provided in most high-level programming languages to perform common tasks

SKILLS CRITICAL THINKING**SKILLS** REASONING**SKILLS** REASONING**SKILLS** CRITICAL THINKING**SKILLS** PROBLEM SOLVING

In addition to user-written subprograms, most high-level programming languages have a set of **built-in functions** for common tasks. These are designed to save the programmer time, such as functions that print, count the number of characters in a string and generate random numbers.

CHECKPOINT**Strengthen**

S1 What are the benefits of using subprograms?

S2 What is meant by the scope of a variable? Use your own examples.

S3 What happens when a global variable and a local variable share the same name?

S4 Provide an example of a common built-in function.

Challenge

C1 Create and implement a calculator program that:

- allows the user to enter a set of numbers
- uses separate functions to calculate the mean, mode and median
- allows the user to select which function they want
- uses appropriate validation.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try redoing the activities in this section.

SUMMARY

- A subprogram is a section of code within a larger piece of code that performs a specific task. It can be used at any point in the program.
- A function is a subprogram that returns a value to the main program.
- A procedure is a subprogram that does not return a value to the main program.
- Parameters are values that are passed to a subprogram when it is called.
- Local variables can only be accessed from within the subprogram in which they are created.
- Global variables can be accessed anywhere in the program, including inside subprograms.
- Built-in functions are functions provided in a high-level programming language to perform common tasks.

11 TESTING AND EVALUATION

DID YOU KNOW?

The Ariane 5 space rocket was launched by the European Space Agency in 1996. Its flight lasted 39 seconds before it exploded. Data about velocity was stored in 16-bit variable. This had been fine for the Ariane 4 but the 5 model was much faster and the value was too large. This one software error set off a chain of events leading to self-destruction.

However carefully an algorithm has been created and coded into a program, there will always be mistakes and errors. Before a program goes live, it should be thoroughly tested and evaluated to ensure it meets the requirements.

LEARNING OBJECTIVES

- Describe different types of error in programs
- Design and use test plans and test data
- Determine what value a variable will hold at a given point in a program
- Evaluate the strengths and weaknesses of a program and suggest improvements

LOGIC ERRORS

EXTEND YOUR KNOWLEDGE

Research other problems and disasters that have been caused by software failures.

Logic errors occur when the thinking behind an algorithm is incorrect so that the output isn't what is expected or intended. Ideally, logic errors should be identified and fixed at the design stage.

The following algorithm is intended to work out whether a learner has passed a test. Learners need a score of 80 or above to pass. However, a logic error in the algorithm means that it produces an incorrect and unexpected result.

```
IF testScore <= 80 THEN
    SEND 'Pass' TO DISPLAY
ELSE
    SEND 'Failed' TO DISPLAY
END IF
```

DID YOU KNOW?

In computer programming the order of precedence (the order in which you do each calculation) is the same as in mathematics and science – **BIDMAS**.

This is how $3^2 \times 9 + (5 - 2)$ would be evaluated.

Brackets	$32 \times 9 + (3)$
Indices	$9 \times 9 + (3)$
Division	
Multiplication	$81 + (3)$
Addition	84
Subtraction	

To calculate $24/3 - 2$, the division would be calculated before the subtraction.

$24/3 = 8$
 $8 - 2 = 6$

WORKED EXAMPLE

Here is an algorithm to find the average of two numbers.

```
RECEIVE number1 FROM KEYBOARD
RECEIVE number2 FROM KEYBOARD
SET average TO number1 + number2 / 2
SEND average TO DISPLAY
```

This seems logical. Two numbers are input, they are added together and then they are divided by 2.

However, if this algorithm was given 12 and 6 as the two numbers it would return 15 as the average instead of 9. There is a logic error.

Instead of adding the two numbers and then dividing by 2, as the developer intended, it is dividing the second number by 2 and then adding the result to the first number.

The developer should have written the third line as:

```
SET average TO (number1 + number2) / 2
```

SKILLS PROBLEM SOLVING

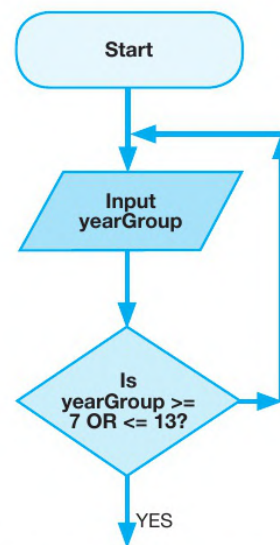
ACTIVITY 28

LOGIC ERRORS

This is part of a larger algorithm designed to ensure that input data falls within a certain range. It is part of a school management system. It checks that the 'year group' entry is acceptable. It has students aged 11 to 18 years with year groups of 7 to 13.

The staff using the system congratulated themselves on never making an error when entering the year group. However, when student lists were printed out, they immediately received complaints.

With a partner, discuss this algorithm and find the logic error in the algorithm?



▲ Figure 2.4 Flowchart showing an error in an algorithm

WORKED EXAMPLE

Study this algorithm.

```

WHILE index < 10 DO
    SET index TO 1
    SEND index TO DISPLAY
    SET index TO index + 1
END WHILE
  
```

The expected output is 1, 2, 3, 4, 5, 6, 7, 8, 9.

But there is a logic error. The variable 'index' is initialised in the wrong place. It should be done before the start of the **WHILE** loop.

This algorithm would loop forever because at each turn in the loop the variable index is set to 1. It will never reach 10. This is an

SUBJECT VOCABULARY

infinite loop a loop that is never-ending since the condition required to **terminate** the loop is never reached

GENERAL VOCABULARY

terminate to cause something to end or stop

SKILLS

REASONING,
PROBLEM SOLVING

example of an '**infinite loop**'. The algorithm should have been written as shown below.

```
SET index TO 1
WHILE index < 10
    SEND index TO DISPLAY
    SET index TO index + 1
END WHILE
```

ACTIVITY 29

FINDING AND CORRECTING ERRORS

Find and correct the errors in these algorithms.

Example 1

```
SET index TO 1
WHILE index < 10
    SEND index TO DISPLAY
END WHILE
```

Example 2

```
SET index TO 1
WHILE index < 10
    SEND index TO DISPLAY
    SET index TO index - 1
END WHILE
```

Example 3

```
SET index TO 1
WHILE index < 1
    SEND index TO DISPLAY
    SET index TO index + 1
END WHILE
```

TRACE TABLES

The formal way of checking the logic of an algorithm is to use a **trace table**.

SUBJECT VOCABULARY

logic error an error in an algorithm that results in incorrect or unexpected behaviour

trace table a technique used to identify any logic errors in algorithms. Each column represents a variable or output and each row a value of that variable

WORKED EXAMPLE

Create a trace table for the following algorithm written in Pearson Edexcel pseudocode.

```
SET number TO 3
FOR index FROM 1 TO 5 DO
    SET number1 TO number * index
    SET number2 TO number1 * 2
    IF number2 > 20 THEN
        SEND number2 TO DISPLAY
    END IF
END FOR
```


This algorithm can be traced as in Table 2.12.

NUMBER	INDEX	NUMBER1	NUMBER2	OUTPUT
3				
3	1	3	6	
3	2	6	12	
3	3	9	18	
3	4	12	24	24
3	5	15	30	30

▲ Table 2.12 Trace table

The value of the variable `number` remains at 3 throughout, but as the index increases from 1 to 5, then so do the values of `number1` and `number2`.

When the value of `number2` is greater than 20, its value is output.

WORKED EXAMPLE

Here is a program.

Python

```
y = 2
for x in range(1, 7):
    y = y + x
print(y)
```

Java

```
class Main {
    public static void main(String[] args) {
        int y = 2;
        for(int x = 1; x < 7; x++) {
            y += x;
        }
        System.out.println(y);
    }
}
```

C#

```

int number = 3;
int number1, number2;

for (int index = 1; index <= 5; index++)
{
    number1 = number * index;
    number2 = number1 * 2;
    if (number2 > 20)
    {
        Console.WriteLine(number2);
    }
}

```

X	Y	OUTPUT	EXPLANATION
1	2		When the loop starts, X becomes 1 and Y already is equal to 2.
2	3		When X is incremented to 2, Y is equal to 3 (2+1) from the previous loop.
3	5		When X is incremented to 3, Y= 3 + 2 from the previous loop.
4	8		Explanations as above.
5	12		
6	17		
6	23	23	The final value of Y is output.

▲ Table 2.13 Trace table of the programs above

ACTIVITY 30

Complete a trace table for this program written in a high-level language.

Python

```

number1 = 2
number2 = 3
for index in range(1, 6)
    number1 = number1 * index
    number2 = number2 + number1

```

Java

```
class Main {
    public static void main(String[] args) {
        int number1 = 2;
        int number2 = 3;
        for(int index = 1; index < 6; index++) {
            number1 = number1 * index;
            number2 = number2 + number1;
        }
    }
}
```

C#

```
int number1 = 2;
int number2 = 3;
for (int index = 1; index < 6; index++)
{
    number1 = number1 * index;
    number2 = number2 + number1;
}
```

SKILLS PROBLEM SOLVING**ACTIVITY 31**

The following program is intended to count the number of female learners in a class.

Python

```
gender = ['M', 'M', 'F', 'M', 'F', 'F', 'M', 'F', 'M', 'F']
length = len(gender)
count = 0
for index in range(length):
    if gender[index] == 'F':
        count = count + 1
```

Java

```
class Main {
    public static void main(String[] args) {
        String[] gender = {"M", "M", "F", "M", "F", "F", "M", "F", "M", "F"};
        int length = gender.length;
        int count = 0;
        for(int index = 0; index < length; index++) {
            if(gender[index].equals("F")) {
                count++;
            }
        }
    }
}
```


C#

```

int length, count;
char[] gender;
gender = new char[] { 'M', 'M', 'F', 'M', 'F', 'F', 'M',
                      'F', 'M', 'F' };

count = 0;
length = gender.Length;
for (int index = 0; index < length; index++)
{
    if (gender[index] == 'F')
    {
        count += 1;
    }
}

```

- 1 Create and complete a trace table for this algorithm.
- 2 Create a trace table for the algorithm below using the following sample data: 3, 12, 21, 28, 0.

Python

```

total = 0
number = int(input('Please enter the number'))
while number > 0:
    total = total + number
    number = int(input('Please enter the number'))
print total

```

Java

```

import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int total = 0;
        System.out.print("Please enter the number: ");
        int number = scanner.nextInt();
        while(number > 0) {
            total += number;
            number = scanner.nextInt();
        }
        System.out.println(total);
    }
}

```

C#

```
int total = 0;
int number;
string numberString;

Console.WriteLine("Please enter the number");
numberString = Console.ReadLine();
number = int.Parse(numberString);

while (number > 0)
{
    total += number;
    Console.WriteLine("Please enter the number");
    numberString = Console.ReadLine();
    number = int.Parse(numberString);
}
Console.WriteLine(total);
```

SYNTAX ERRORS

Syntax errors occur when the grammar rules of a programming language are not followed.

They prevent the code from being compiled or translated.

Examples of syntax errors are:

- writing 'prnit' instead of 'print'
- missing out a closing bracket
- missing out quotation marks.

RUNTIME ERRORS**SUBJECT VOCABULARY**

runtime error an error that occurs while the program is running – the operation the computer is asked to do is impossible to execute

Runtime errors occur during program execution and are the most difficult to predict and spot.

This program is designed to take two numbers, divide the first number by the second number and output the result. It will work as intended at least some of the time. However, if the user entered 5 and 0, a runtime error would occur because it is impossible for the computer to divide 5 by 0.

Python

```
firstNumber = int(input('Please enter the first number'))
secondNumber = int(input('Please enter the second
number'))
result = firstNumber / secondNumber
print(result)
```

Java

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Please enter the first number: ");
        int firstNumber = scanner.nextInt();
        System.out.print("Please enter the second number: ");
        int secondNumber = scanner.nextInt();
        scanner.close();

        // do the division (converting all integers to reals)
        double result = (double)firstNumber / (double)
secondNumber;
        System.out.println(result);
    }
}
```

C#

```
int firstNumber, secondNumber, result;
string firstNumberString, secondNumberString;

Console.WriteLine("Please enter the first number");
firstNumberString = Console.ReadLine();
firstNumber = int.Parse(firstNumberString);

Console.WriteLine("Please enter the second number");
secondNumberString = Console.ReadLine();
secondNumber = int.Parse(secondNumberString);

result = firstNumber / secondNumber;
Console.WriteLine(result);
```

ERROR SUMMARY

This table gives a summary of the three types of error you are likely to encounter.

TYPE OF ERROR	DESCRIPTION
Logic	The program seems to run normally; however, there is an error in the logic of the program, which means it does not produce the result you expect.
Syntax	Syntax refers to the rules of the programming language. A syntax error means that part of the code breaks the rules of the language, which stops it running.
Runtime	An error that occurs when the computer tries to run code that it cannot execute.

▲ Table 2.14 Summary of the three types of error you are likely to encounter

USING AN INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

SUBJECT VOCABULARY

Integrated Development Environment (IDE) a package that helps programmers to develop program code. It has a number of useful tools, including a source code editor and a debugger

debugger a computer program that assists in the detection and correction of errors in other computer programs

You probably already have first-hand experience of using an **Integrated Development Environment (IDE)** when writing code. It's definitely worth taking some time to get to know the IDE that comes with the language you are using. Useful features such as syntax highlighting, code auto complete and auto indent will help to make your programming experience far less stressful, especially at the beginning.

One of the most useful features of an IDE is the **debugger**. One of its tasks is to flag up syntax errors in the code and issue helpful error messages. It is really important that you get lots of practice interpreting error messages and fixing errors in your code.

HINT

Here are some error messages that you are likely to see in popular IDEs.

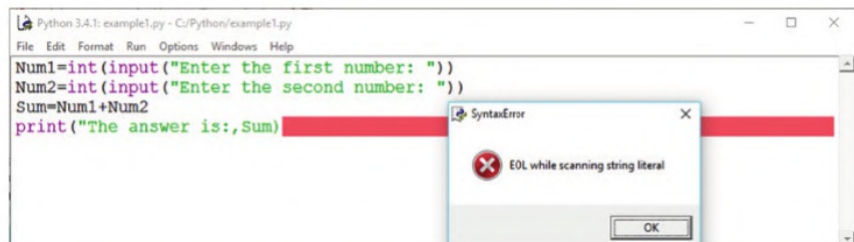
IDLE (a popular Python IDE)

- **SyntaxError**: invalid syntax – *part of the code breaks one of the rules of the programming language.*
- **IndentationError**: expected an indented block – *statements after a colon must be indented.*
- **TypeError**: Can't convert 'int' object to str implicitly – *trying to join a string and an integer together.*
- **ZeroDivisionError**: integer division or modulo by zero – *trying to divide a value by 0.*
- **NameError**: name is not defined – *referring to a variable or subprogram that does not exist.*

Eclipse (a popular Java IDE)

- **;** expected – *each statement should end with a semicolon.*
- Cannot find symbol – *referring to a variable or subprogram that does not exist.*
- Incompatible types – *trying to mix data of different types, for example a string and an integer.*

► **Figure 2.5** A syntax error flagged up by an IDE in Python code



THE TEST PLAN

At the start of a programming project, it is crucial to make a list of the requirements that the program is expected to meet. Throughout the development phase of the project, you should regularly refer back to this list of requirements to check that you are on track to achieve them.

Deciding how to test the finished program to make sure that it fully meets the requirements can't be left to the last moment either. As part of the design stage, you should create a test plan listing the tests you will carry out, the data that will be used for each test and the expected result.

MARKS TO GRADES

```

RECEIVE examMark FROM (INTEGER) KEYBOARD
IF examMark >= 80 THEN
    SEND 'A' TO DISPLAY
ELSE
    IF examMark >= 70 THEN
        SEND 'B' TO DISPLAY
    ELSE
        IF examMark >= 60 THEN
            SEND 'C' TO DISPLAY
        ELSE
            IF examMark > 0 THEN
                SEND 'D' TO DISPLAY
            ELSE
                SEND 'FAIL' TO DISPLAY
            END IF
        END IF
    END IF
END IF

```

This algorithm converts an exam mark into a grade. It should only accept marks between 0 and 100.

The test plan extract in Table 2.15 shows some of the tests that have been planned for the finished program to ensure that it meets all the requirements.

TEST NO.	PURPOSE OF THE TEST	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	ACTION NEEDED/ COMMENTS
1	To check correct conversion of valid mark	0	'FAIL'		
		55	'D'		
		65	'C'		
		75	'B'		
		85	'A'		
2	To check correct conversion of boundary mark	0	'FAIL'		
		1, 59	'D'		
		60, 69	'C'		
		70, 79	'B'		
		80, 100	'A'		
3	To check response to erroneous mark	-5	Error message		
		105	Error message		

▲ Table 2.15 Test plan extract

Only the first four columns of the test plan table can be completed at the design stage. The remaining columns are filled in when the program is complete.

At the start, it's unlikely that you'll know every test that will be needed to ensure the program works as intended. The test plan is not a fixed document. If additional tests are required, they should be added to the test plan.

If a test produces the expected result, you can simply write 'None' in the 'Action needed/comments' column. If, however, that is not the case then you should instead note what went wrong and what you did to put it right.

It is important to select suitable test data for the tests. Test data falls into three different categories: normal, boundary and erroneous.

Normal data	Data that is well within the limits of what should be accepted by the program.	Test 1 uses normal data to check if marks are converted into grades correctly (e.g. a mark of 65 should be converted to a grade C).
Boundary data	Data that is at the outer limits of what should be accepted by the program.	Test 2 uses boundary data to check that the program works correctly with marks at the upper and lower boundaries (e.g. a mark of 60 and a mark of 69 should both convert to a grade C).
Erroneous data	Data that should not be accepted by the program.	Test 3 uses erroneous data to check that the program does not accept it.

▲ Table 2.16 Test data categories

Testing is just as important as writing code because it ensures that the finished program works correctly and fully meets the requirements. You should use a 'bottom up' approach to testing (i.e. test each subprogram as you develop it and then test the whole program once it is finished).

Here is the completed test plan for the grade calculator program. As you can see, the expected result was not produced when the program was tested with erroneous data. A range check had to be added to the program to ensure that only marks between 0 and 100 can be entered.

TEST NO.	PURPOSE OF THE TEST	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	ACTION NEEDED/ COMMENTS
1	To check correct conversion of valid mark	0 55 65 75 85	'FAIL' 'D' 'C' 'B' 'A'	'FAIL' 'D' 'C' 'B' 'A'	None
2	To check correct conversion of boundary mark	0 1, 59 60, 69 70, 79 80, 100	'FAIL' 'D' 'C' 'B' 'A'	'FAIL' 'D' 'C' 'B' 'A'	None
3	To check response to erroneous mark	-5 105	Error message Error message	'FAIL' 'A'	A range check has been added to ensure that only valid marks can be added.

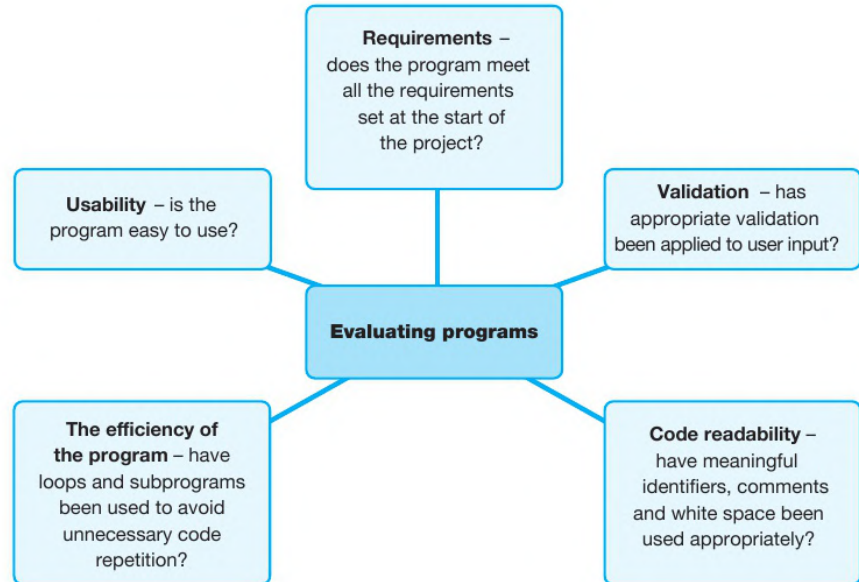
▲ Table 2.17 The completed test plan for the grade calculator program

After you have carried out all of the tests and made all the necessary changes, the program should be retested. This is to ensure that the improvements you have made haven't introduced any new errors into the program.

EVALUATING PROGRAMS

You need to be able to identify the strengths and weaknesses of your own programs as well as those created by other programmers. This will enable you to identify techniques that work well and aspects that could be improved.

► **Figure 2.6** Consider the aspects shown when evaluating a program



SKILLS PROBLEM SOLVING

ACTIVITY 32

CREATING A GUESSING GAME ALGORITHM

- 1 Develop an algorithm for a simple guessing game. The algorithm must:
 - a generate a random number between 1 and 6
 - b ask the user to input a number between 1 and 6
 - c reject any values outside the acceptable range
 - d display 'Well Done' if the user guesses correctly or 'Try Again' if their guess is incorrect.
- 2 Develop a test plan for this guessing game program.
- 3 Can you write the guessing game program in the high-level programming language you are studying?
- 4 Test your program by carrying out the tests you planned. Ensure you update your test plan after completing each test.

SKILLS CRITICAL THINKING,
REASONING

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS REASONING

CHECKPOINT

Strengthen

S1 What are the three types of error associated with program development? Can you identify the stage(s) of development at which they are most likely to occur?

S2 Outline the function of a trace table.

S3 What are the features of an IDE that help programmers write error-free code?

S4 What is the function of a test plan in program development?

S5 What is meant by normal, boundary and erroneous data?

S6 Why might it be necessary to retest a program once all the planned tests are completed?

Challenge

C1 Think of a problem which could be solved using a computer program. What are the requirements for the program? Create a solution and draw up a test plan for it.

C2 Develop and implement the program. Conduct your planned tests and make any necessary changes to your program, ensuring your test plan is kept up to date.

C3 Reflect on your program and provide an evaluation.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try completing Activities 28–32.

SUMMARY

- Logic errors occur when there is an error in the logic of the code, causing the program to produce an unexpected result.
- A syntax error occurs when part of the code breaks the rules of the programming language.
- A runtime error occurs while the program is running and it is asked to do something that is impossible to do.
- Trace tables can be used to manually trace the execution of an algorithm, allowing you to track the changes in variable values.
- Before creating a program, it is important to produce a test plan that outlines how the final program will be tested to ensure it meets the requirements.
- Evaluating a program involves considering its strengths and weaknesses and identifying areas for improvement.

UNIT QUESTIONS

SKILLS

CRITICAL THINKING,
REASONING

A02

- 1 a Identify the line number(s) that show one example of each of these structural components in the program below: (6)

- variable initialisation
- type declaration
- selection
- iteration
- data structure
- subprogram.

```

1 SET valuesArray TO [3, 9, 12, 16, 4, 98]
2 REAL max
3
4 FUNCTION maxCalc(values)
5 BEGIN FUNCTION
6     SET length TO LENGTH(values)
7     SET max TO 0
8     FOR index = 0 TO length - 1 DO
9         IF values[index] > max THEN
10             SET max TO values[index]
11         END IF
12     END FOR
13 RETURN max
14 END FUNCTION
15
16 SET max TO maxCalc(valuesArray)
    SEND max TO DISPLAY

```

- b Identify the line number where a subprogram call is made. (1)

HINT

Question 1 is testing your ability to identify the seven key structural components of a program. Make sure you have identified the line numbers in which the components can be found.

SKILLS

CRITICAL THINKING,
REASONING

A03

- 2 Describe what this algorithm does. (2)

```

SET scores TO [45, 67, 34, 98, 52]
SET length TO LENGTH(scores)
SET count TO 0
FOR index FROM 0 TO length - 1 DO
    IF scores[index] >= 50 THEN
        SET count TO count + 1
    END IF
END FOR

```


SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
REASONING

A02

- 3 Draw and complete a trace table for this algorithm with these column headings: (5)
- length
 - count
 - index
 - scores[index].

HINT

Question 3 tests your ability to trace an algorithm using a trace table. Remember each variable change should be recorded and each time a line of code alters the value of a variable or variables a new row of the trace table should be completed.

SKILLS

CRITICAL THINKING,
REASONING

A02

- 4 Open file Q01a. Answer these questions about the code.
- a State the name of a user-defined subprogram. (1)
 - b State the name of one in-built subprogram. (1)
 - c State the names of one input parameter. (1)
 - d State the name of a global variable. (1)
 - e State the name of a local variable. (1)
 - f State the line number of the command that 'calls' the variable. (1)

HINT

These questions are asking you to 'state' various elements in the program. You do not have to describe them or explain how they function.

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING,
REASONING

A02

A03

- 5 Open file Q02a. Answer these questions about the code. (1)
- A data structure has been used to store the results of a survey to find favourite brands of automobiles. (2)
- They have been stored in ascending order.
- a Name this type of data structure. (1)
 - b The user is asked to enter the name of a brand and their input is converted into upper case. Explain why this is done. (2)
 - c Complete the program to search for the brand name entered in the data structure. If it is present, then the program should inform the user of its position, e.g. Audi is in position 1. (6)
- The user should also be informed if it is not in the list. (6)
- Save your completed program as Q02b.

HINT

This question is asking you to traverse a list to find a particular item. When you print out the result text and variables have to be concatenated.

SKILLS

PROBLEM SOLVING

A03

- 6 Open file Q03a. (10)
- It shows a list of users and their (not very strong) passwords stored in a list. Complete the program so that a user can enter their user name. If the name exists, then they should be asked for their password. They should be informed of the following:
- if the username and password are correct
 - if the name they entered is not recognised.

HINT

The question requires you to use a loop and selection.



UNIT 3

DATA

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

In this unit, you will learn how data such as text, images and sound are represented and **manipulated** in computer systems using the binary system. When we look at an image, we see colours and shapes but to a computer, it is millions of 1s and 0s. You will also learn how this data can be **compressed** to make it easier to store and transmit and how it can be kept secure by encrypting it.

12 BINARY

SUBJECT VOCABULARY

binary information represented by only two values (e.g. a voltage or no voltage; on or off). There are no communication errors or misunderstandings because there are no small differences

digital information represented by certain fixed values (e.g. high, medium or low). Any signal between these values would be meaningless and not used. Sending and receiving systems do not have to be as accurate as for analogue communication

analogue using signals or information represented by a quantity (e.g. an electric voltage or current) that is continuously variable. Changes in the information being represented are indicated by changes in voltage. This method requires very accurate sending and receiving systems

Binary is a base-2 numeral system using only two digits: 0 and 1. It is a positional notation where digits have place values, like the denary or decimal system that we are familiar with. For example, in the decimal system, each 1 in the number 111 represents a different value i.e. from left to right they represent 100, 10 and 1.

LEARNING OBJECTIVES

- Understand that computers use binary to represent data (numbers, text, sound, graphics) and program instructions
- Understand how computers represent and manipulate numbers (unsigned integers, signed integers [sign and magnitude, two's complement])
- Be able to convert between binary and denary whole numbers (0–255)
- Understand how to perform binary arithmetic (adds, shifts [logical and arithmetic]) and understand the concept of overflow
- Understand why hexadecimal notation is used and be able to convert between hexadecimal and binary
- Understand that file storage is measured in bytes and be able to calculate file sizes

WHY BINARY?

GENERAL VOCABULARY

manipulate to handle or control something in a skilful manner

compressed pressed into a smaller space

transistor a device that controls electronic current

intensity the strength of something that can be measured, e.g. light, sound, heat

frequency the number per time unit, e.g. number per second

transmit cause something to move from one place to another

Binary is needed to represent data and program instructions because of the way in which computers work.

The processor, which processes all of the data and instructions, contains billions of **transistors** which are connected together to form circuits.

The transistors act as switches, similar to light switches. They have only two states: on or off; they either **transmit** an electric current or they do not. A system with separate states is said to be **digital**. If there are two states, the system is binary. There are no in-between states with different levels of current as there would be in a dimmer switch, which produces different levels of brightness in a bulb. A system such as this, where there is a continuous range between two values, is said to be **analogue**.

As there are only two states, on or off, the states are represented by the digits of the binary number system, 1 and 0. All of the data and program instructions processed by a computer are nothing more than streams of millions of 1s and 0s.

Numbers, text, graphics and sound are all represented in the same way, as a series of 1s and 0s. The program instructions that the processor is following allow it to interpret them in different ways.

REPRESENTING INFORMATION

SUBJECT VOCABULARY

binary digit (bit) the smallest unit of data that is represented in a computer. It has a single binary value, either 1 or 0

DID YOU KNOW?

Humans have digital systems. The nerves which control all of our actions are a digital system. They transmit nerve impulses (bursts of electricity) from the sense organs to the brain or from the brain to the muscles. These impulses are all of the same **intensity**. The **frequency** of impulses (rather than the size) signals the strength of whatever caused it.

GENERAL VOCABULARY

intensity the strength or size of the impulses

frequency the number of impulses per second

There are only two digits: 1 and 0. However, to represent the symbols in text, there must be at least 52 separate items of information – 26 lower-case and 26 upper-case letters. Then there are punctuation symbols – the full stops, question marks, etc.

When we write, we combine the letters in our alphabet to create words and combine these into sentences to express meaning. In a similar way, the two **binary digits (bits)** can be combined into groups.

A single bit offers two possibilities. In a graphics program it could give black (1) or white (0), for example. If we wanted colour, we'd need more combinations. If two bits are used this gives four possible combinations (00, 01, 10, 11), each of which can represent a colour; so four colours could be used.

Three bits would give eight combinations, 000, 001, 010, 011, 100, 101, 110, 111, allowing eight different colours to be represented. With four bits there would be 16 combinations.

There is obviously a mathematical relationship between the number of bits used and the number of possible colours.

- With 1 bit there are 2 possible colours.
- With 2 bits there are 4 possible colours.
- With 3 bits there are 8 possible colours.
- With 4 bits there are 16 possible colours.

The number of possible colours is equal to 2 to the power of the number of bits used in the combination.

$$2^1 = 2$$

$$2^2 = 2 * 2 = 4$$

$$2^3 = 2 * 2 * 2 = 8$$

$$2^4 = 2 * 2 * 2 * 2 = 16$$

NUMBER SYSTEMS

Binary is a number system based on two digits, 0 and 1.

We are more used to the denary system which has ten digits, 0 to 9, but the binary system functions in the same way. The difference is that the denary system works in powers of 10, while the binary system uses powers of 2.

DENARY SYSTEM

Every digit has a place value, and the one to the left has a value 10 times higher than the one to the right.

This table shows the place values for the digits in the number 3639.

PLACE VALUES	10^3	10^2	10^1	10^0
	1000	100	10	1
	3	6	3	9

Therefore, 3639 can be written as:
 $(3 * 1000) + (6 * 100) + (3 * 10) + (9 * 1)$

BINARY SYSTEM

SUBJECT VOCABULARY

byte the basic combination of bits used to represent an item of information. A byte typically consists of 8 bits

The binary system has similar place values, but they increase by powers of 2.

The table shows the place values for the digits in a **byte**.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1

CONVERTING BINARY TO DENARY

This table can be used to convert a binary number to a denary one.

The denary equivalent of the binary number 10101101 can be calculated as shown in the table below:

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	0	1	0	1	1	0	1

$$(1 * 128) + (1 * 32) + (1 * 8) + (1 * 4) + (1 * 1) = 128 + 32 + 8 + 4 + 1 = 173$$

SKILLS REASONING,
PROBLEM SOLVING

KEY POINT

Calculators are **not** allowed in the exam. These examples should not require one.

ACTIVITY 1

DENARY EQUIVALENTS

What are the denary equivalents of the following binary numbers?

- a 00111001
- b 11000110
- c 10101010

CONVERTING DENARY TO BINARY

The table of place values can be used to convert denary numbers to binary.

WORKED EXAMPLE

Convert denary 213 to binary. Use the table and fill it in as you calculate the equivalent.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1

Move from left to right:

Step 1: is the number (213) equal to or greater than 128? If it is, place a 1 at this place value and if not, place a 0.

In this case, 213 is greater than 128 so place a 1 in this column and calculate the remainder.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1							

The remainder is 85.

Step 2: is 85 greater than or equal to 64? Yes, therefore place a 1 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1						

The remainder is 21.

Step 3: is 21 greater than or equal to 32. No, therefore place a 0 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0					

The remainder is still 21.

Step 4: is 21 equal to or greater than 16? Yes, therefore place a 1 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0	1				

The remainder is 5.

Step 5: is 5 equal to or greater than 8? No, therefore place a 0 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0	1	0			

The remainder is still 5.

GENERAL VOCABULARY

vice versa used to say the opposite of what you have just said

validate ensure that something is acceptable, e.g. numbers are in a certain range

EXTEND YOUR KNOWLEDGE

The methods to convert binary to denary and **vice versa** can be performed by computer programs. Write and test programs that will carry out these conversions. The programs should ask a user to enter either a denary or an 8-bit binary number.

Remember to **validate** their input:

- the denary number should be no larger than 255
- the binary number should have eight digits and each digit must be either 1 or 0.

Step 6: is 5 equal to or greater than 4? Yes, therefore place a 1 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0	1	0	1		

The remainder is 1.

Step 7: is 1 equal to or greater than 2? No, therefore place a 0 in this column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0	1	0	1	0	

The remainder is still 1.

Step 8: is 1 equal to or greater than 1? Obviously, they are equal and therefore place a 1 in the last column.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	1	1	0	1	0	1	0	1

Therefore, denary 213 is represented by binary 11010101.

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 2

BINARY EQUIVALENTS

What are the binary equivalents of the following denary numbers?

- a 69
- b 193
- c 239

BINARY ARITHMETIC

Binary numbers can be manipulated in the same way as denary ones.

DENARY ADDITION

When addition is performed in denary, a 'carry over' is used if the result is greater than 9.

WORKED EXAMPLE

369 + 733

PLACE VALUES	10^3	10^2	10^1	10^0
	1000	100	10	1
Carry over	1	1	1	
		3	6	9
		7	3	3
Total	1	1	0	2
		100 + 300 + 700 = 1100 The 1000 is carried over leaving 100.	10 + 60 + 30 = 100 The 100 is carried over.	9 + 3 = 12 The 10 is carried over leaving 2.

BINARY ADDITION

Binary addition works in the same way, but a 'carry over' is needed if the result is greater than 1.

WORKED EXAMPLE

11010110 + 01100111

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
Carry over	1				1	1		
1	1	1	0	1	0	1	1	0
	0	1	1	0	0	1	1	1
Total	0	0	1	1	1	1	0	1
	128 + 128 = 256 This should be carried over.	64 + 64 = 128 The 128 is carried over.	No carry over needed.	No carry over needed.	No carry over needed.	4 + 4 + 4 = 12 The 8 is carried over leaving one 4.	2 + 2 = 4 The 4 is carried over.	No carry over needed.

SUBJECT VOCABULARY

overflow error this condition occurs when a calculation produces a result that is greater than the computer can deal with or store. When this happens, the microprocessor is informed that an error has occurred

SKILLS

REASONING,
PROBLEM SOLVING

EXTEND YOUR KNOWLEDGE

Overflow errors have led to many real-life disasters. The most famous was the crash of the Ariane 5 space rocket, which exploded 40 seconds after its launch in 1996. Others include the failure of the Patriot missile defence system in 1991. Do further research to find out more details about these 'overflow' disasters and problems.

We are adding 8-bit numbers, and this has caused a problem. All 8 bits have been used and the 1 that was carried over in the last column has nowhere to go – it has been 'carried out'. Therefore, the result of the calculation would be wrong. This is called an **overflow error**.

This condition occurs when a calculation produces a result that is greater than the computer can deal with or store. When this happens, the processor is informed that an error has occurred.

ACTIVITY 3

BINARY ADDITIONS

Carry out the following binary additions.

- a $10011010 + 11010111$
- b $00001101 + 10101010$
- c $11010111 + 10001010$

SIGNED AND UNSIGNED NUMBERS

SUBJECT VOCABULARY

integer a whole number (e.g. 3, 6, 9)
most significant bit (MSB) the bit with the highest value in a multiple-bit binary number

So far, we have been considering whole numbers or **integers** that are positive, but how can we represent negative numbers? When an integer is indicated as being positive or negative, it is described as being signed.

In denary, this is simple, a '+' or a '-' symbol is placed in front of the number. If we are using 8-bit numbers in binary, we cannot do this because we cannot simply add an extra bit. Also, the processor would not understand the '+' and '-' symbols.

SIGN AND MAGNITUDE

In a multiple-bit binary number, the left-most bit (the one with the greatest value) is called the **most significant bit (MSB)**. We can use this to represent signed integers.

It is set to 1 for negative and 0 for positive.

Therefore, using this method, binary 00011100 represents denary +28 and binary 10011100 represents -28

The most significant bit is being used for the sign, so the largest positive number represented by a byte is 127.

ADDING SIGNED INTEGERS

If the binary equivalents of denary +28 and denary -28 are added, we get the following result.

0	0	0	1	1	1	0	0
1	0	0	1	1	1	0	0
1	0	1	1	1	0	0	0

Adding denary +28 and –28 we expect the result to be 0, but the binary result is equivalent to –56.

Also using this method, 0 could be both positive and negative.

Positive zero	0	0	0	0	0	0	0	0
Negative zero	1	0	0	0	0	0	0	0

That clearly doesn't make sense and so another method is needed to represent negative numbers in calculations.

TWO'S COMPLEMENT

The most common method used to represent signed integers in modern computers is two's complement. This is a method of representing binary numbers to show whether they are positive or negative. This method works on the principle above – the result of any number added to its negative equivalent should be 0.

FINDING THE TWO'S COMPLEMENT OF A BINARY NUMBER

To find the two's complement of a binary number:

- flip all of the bits – change 1s to 0s and 0s to 1s
- add 1 to the result.

Number	0	0	0	1	1	1	0	0
Flip	1	1	1	0	0	0	1	1
Add 1 to flipped value	1	1	1	0	0	1	0	0

HINT

The most significant bit (MSB) represents a negative number rather than a sign. The MSB is 1, indicating a negative number. Therefore, the two's complement of binary 00011100 is binary 11100100, which is equivalent to denary –28.

CONVERTING TWO'S COMPLEMENT NUMBERS BACK INTO DENARY

This example shows that the two's complement binary number 11100100 represents denary –28. We can check whether this statement is true by converting it back from two's complement.

METHOD 1 – REVERSE CALCULATION

The positive equivalent of a negative integer in two's complement format can be found by flipping the bits and adding 1.

Number –28	1	1	1	0	0	1	0	0
Flip	0	0	0	1	1	0	1	1
Add 1 to flipped value	0	0	0	1	1	1	0	0

The result is denary 28.

This shows that 11100100 (–28) is the two's complement representation of 00011100 (+28).

METHOD 2 – PLACE VALUES

Using place values, the two's complement number can be converted directly into denary.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	-128	64	32	16	8	4	2	1
	1	1	1	0	0	1	0	0

The MSB of the two's complement binary number is 1, indicating that it is negative. Therefore, the denary equivalent of the MSB is taken as being negative -128.

The denary equivalent will be

$$-128 + 64 + 32 + 4 = -128 + 100 = -28$$

This shows that two's complement 11100100 is equivalent to denary -28.

WORKED EXAMPLE

Convert -69 to an 8-bit two's complement binary number.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
69	0	1	0	0	0	1	0	1
Flip	1	0	1	1	1	0	1	0
Add 1	1	0	1	1	1	0	1	1

So denary -69 is 10111011 in two's complement binary.

We can check this using the place value method.

$$-128 + 32 + 16 + 8 + 2 + 1 = -128 + 59 = -69$$

ADDING -28 TO +28

The two's complement of 00011100 is 11100100.

So, adding the binary equivalents of denary +28 and -28 gives:

28	0	0	0	1	1	1	0	0
-28	1	1	1	0	0	1	0	0
Result of addition	0	0	0	0	0	0	0	0

There is an overflow, but this time the result is 0.

EXTEND YOUR KNOWLEDGE

Write and test a program in the language you are using which will calculate the two's complement of an 8-bit binary number. Remember to validate the binary number entered.

DID YOU KNOW?

Additions using two's complement can be used for carrying out subtraction in binary. For example, $28 - 5$ is the same as $28 + (-5)$.

WORKED EXAMPLE

Add 28 and -5 .

This is more difficult. The result should be 23.

The two's complement representation of -5 is:

5	0	0	0	0	0	1	0	1
Flip	1	1	1	1	1	0	1	0
Add 1	1	1	1	1	1	0	1	1

Adding binary 28 to binary -5 gives

0	0	0	1	1	1	0	0
1	1	1	1	1	0	1	1
0	0	0	1	0	1	1	1

The result is 00010111, which is the binary equivalent of 23.

WORKED EXAMPLE

Add -28 to $+5$.

This is slightly more difficult. The result should be -23 .

-28 in two's complement	1	1	1	0	0	1	0	0
5	0	0	0	0	0	1	0	1
Result	1	1	1	0	1	0	0	1

We can show that this is equal to -23 .

$$-128 + 64 + 32 + 8 + 1 = -128 + 105 = -23$$

SKILLS

REASONING,
PROBLEM SOLVING

ACTIVITY 4

TWO'S COMPLEMENT REPRESENTATION

- What is the two's complement representation of the following denary numbers?
 - -113
 - -56
 - -90
- Showing your working, carry out the binary addition of $90 + (-33)$.

BINARY SHIFTS

In denary, when we want to multiply a number by 10, we move each number one place value to its left and add a 0 to the end. For example, $13 * 10 = 130$.

PLACE VALUES	10^3	10^2	10^1	10^0
	1000	100	10	1
Original number			1	3
Multiply by 10. The digits are moved to the left and a 0 is added.		1	3	0

To divide by 10, we do the **reverse** – move each number one place value to its right.

A left **shift** is used for multiplication by 10 and a right shift for division by 10.

If the number is being multiplied by 100 (10^2), then each digit is moved two places to its left.

In binary left and right, **binary shifts** can be used for multiplication and division by powers of 2.

GENERAL VOCABULARY

reverse the opposite

shift to move something to another place

SUBJECT VOCABULARY

binary shift an operation done on all the bits of a binary value in which they are moved by a specific number of places to either the left or right

LOGICAL SHIFTS

For unsigned numbers, logical shifts are performed in the same way as in denary.

If we multiply the unsigned binary number 00010100 (20) by 2^2 (4), there should be two shifts to the left.

The left-most bits drop off the end and are replaced by 0s at the right.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	0	0	0	1	0	1	0	0
Result of shift	0	1	0	1	0	0	0	0

The product is $(64 * 1) + (16 * 1) = 80$ in denary as expected.

If we divide the same number by 4 (2^2), there would be two shifts to the right. This time the right-most bits drop off the end and are replaced by 0s at the left.

PLACE VALUES	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
	0	0	0	1	0	1	0	0
Result of shift	0	0	0	0	0	1	0	1

The result is $(4 * 1) + (1 * 1) = 5$ in denary as expected.

SKILLS

REASONING,
PROBLEM SOLVING

ACTIVITY 5

LOGICAL SHIFTS ON UNSIGNED INTEGERS

What are the results of the following shifts on unsigned integers?

a $00111010 * 2^3$

b $10011101 / 2^4$

A logical shift on integers can make a number less precise. For example, if we divide 00100001 (33 in denary) by 2 the digits would be shifted one place to the right.

PLACE VALUES	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
	128	64	32	16	8	4	2	1
	0	0	1	0	0	0	0	1
Result of shift	0	0	0	1	0	0	0	0

SUBJECT VOCABULARY

arithmetic shift used for signed binary numbers. When performing a right shift, the bits at the left are replaced by copies of the most significant bit

The result is 00010000, which is 16, but the correct result should be 16.5.

ARITHMETIC SHIFTS

Arithmetic shifts are used with signed numbers expressed in two's complement format.

LEFT ARITHMETIC SHIFT

A left arithmetic shift is identical to a left logical shift except that the left-most bit (MSB) is not included because it must remain in place to indicate the sign.

WORKED EXAMPLE

Calculate the product of $-36 * 2$

-36 in two's complement format is:

36	0	0	1	0	0	1	0	0
Flip	1	1	0	1	1	0	1	1
Add 1	1	1	0	1	1	1	0	0

Therefore, -36 in two's complement format is 11011100.

If this is multiplied by 2, there should be one shift to the left with a 0 added at the right.

Leaving the MSB in place and shifting the others to the left will produce:
10111000

This should be equal to -72 in two's complement **notation**.

We can see if this is correct by proving that 10111000 is the negative of 72 in two's complement.

-72	1	0	1	1	1	0	0	0
Flip	0	1	0	0	0	1	1	1
Add 1	0	1	0	0	1	0	0	0

01001000 in binary is equal to 72 in denary.

GENERAL VOCABULARY

notation the system of symbols used to represent information

RIGHT ARITHMETIC SHIFT

When dividing signed binary numbers in two's complement format by powers of 2, the bits are shifted to the right. They are replaced at the left by copies of the MSB.

WORKED EXAMPLE

Calculate $-72 / 2^2$. That is, -72 divided by 4. The result should be -18 . -72 in two's complement format is 10111000; we showed this in the previous worked example.

Performing two right arithmetic shifts and adding the MSB at the left will give:

11101110

This should be -18 in two's complement notation.

We can test it as before.

-18	1	1	1	0	1	1	1	0
Flip	0	0	0	1	0	0	0	1
Add 1	0	0	0	1	0	0	1	0

The result is 00010010, which is equivalent to $(1 * 16) + (1 * 2) = 18$.

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 6**ARITHMETIC SHIFTS**

- 1 Show the following division:
 - a $10010000 / 2^2$ in binary
 - b $11110110 / 2^1$ in binary
 - c $11000000 / 2^3$ in binary.
- 2 Show that your answers are the expected results.

HEXADECIMAL NUMBERS**SUBJECT VOCABULARY**

hexadecimal a base-16 number system. There are 16 digits and the place values increase in powers of 16

So far, only 8-bit binary numbers have been considered. However, it is common for data and commands to be represented by 32 and 64 bits.

It is difficult for humans to remember and manipulate 8-bit binary numbers, but almost impossible to use 32-bit ones. Humans get confused with large binary numbers, so we use **hexadecimal** numbers to represent them. Computers never use hexadecimal numbers.

The hexadecimal system uses 16 digits from 0 to 15. Our denary system has only digits for 0 to 9, so the higher hexadecimal digits are represented by upper case letters.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F



DID YOU KNOW?

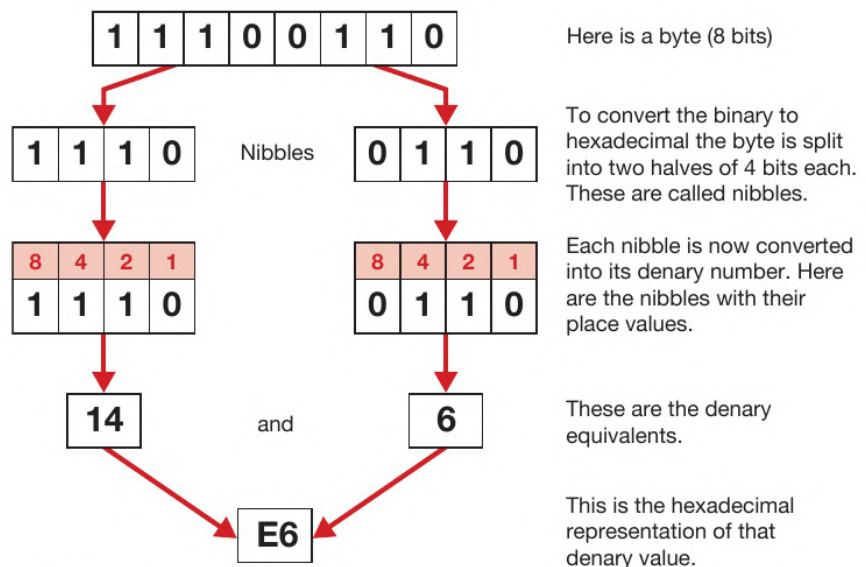
A common use of hexadecimal numbers is to describe colours used in computer graphics. Each of the colours has a value of 24 bits; 8 bits for each of the red, green and blue elements that make it up. The 24 bits are represented in hexadecimal so that they are easier for humans to use.

For example, it is far easier to set the colour to red by entering #FF0000 than having to type 111111110000000000000000.

A # symbol is used to indicate a hexadecimal number.

CONVERTING BINARY TO HEXADECIMAL

The following diagram shows how to convert binary to hexadecimal.

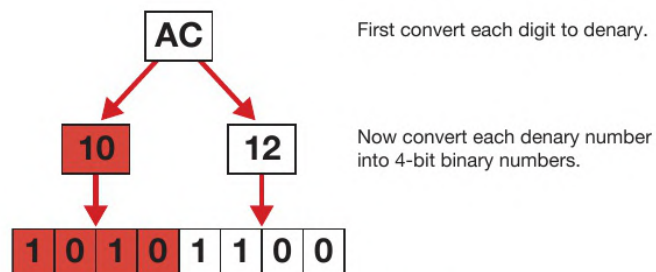


▲ Figure 3.1 Converting binary numbers to hexadecimal

It is far easier to say and remember 'E6' than '11100110'.

CONVERTING HEXADECIMAL TO BINARY

This is the reverse of the method used above.



▲ Figure 3.2 Converting hexadecimal to binary numbers

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 7

HEXADECIMAL AND BINARY CONVERSION

- 1 How can the following binary numbers be expressed in hexadecimal?
 - a 01101110
 - b 10011100
 - c 00101010
- 2 How can the following hexadecimal numbers be expressed in binary?
 - a A6
 - b 9C
 - c 2D

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS CRITICAL THINKING

SKILLS REASONING

CHECKPOINT

Strengthen

S1 Carry out the following addition: $11011001 + 10010010$.

S2 Perform the following calculation in binary: $25 + (-13)$.

S3 Multiply the unsigned integer 01001001 by 2^3 .

S4 What is 11001101 in hexadecimal numbers?

Challenge

C1 Why is binary used to represent data and program instructions in computer systems?

C2 What is the difference between a logical right shift and an arithmetic right shift?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again:

- For S1, try Activity 3.
- For S2, try Activity 4.
- For S3, try Activity 5.
- For S4, try Activity 7.

SUMMARY

- In a computer, all data and programming instructions are represented by streams of 1s and 0s.
- Place values in the binary system increase and decrease by powers of 2.
- An overflow error occurs when a computer attempts to handle a number that is too large (e.g. a number requiring 9 bits instead of 8 in an 8-bit processor).
- To multiply a binary number by powers of 2, you can use left shifts.
- To divide a binary number by powers of 2, you can use right shifts.
- Hexadecimal is used to represent binary numbers because it is easier for humans to remember and use.
- Binary, denary and hexadecimal numbers can be converted from one to the other.

13 DATA REPRESENTATION

All the data that a computer processes is a series of 1s and 0s, whether they are numbers, words, images or sounds. All data to a computer consists of these two digits.

LEARNING OBJECTIVES

- Understand how computers encode characters using ASCII and Unicode
- Understand how bitmap images are represented in binary (pixels, resolution, colour depth)
- Understand how sound, an analogue signal, is represented in binary
- Understand the limitations of binary representation of data (sampling frequency, resolution) when constrained by the number of available bits

REPRESENTATION OF TEXT

SUBJECT VOCABULARY

character set the defined list of characters recognised by a computer's hardware and software

GENERAL VOCABULARY

interchangeable be able to be exchanged between different devices

When you press a key on the keyboard, a stream of 1s and 0s are input into the processor. The computer cannot recognise any characters such as 'c', '@' or 'D', only a combination of 1s and 0s. The defined list of characters recognised by a computer's hardware and software is known as its **character set**.

Obviously, an industry standard is needed, or data will not be **interchangeable** between computers from different manufacturers. This standard is provided by the American Standard Code for Information Interchange, or ASCII code.

DID YOU KNOW?

Long before ASCII, another standard binary code was invented to represent text. This was the Morse code, which uses lights, tones or clicks that are either of short ('dots') or long ('dashes') duration. The duration of a dash is three times the duration of a dot. Each character is represented by a sequence of dots and dashes. For example, the letter 'C' is represented by 'dash dot dash dot'.

Morse Code

A ···—	M —··—	Y —·—·	6 —··—
B —···	N ···—	Z —··—	7 —··—
C —··—	O —··—	A ···—	8 —··—
D —···	P —··—	Q —··—	9 —··—
E ···	Q —··—	U —··—	· —··—
F ····	R ···—	Ch —··—	· —··—
G —··—	S ····	0 —··—	· —··—
H —···	T —··—	1 —··—	· —··—
I ···	U —··—	2 —··—	· —··—
J —··—	V —··—	3 —··—	· —··—
K —··—	W —··—	4 —··—	· —··—
L —··—	X —··—	5 —··—	· —··—

Originally, the ASCII code consisted of 7 bits and so 128 characters could be represented. All of the lower- and upper-case English characters, punctuation marks and control actions such as backspace, shift on, shift off and carriage return were represented.

The printable characters of the 7-bit ASCII code are shown in Figure 3.3. The codes for control actions (mentioned above) are not shown.

If you look at the table, you can see that the codes are grouped according to function. Important groups are 65 to 90 for the upper-case alphabetic characters and 97 to 122 for their lower-case equivalents. The numeric characters 0 to 9 are represented by codes 48 to 57.

D	B	C	D	B	C	D	B	C	D	B	C
32	00100000	space	57	00111001	9	82	01010010	R	107	01101011	k
33	00100001	!	58	00111010	:	83	01010011	S	108	01101100	l
34	00100010	"	59	00111011	;	84	01010100	T	109	01101101	m
35	00100011	#	60	00111100	<	85	01010101	U	110	01101110	n
36	00100100	\$	61	00111101	=	86	01010110	V	111	01101111	o
37	00100101	%	62	00111110	>	87	01010111	W	112	01110000	p
38	00100110	&	63	00111111	?	88	01011000	X	113	01110001	q
39	00100111	'	64	01000000	@	89	01011001	Y	114	01110010	r
40	00101000	(	65	01000001	A	90	01011010	Z	115	01110011	s
41	00101001	)	66	01000010	B	91	01011011	[	116	01110100	t
42	00101010	*	67	01000011	C	92	01011100	\	117	01110101	u
43	00101011	+	68	01000100	D	93	01011101	]	118	01110110	v
44	00101100	,	69	01000101	E	94	01011110	^	119	01110111	w
45	00101101	-	70	01000110	F	95	01011111	_	120	01111000	x
46	00101110	.	71	01000111	G	96	01100000	`	121	01111001	y
47	00101111	/	72	01001000	H	97	01100001	a	122	01111010	z
48	00110000	0	73	01001001	I	98	01100010	b	123	01111011	{
49	00110001	1	74	01001010	J	99	01100011	c	124	01111100	
50	00110010	2	75	01001011	K	100	01100100	d	125	01111101	}
51	00110011	3	76	01001100	L	101	01100101	e	126	01111110	~
52	00110100	4	77	01001101	M	102	01100110	f	127	01111111	DEL
53	00110101	5	78	01001110	N	103	01100111	g			
54	00110110	6	79	01001111	O	104	01101000	h			
55	00110111	7	80	01010000	P	105	01101001	i			
56	00111000	8	81	01010001	Q	106	01101010	j			

KEY: **D** = denary
B = binary
C = character

▲ Figure 3.3 The printable characters of the 7-bit ASCII code

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 8

FINDING ACSII CODES FOR CHARACTERS

Using the table in Figure 3.3, write down the ASCII codes for the characters in the following phrase:

ASCII code

SUBJECT VOCABULARY

encryption the process of turning information (e.g. a message) into a form that only the **intended recipient** can decode, or **decrypt** and read. The message is **encoded** using an agreed method or algorithm. This is called the key. The encrypted message is called a cipher

HINT

The groups of the characters can be important when you are writing a program. For example, if you are **encrypting** some text using a simple Caesar cipher shift key (see page 145), you can easily find the corresponding letters.

For example, suppose the **encryption** key is +3 (a right shift) and you enter the letter 'D'. The ASCII code for 'D' is 68 and so the encrypted value would be 71(G). If the encryption key is -3 (a left shift), then the plain text letter 'D' results in the cipher text letter 'A' (ASCII 65).

ANALYSING A STRING

GENERAL VOCABULARY

encrypt to turn information into a code so that it cannot be read by everyone

intended recipient the person that the message is written for

decrypt to change encoded information

encode to change information into a different form so that it can be read by a computer

To a computer, a string is just a stream of binary codes that represent the characters. Programming languages have functions that will return the denary equivalents of those codes and vice versa.

The Pearson Edexcel pseudocode does not have these functions built in, but in the Python programming language they are `ord()` and `chr()`.

Therefore

- `ord('c')` would return 99, and
- `chr(100)` would return 'd'.

Using the `ord()` function, the ASCII codes of the characters can be found when traversing a string.

By inserting the `ord()` function into the Pearson Edexcel pseudocode, the algorithm that does this can be displayed.

```
SEND "Please enter the text" TO DISPLAY
RECEIVE myString FROM (STRING) KEYBOARD
FOR index FROM 0 TO LENGTH(myString) - 1 DO
    SET number TO ord(myString[index])

    SEND number TO DISPLAY
END FOR
```

#The function is called using the character in the string as an argument.

By using the `chr()` function you can build up a string using characters entered by their denary codes.

The following algorithm will allow you to enter ten numbers which will be converted to characters and appended to the string.

```
SET myString TO ""
FOR index FROM 0 TO 10 DO
    SEND "Please enter a number in the range 65 to 90 or in the range 97 to 122" TO DISPLAY
    RECEIVE number FROM (INTEGER) KEYBOARD
    SET character TO chr(number)
    SET myString TO myString + character
END FOR
```

#This will create an empty string.

CREATING THE FUNCTIONS

The pseudocode does not have these functions built in. You need to define them using the commands and statements that are available so that you can call them when required.

The `ord()` function needs to take the character as a parameter and return its denary code.

You can do this in the following way.

```
FUNCTION ord(character)
BEGIN FUNCTION
    SET arrayAscii TO [["!"][33], [" "][34], ...["z"][122]] #A two-dimensional array containing all
                                                         of the characters with their denary codes.
    FOR index FROM 0 TO LENGTH(arrayAscii) - 1 DO
        IF character = arrayAscii[index, 0] THEN
            code = arrayAscii[index, 1]
        END IF
    END FOR
    RETURN code
END FUNCTION
```

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 9

FINDING CHARACTERS FROM ASCII CODES

Create a function in pseudocode that will return the character from the denary code entered.

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 10

ENCRYPTION

- 1 Create an algorithm that will encrypt the following sentence:
'The ASCII code represents characters.'
- 2 Encrypt it with a shift of 3 to the right (i.e. A should be encrypted as D).
(Remember: the spaces and the full stop should not be changed.)
- 3 Output the encrypted text.
- 4 Present your algorithm as pseudocode or code and test it in the programming language you are studying.

THE EXTENDED ASCII CODE

GENERAL VOCABULARY

compatibility being able to be used with different types of devices

standardisation making things of the same type have the same basic features

In the ASCII code, there are 96 printable characters but there was always a need for more characters in order to accommodate foreign languages, mathematical symbols and special symbols for drawing pictures.

As computers process data in 8-bit bytes, ASCII was extended to an 8-bit code, which allows 256 codes. Unfortunately, there was no **standardisation**. Different manufacturers such as IBM® and Apple® created their own versions with different characters represented by the same codes. With the huge

popularity of home computing in the 1980s and success of manufacturers such as Atari® and Commodore®, each creating their own operating system, there were many different versions of extended ASCII.

UNICODE

To overcome the problem of multiple versions of ASCII, the Unicode Consortium was founded to develop and promote a Unicode standard. This represents and handles text in most of the world's writing systems using 2 or 4 bytes.

Unicode has become the universal standard recognised and used by the major hardware and software manufacturers, such as IBM, Apple, Microsoft®, Oracle® and Sun®. Unicode can represent the characters in all known languages. For **compatibility** and because ASCII was the recognised standard, Unicode characters 0 to 127 are the same as ASCII. In English-language documents, Unicode is represented using the same codes as in 8-bit ASCII.

REPRESENTATION OF BITMAP IMAGES

SUBJECT VOCABULARY

pixel short for 'picture element', the smallest single point of colour in a graphic image

resolution the number of pixels per inch when the image is displayed (for example, on a monitor or on paper)

Images are also represented and stored as binary code.

The basic unit of a bitmap image is the **pixel**.

IMAGE SIZE AND RESOLUTION

Image size is usually written as two numbers, for example 640 × 480, 2240 × 1680, or 4064 × 2704.

The first number indicates the number of pixels along the width of the image and the second is the number in the height. Thus a 640 × 480 image is made up of 307 200 pixels and a 4064 × 2704 image comprises 10 989 056 pixels.

The following images are the same size but have different **resolutions**.



▲ 4288 × 2824 pixels



▲ 100 × 66 pixels

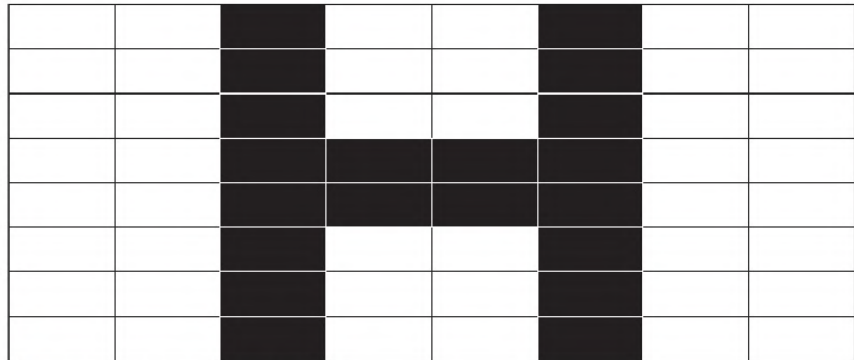
There are different numbers of pixels per square inch in the two images. Each pixel in the 100 × 66 image is a larger size than those of the 4288 × 2848 image. The greater the number of pixels within a given area the higher the resolution and the more detail shown. An image of size 100 × 66, when displayed in the same area as one of 4288 × 2848 pixels, will have a far lower resolution as there are far fewer pixels. The lower the resolution, the less the amount of detail seen.

ENCODING THE PIXEL INFORMATION

Each pixel is represented by a 'bit pattern' – often the same pattern used to represent a character in the ASCII code.

The number of different colours that can be used is decided by the number of bits used to encode colour. For example, if 1 bit is used for each pixel, then there will be only two colours – black and white.

Figure 3.4 shows an encoding system with 1 bit used to represent each pixel.



▲ Figure 3.4 1-bit encoding forming the letter 'H'

If 0 represents black and 1 white, the letter 'H' could be encoded with the following bit pattern.

```
11011011
11011011
11011011
11000011
11000011
11011011
11011011
11011011
```

The complete image consists of 64 pixels and because 1 bit is used for each square, the size of the image is 64 bits or 8 bytes.

As the **colour depth** increases so does the number of colours that can be represented.

- If 1 bit is used, then the number of colours is 2^1 or 2.
- If 3 bits are used the number of colours is 2^3 or 8.
- If 8 bits are used the number of colours is 2^8 or 256.

The current standard represents the colour of each pixel in 24 bits. There are 8 bits for each of the red, blue and green primary colours.

There are therefore 256 variations of each primary colour contributing to the overall colour of the pixel and that gives $256 \times 256 \times 256$ or 16 777 216 different colours.

SUBJECT VOCABULARY

colour depth the number of bits used to encode the colour of each pixel

DID YOU KNOW?

As the size of the image and its colour depth increases, so does the amount of data that has to be stored and manipulated. If you were visiting a website with 4288×2848 24-bit images, you would quickly become frustrated as you waited for them all to download unless you had a very fast connection. Lower resolution images have to be used as a compromise between quality and file size.

This standard allows lifelike images, so it is often referred to as true colour.

This shape is filled with a colour named 'Alice blue' . In binary you would have to enter 1111000011110001111111 every time you wanted to use it. However, as mentioned on page 123, hexadecimal comes to our rescue. Using hexadecimal, you would only have to enter #F0F8FF.

FILE SIZES

The file size for a bitmap image is calculated by finding the total number of pixels and multiplying that by the number of bits used to represent each pixel, or:

Width $\times$ Height $\times$ Colour depth

The file size of the left-hand image on page 125 is:

4288 (width) $\times$ 2848 (height) $\times$ 24 (bit colour depth) = 293 093 376 bits

That is, 36 636 672 bytes.

SKILLS REASONING, PROBLEM SOLVING

KEY POINT

Calculators are not allowed in the exam so you will be asked to just create an expression to calculate file sizes without showing a final value.

ACTIVITY 11

IMAGE FILE SIZES

Create expressions and calculate the file sizes of the following images. Express the sizes in bits and bytes.

- a A 256-colour image with a size of 640×480 pixels.
- b A true-colour image with a size of 640×480 pixels.

REPRESENTATION OF SOUND

GENERAL VOCABULARY

vibrations quickly moving backwards and forwards about a fixed point

All sounds are caused by **vibrations**. As objects such as our vocal cords or guitar strings vibrate backwards and forwards, they push the air molecules alongside them, sending a wave of compressed molecules through the air. When these compression waves, or sound waves, reach our ears they set up vibrations in tiny sensory hairs in the inner ear. This sends nerve impulses to the brain, which interprets them as the sounds we hear.

► **Figure 3.5** Sound waves travelling through the air from a vibrating bell

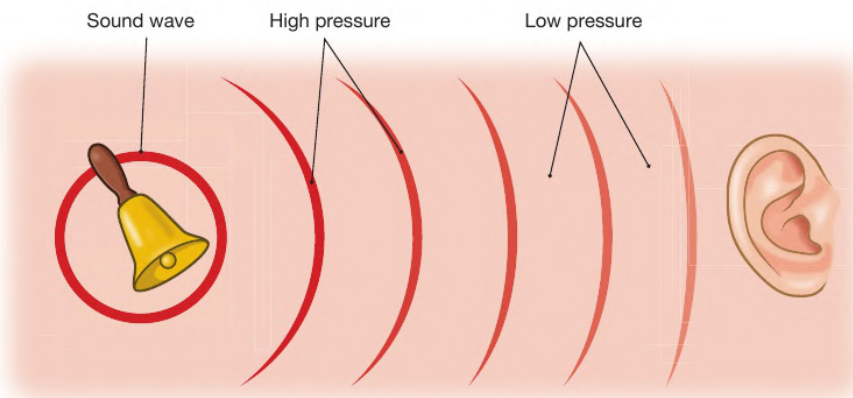
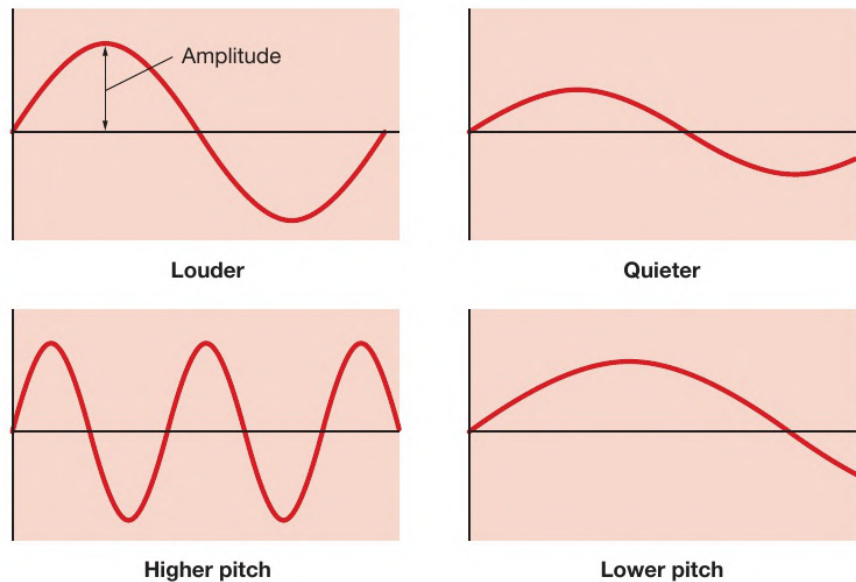


Figure 3.6 illustrates the different waves for loud and quiet and high- and low-pitched sounds.

► **Figure 3.6** The bigger the wave, the louder the sound; the greater the frequency of the waves, the higher the pitch



RECORDING SOUND WAVES

GENERAL VOCABULARY

amplification increasing the strength of a sound; making it louder

proportional they change at the same rate, so that the relationship between them does not change

vinyl record an analogue sound storage system in the form of a flat disc

ANALOGUE RECORDING

Analogue recording represents the continuous changes in air pressure caused by sound waves, as changes in voltage. The changes in voltage are directly **proportional** to the changes in air pressure and these in turn are stored, for example, as grooves of varying depths in **vinyl records** or as magnetised areas of a magnetic tape.

When the disc is played, the movements of another stylus through the groove are converted back into voltage changes. After **amplification**, these cause the diaphragm of a speaker to vibrate, reproducing the original sound.

It is called analogue recording because the changes in voltage exactly mirror the changes in air pressure (or are analogous to them). The complete sound waves are recorded.

DIGITAL RECORDING

It was stressed in the previous section that computers do not represent data in an analogue way where very small changes in voltage are used to represent changes in the data. Computers are digital and their transistors are either on or off, with nothing in between. Therefore, sound data (like any other form of computer data) must be represented as streams of 1s and 0s.

Continuous change cannot be represented by a digital stream, so digital recordings take a series of sound 'snapshots'. When these are played back rapidly in succession the sound produced seems to be continuous. These 'snapshots' of the sound are called samples and the process of taking them is called **sampling**.

SUBJECT VOCABULARY

sampling taking measurements of a sound wave at regular but different intervals of time (e.g. 44 100 samples per second)

► **Figure 3.7** How a sound wave is sampled digitally

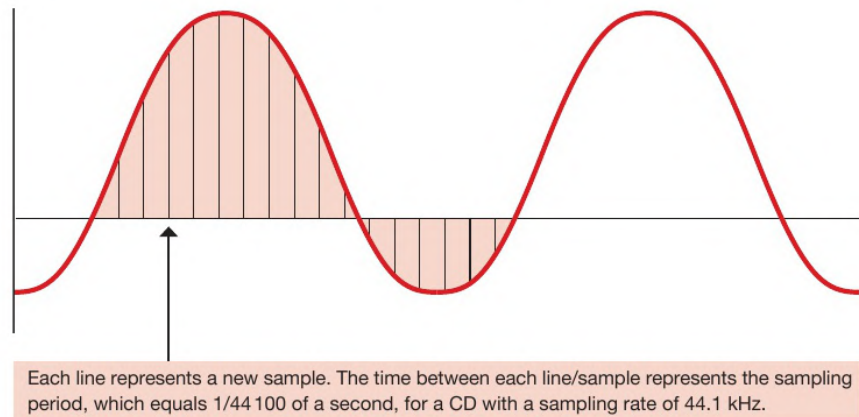
DID YOU KNOW?

A similar technique is used for producing movies on film. If a series of images (or frames) of scenes with small differences between them are played back in rapid succession, our brains think that we are watching moving objects.

SUBJECT VOCABULARY

fidelity the relationship between the original sound and the recording. The greater the fidelity, the greater the similarity between them. Very high accuracy is called hi-fidelity or just hi-fi

Figure 3.7 shows an analogue sound wave being sampled.



The sampling of the continuous sound wave is carried out by an analogue-to-digital recorder (ADR), which converts the sound into a stream of 1s and 0s. A digital-to-analogue converter is required to convert the binary data back into an analogue wave.

The complete wave cannot be recorded and so a digital recording can never be as accurate or have the same accuracy, or **fidelity**, as an analogue recording, but it does have other benefits such as:

- Equipment to record and process digital sound is relatively cheap and has allowed people to record music at home.
- It is easily edited using computer equipment.
- It is easily copied.
- Unlike vinyl and tape, digital files do not physically **deteriorate** with use, although they may become corrupted if the storage medium is damaged.
- Digital files are more portable than records and tapes and can be copied to any digital medium, such as a hard drive, emailed, downloaded and streamed.

FIDELITY

The fidelity of the recording is influenced by two factors.

SAMPLE RATE

This is the number of samples taken per second – the higher the rate the higher the fidelity.

Figure 3.8 shows the effect of sampling at different rates.

A sampling rate of 44 100 per second (44.1 kHz) is used for CDs and of 96 000 Hz (96 kHz) for Blu-ray™ audio.

HOW PRECISE THE BIT DEPTH OR SAMPLING IS

One important detail is the **dynamic** range of the sound – the range of volumes of sound in the music.

- Using 8 bits allows 2^8 **gradations** to be measured.
- 16 bits allows (2^{16}) 65 536.
- 24 bits allows (2^{24}) over 16.7 million.

GENERAL VOCABULARY

deteriorate to get worse over time

dynamic continuously changing

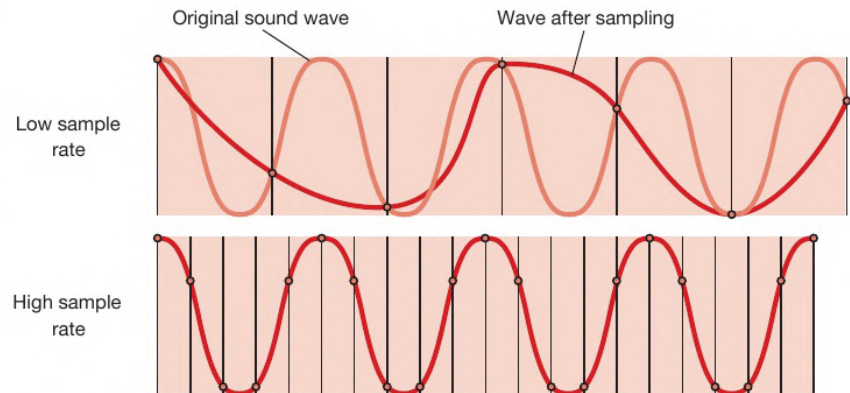
gradations gradual changes from one state to another

Using more bits therefore allows for much smaller gradations in the volume differences.

For CD recordings a **bit depth** of 16 bits is used.

SUBJECT VOCABULARY

bit depth the number of bits used to encode the information from each sample. Increasing the number of bits used increases the amount of detail contained in each sample



▲ Figure 3.8 The effect of sampling at different rates

DIGITAL AUDIO FILE SIZES

The size of a digital audio sound file depends on the following:

- sample rate per second
- bit depth
- duration of recording
- number of channels – mono (one channel) or stereo (two channels).

CALCULATING FILE SIZE

WORKED EXAMPLE

Number of samples per second = 44 100

Bit depth = 16 bits

Duration = 2.5 minutes

Number of channels = 2

The size of the file is found from the following formula:

file size in bits = sample rate * bit depth * duration (in seconds) * number of channels

Therefore, the file size of the above recording is:

$44\,100 * 16 * 2.5 * 60 * 2 = 211\,680\,000$ bits or 26 460 000 bytes.

SKILLS REASONING, PROBLEM SOLVING

ACTIVITY 12

AUDIO FILE SIZES

What is the file size of a stereo recording of three minutes' duration with a sample rate of 44 100 and a bit depth of 24 bits? Give your answer in bits and bytes.

SKILLS PROBLEM SOLVING**SKILLS** PROBLEM SOLVING**SKILLS** REASONING**SKILLS** CRITICAL THINKING**CHECKPOINT****Strengthen**

S1 Represent the following sentence using the denary ASCII codes for the characters: Data is represented as bits.

S2 An image is described as being 640×480 . If it has a bit depth of 24, calculate the size of the file in bits.

Challenge

C1 Why are some digital images clearer and display more detail than others?

C2 What are the factors influencing the file size of a digital audio recording?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again:

- For S1, try Activity 8.
- For S2, try Activity 11.

SUMMARY

- All data is represented by bit patterns – sequences of 1s and 0s.
- The ASCII code represents text characters as patterns of 7 or 8 bits.
- A pixel is the smallest single point of colour in a graphic image.
- The amount of colour information used for each pixel is called the bit depth.
- The file size of a bitmap image is calculated by multiplying the number of pixels in the width of the image by the number in the height and then multiplying by the number of bits used to encode each pixel.
- Digital sound recordings are produced by sampling analogue sound waves.
- The sample rate and the bit depth affect the fidelity of the recording and also the file size.

14 DATA STORAGE AND COMPRESSION

With the universal use of digital devices, huge amounts of data are being generated and stored, with lots of it online. To ensure that this data can be stored and transmitted efficiently, it can be made smaller to occupy fewer bytes and megabytes.

LEARNING OBJECTIVES

- Understand how to use and convert between binary and denary multiples (as defined by the International Electrotechnical Commission [IEC]): bit, nibble, byte, kibibyte (KiB) 2^{10} , mebibyte (MiB) 2^{20} , gibibyte (GiB) 2^{30} , tebibyte (TiB) 2^{40} , kilobyte (kB) 10^3 , megabyte (MB) 10^6 , gigabyte (GB) 10^9 , terabyte (TB) 10^{12}
- Understand the need for data compression and methods of compressing data (lossless, lossy), and that JPEG and MP3 are examples of lossy algorithms
- Understand how a lossless, run-length encoding (RLE) algorithm works
- Understand that file storage is measured in bytes and be able to calculate file sizes

DATA STORAGE

All data consists of bits – 1s and 0s. On page 131, you calculated the size of an example image file as being 293 093 376 bits or 36 636 672 bytes, and the size of an example sound file to be 211 680 000 bits or 26 460 000 bytes – a typical byte is equivalent to 8 bits.

When converting binary to hexadecimal, you used a unit called a ‘**nibble**’, which is equal to 4 bits.

SUBJECT VOCABULARY

nibble 4 bits or half a byte

decimal prefix multiplies a unit by powers of 10

binary prefix multiplies a unit by powers of 2

If these files are saved on computers running different operating systems, their sizes in megabytes (MB) are displayed differently. For example, in Microsoft Windows® the image file size would be shown as 34.9 MB, while in Ubuntu® it would be shown as 36.6 MB.

This is because of the method used to calculate higher units. Ubuntu uses SI units (International System of Measurements), which treats ‘kilo’ as meaning 1000, as it does when measuring distance and weight. This method is known as **decimal prefix** because it uses powers of 10 (e.g. 10^3 for 1000).

The other method, the **binary prefix**, uses powers of 2, so ‘kilo’ is defined as 2^{10} or 1024. Table 3.1 shows the differences.

DECIMAL PREFIX				BINARY PREFIX			
UNIT	SYMBOL	MAGNITUDE	SIZE	UNIT	SYMBOL	MAGNITUDE	SIZE
kilobyte	KB	10^3 bytes	1000 bytes	kibibyte	KiB	2^{10} bytes	1024 bytes
megabyte	MB	10^6 bytes	1000 kilobytes	mebibyte	MiB	2^{20} bytes	1024 kibibytes
gigabyte	GB	10^9 bytes	1000 megabytes	gibibyte	GiB	2^{30} bytes	1024 mebibytes
terabyte	TB	10^{12} bytes	1000 gigabytes	tebibyte	TiB	2^{40} bytes	1024 gibibytes

▲ Table 3.1 Decimal and binary prefixes

Therefore, every higher unit is 1000 times larger than the previous one with decimal prefix, while in binary prefix it is 1024 times larger.

In 1998, to avoid confusion, the International Electrotechnical Commission (IEC) assigned the binary prefix units with new names.

DECIMAL PREFIX	BINARY PREFIX
kilobyte (KB)	kibibyte (KiB)
megabyte (MB)	mebibyte (MiB)
gigabyte (GB)	gibibyte (GiB)
terabyte (TB)	tebibyte (TiB)

▲ Table 3.2 The International Electrotechnical Commission assigned new names to the binary prefix units

As computers are binary, the binary prefix is the most exact, but the decimal prefix units are still used. This may be because it is far easier to multiply and divide by 1000 than it is by 1024! An image file with a size of 36 000 000 bytes would be 36 megabytes or 34.3322 mebibytes.

SKILLS

REASONING,
PROBLEM SOLVING

ACTIVITY 13

STORAGE CAPACITY

- 1 A hard disk is described as having a storage capacity of 1.5 TB. What is this in:
 - a gigabytes
 - b megabytes
 - c kilobytes
 - d terabytes?
- 2 An image file has a size of 363 143 213 bits. Create an expression to convert this size to mebibytes and show the result.

DATA COMPRESSION

SUBJECT VOCABULARY

compression changing the format of a data file so that the size of the file becomes smaller

bandwidth the amount of data that can be carried on a network in a given period of time

lossless compression compressing a file in such a way that it can be decompressed without any loss of data

lossy compression compression where some of the data is removed; the original file cannot be restored when the lossy file is decompressed

GENERAL VOCABULARY

congestion overcrowding and blocking

Files have to be stored somewhere, for example on local or remote hard disk drives or on USB drives. If they are graphics or audio files, then these files can be very large.

Millions of image and audio files are uploaded and downloaded each day and so there are many benefits to the files being made as small as possible – users don't waste time while their data is being transferred or saved and providers can store them in as little space as possible.

That is why file **compression** is so important. Compressed files use less network **bandwidth** to upload and download. There is less Internet **congestion** and it makes possible the streaming of video and audio files.

Compression algorithms are used to make the files as small as possible. There are two types of compression – **lossless** and **lossy compression**.

LOSSLESS COMPRESSION

DID YOU KNOW?

The World Economic Forum estimates that by 2025, 463 exabytes of data will be produced each day. That is 1 000 000 000 000 000 000 bytes, equivalent to 212 765 957 DVDs per day! Every day:

- 500 million tweets are sent
- 294 billion emails are sent
- 4 **petabytes** of data are created on Facebook®.

SUBJECT VOCABULARY

petabyte 1 petabyte is 1000 terabytes or 1 million gigabytes

When lossless compression is used, no data is lost and the original file can be restored. It is especially useful for files where the same data occurs many times (e.g. text) and where there are relatively few different items of data (i.e. character and punctuation symbols). In a page of text, there are many words that are used more than once. This is referred to as redundancy.

Lossless compression is essential for text, as missing data would completely change the meaning of what was being communicated so that it could not be understood.

RUN-LENGTH ENCODING (RLE)

Run-length encoding is used to reduce the size of a repeating string of items. The repeating string is called a run and is represented by two bytes – the first byte represents the number of times the item of information is repeated and the second, the item of information.

For example, the following string of letters:

cccmmmmmssssssddcccccc

would be represented by:

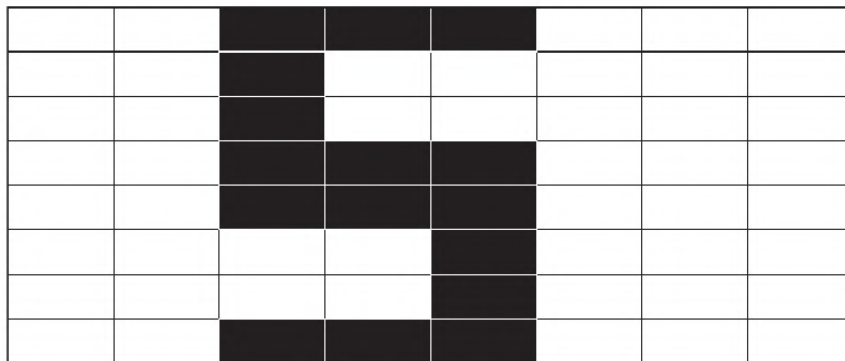
3c5m5s3d6c

If one byte is used to encode each letter in the original uncompressed text, 22 bytes in total are required.

In the RLE version, one byte is used for the letter and one byte is used for the number, making a total of only 10 bytes.

Obviously, this method is only effective where there are strings of the same data. If the original data is very different, without repeating strings, then the encoded version would have a greater file size.

For a black and white image, like the one in Figure 3.9, compression using RLE would be effective. A colour image in which there are very short runs of different colours would not be encoded as effectively.



▲ Figure 3.9 A bitmap diagram of the number 5

To demonstrate run-length encoding, the letters 'w' and 'b' are used to represent the white and black pixels instead of 1s and 0s

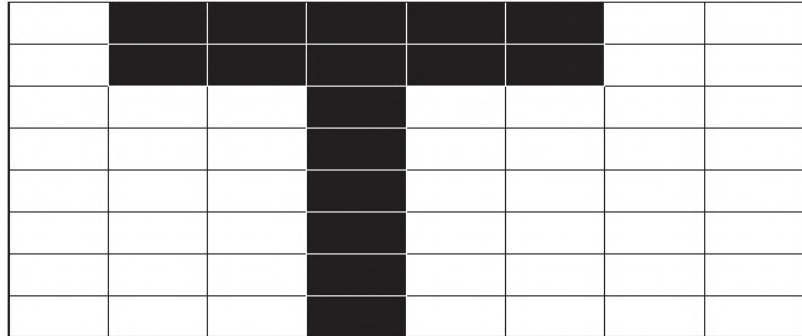
CODE	RLE VERSION	SIZE OF CODED VERSION
wwbbbwww	2w3b3w	6
wwbwwwww	2w1b5w	6
wwbwwwww	2w1b5w	6
wwbbbwww	2w3b3w	6
wwbbbwww	2w3b3w	6
wwwwbwww	4w1b3w	6
wwwwbwww	4w1b3w	6
wwbbbwww	2w3b3w	6
64 bytes		48 bytes

▲ Table 3.3 Run-length encoding of the image of the number 5 in Figure 3.9

The file size for this character has been reduced from 64 to 48 bytes. It could be reduced even further if the number 1 is left out where there is a run of only one. The algorithm could be easily adapted.

SKILLS REASONING,
PROBLEM SOLVING
ACTIVITY 14**RUN-LENGTH ENCODING**

Show the effect of applying run-length encoding to the graphic in Figure 3.10. Set out your answer as in Table 3.3.



▲ Figure 3.10 A bitmap graphic of the letter T

CODING THE RUN-LENGTH ALGORITHM

An RLE algorithm will have to traverse a string and count the number of times each character is repeated in the run. When the character changes, it will have to start counting a new run. The flowchart shows a possible algorithm to do this. First, the input string is checked to see if it actually has any letters (i.e. if the length is greater than 0). If it is, a check is made to see if it only has one letter. If it does, the run will be equal to 1 and the text character will be the one at index position 0 in the string.

If there is more than one character in the input string, then a loop is needed to examine each one.

If the character at index position 1 is equal to the character at position 0, then the length of the run is **incremented** by 1 and the character at index position 2 is checked. If this character is different, then a new run is started for this character and the previous character and its run length are added to the string holding the code. This continues until the algorithm reaches the end of the string.

GENERAL VOCABULARY

increment to increase, usually by a fixed amount

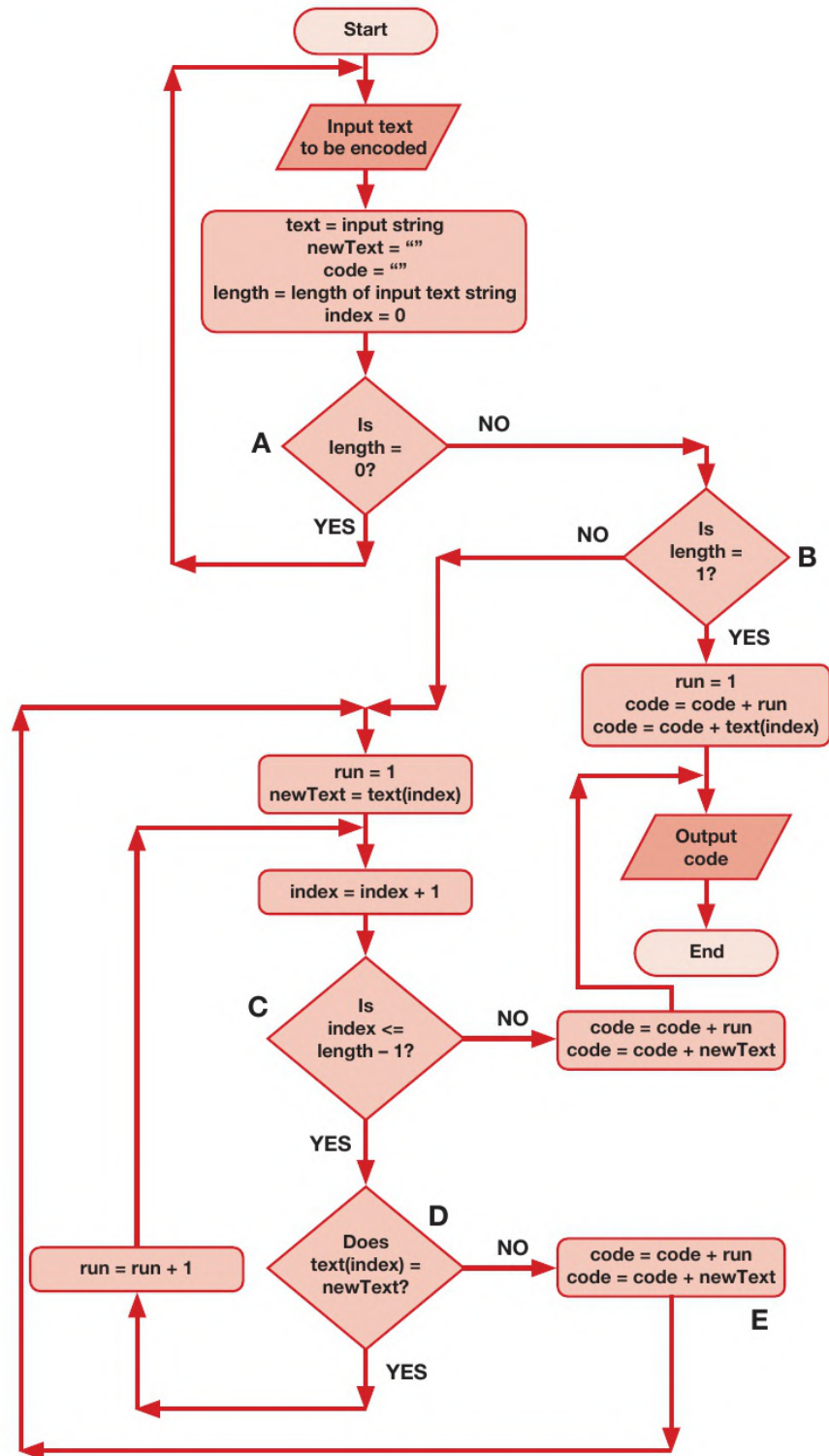
SKILLS CRITICAL THINKING,
REASONING,
PROBLEM SOLVING
EXTEND YOUR KNOWLEDGE

Using the programming language that you are studying, create and test a program to carry out run-length encoding on an input string.

ACTIVITY 15**UNDERSTANDING FLOWCHARTS**

- 1 Six variables are used in the flowchart in Figure 3.11. List the variables and the data items they represent.
- 2 Five items are labelled A to E in the flowchart. Match each item below with the statement that describes it (e.g. A3):
 - 1 This checks to see if the end of the input text has been reached.
 - 2 If there is only one character, it and its run length are added to the output file.
 - 3 This adds the character and its run length to the output string.
 - 4 This checks to see that some text has actually been entered.
 - 5 This checks to see if the next character in the input string is the same as or different from the one that is being checked.

► Figure 3.11 A flowchart of a possible RLE algorithm



LOSSY COMPRESSION

Lossy compression decreases the file size by deleting some of the data. The original file therefore cannot be re-formed entirely when it is decompressed. So, it cannot be used with text or program files.

It can be used for bitmap image and audio files where we often cannot notice that data has been removed.

BITMAP IMAGES

Digital images can comprise millions of pixels each with a 24-bit depth. They result in very large files. However, many of the tiny differences in colour are wasted on us. Our eyesight is not capable of distinguishing these small differences.

A lossy compression algorithm analyses all of the data in the image. When it finds areas with tiny differences, it gives them the same colour values. It can then rewrite the file using fewer bits.

The most commonly used compression technique was developed by the Joint Photographic Experts Group and produces JPEG files with the extension jpg.

The images below demonstrate JPEG compression. The one on the left has a file size of 6 MB, while the compressed version on the right is only 350 KB.

They both have the same number of pixels, but the algorithm made pixels with similar colour values the same. The size has been reduced over 17 times, but to our eyes there is little difference in detail and clarity.



▲ Digitally compressing an image makes little difference to our eyes but huge savings in file size

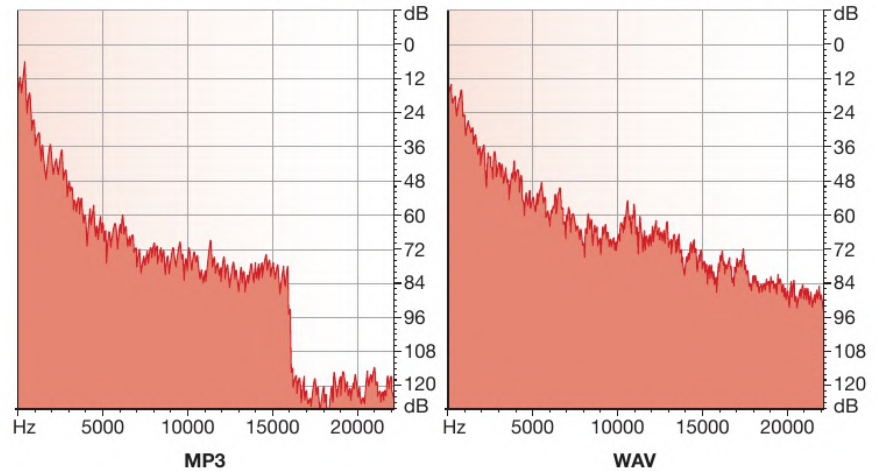
AUDIO FILES

Uncompressed audio files that contain all of the sampled sound data are saved in Waveform Audio (WAV) format. A 3-minute recording typically has a file size of about 30 MB.

Much of the data in an audio file encodes tones and frequencies that our ears cannot hear and small differences in volume and frequency that they cannot distinguish. The most commonly used algorithm compresses them and encodes them as MPEG-1 audio layer 3 files. Or, as they are commonly called, MP3 files, by removing the redundant or excess data.

A 30 MB WAV file can be compressed to a 3 MB MP3 equivalent. The following graphs show the differences in dynamic range between a WAV file and its MP3 equivalent. They clearly show the dip in range at high frequencies.

► **Figure 3.12** The difference in ranges between an MP3 and a WAV sound sample can clearly be seen



SKILLS CRITICAL THINKING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS REASONING

SKILLS CRITICAL THINKING

CHECKPOINT

Strengthen

S1 List the units used to measure file size in ascending order starting at 'bit'.

S2 Convert 2 terabytes into bytes.

S3 Encode the following using run-length encoding and calculate the file size of the original and the encoded versions: aaabbbaaaaaaacccddd

Challenge

C1 Why is it difficult to compress digital images using lossless compression algorithms?

C2 Why does an audio file compressed using a 'lossy' algorithm often sound the same as the original uncompressed version?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again:

- For S1, try Activity 13.
- For S2, try Activity 13.
- For S3, try Activity 14.

SUMMARY

- File sizes are measured in bits, nibbles, kilobytes and kibibytes, megabytes and mebibytes, gigabytes and gibibytes and terabytes and tebibytes.
- Each unit is 1000 or 1024 times larger than the previous one depending on the prefix used.
- Files can be compressed using lossy and lossless compression algorithms.
- In lossy compression some of the data is lost.
- In lossless compression none of the data is lost.
- Run-length encoding is an example of lossless compression.

15 ENCRYPTION

All data needs to be kept secure. One method that has been used for centuries is to use a code or cipher, so that only those who know the rules of the cipher can read the data. As computers can carry out calculations far more rapidly than a human brain, much more complex and secure ciphers can now be used.

LEARNING OBJECTIVES

- Understand the need for data encryption
- Understand how encryption algorithms work (e.g. pigpen cipher, Caesar cipher, Vigenère cipher and rail fence cipher)

DID YOU KNOW?

HTTPS stands for Hypertext Transfer Protocol (HTTP) with the Secure Socket Layer (SSL)/ Transport Layer Security (TLS) protocol added to it. It ensures that data transfer is encrypted.

Ever since people began to use writing, encryption has been used to make messages secret. In ancient Egypt, scribes slightly changed hieroglyphics so that the general public could not understand them.

When we send sensitive information such as bank or credit card details over an Internet connection, we check (or should check) the URL to ensure that it starts with 'HTTPS:'. This means it is a secure connection. Internet connections are not entirely safe, and messages can easily be read by other people. The secure connection ensures that the transferred data is encrypted and is read only by the intended recipient.

DID YOU KNOW?

Other methods have been used to hide messages. A Roman general sent a secret message by shaving off a slave's hair and tattooing the message on his head. When his hair had grown, the slave delivered the message to the recipient who had been told how to find it.

MODERN METHODS OF DATA ENCRYPTION



ASYMMETRIC ENCRYPTION

This method encrypts and decrypts data using two different keys. Every user has two keys, a public key, known to everyone, and a private key, known only to themselves. A message encrypted with a particular public key can only be decrypted by the corresponding private key.

If person A wanted to send an encrypted file to person B, person A would encrypt it with person B's public key. On receipt, person B would then decrypt the file with their private key.

It is commonly used to send encrypted messages because anyone who wants to send an encrypted message can get the intended recipient's public key from a public directory.

SYMMETRIC ENCRYPTION

Symmetric encryption algorithms encrypt and decrypt a message using the same key. This is the method used by a HTTPS connection. The client contacts the server and they establish a secure connection. The client generates the key using an agreed algorithm and sends it to the server using the server's public key (**asymmetric** encryption). The server decrypts it using its private key and then they both use the common key for their secure transmissions.

GENERAL VOCABULARY

symmetric where the two parts of something are matching

asymmetric where the two parts of something are not matching

PLAIN TEXT (INPUT)	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
CIPHER TEXT (OUTPUT)	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B

▲ Table 3.4 The Caesar cipher with a key of -2

With a key of -3 the letters of the alphabet shift three places to the left, so the word 'SECRET' becomes 'PBZOBQ'.

Notice how the letters at the end move to the start of the alphabet. So if the message was:

I WAS SO MUCH OLDER THEN

the encrypted version using a key of $+5$ is:

N BFX XT RZHM TQIJW YMJS

If the recipient knows the key, they can decrypt the message by shifting the letters in the opposite direction by the required number of places.

SKILLS

REASONING,
PROBLEM SOLVING

ACTIVITY 17

USING THE CAESAR CIPHER

Encrypt the following using the Caesar cipher with a key of $+3$.

'THIS IS A MESSAGE.'

HINT

- The ASCII codes are 65 to 90 for the upper-case alphabetic characters and 97 to 122 for lower-case ones.
- A key of $+3$ means that the characters in the message are shifted three places to the right. While if the key is -3 the characters are shifted to the left.
- Have a look at Activity 8 on page 126 and the Hint on page 127.
- Have a look at analysing a string on page 127.
- Remember that string index positions start at 0.
- Have a look at Activity 10 on page 128.

ALGORITHM FOR THE CAESAR CIPHER

This is a perfect example to practise your computational thinking skills.

You are going to write a program to encrypt and decrypt a message using a specific key using the Caesar cipher method.

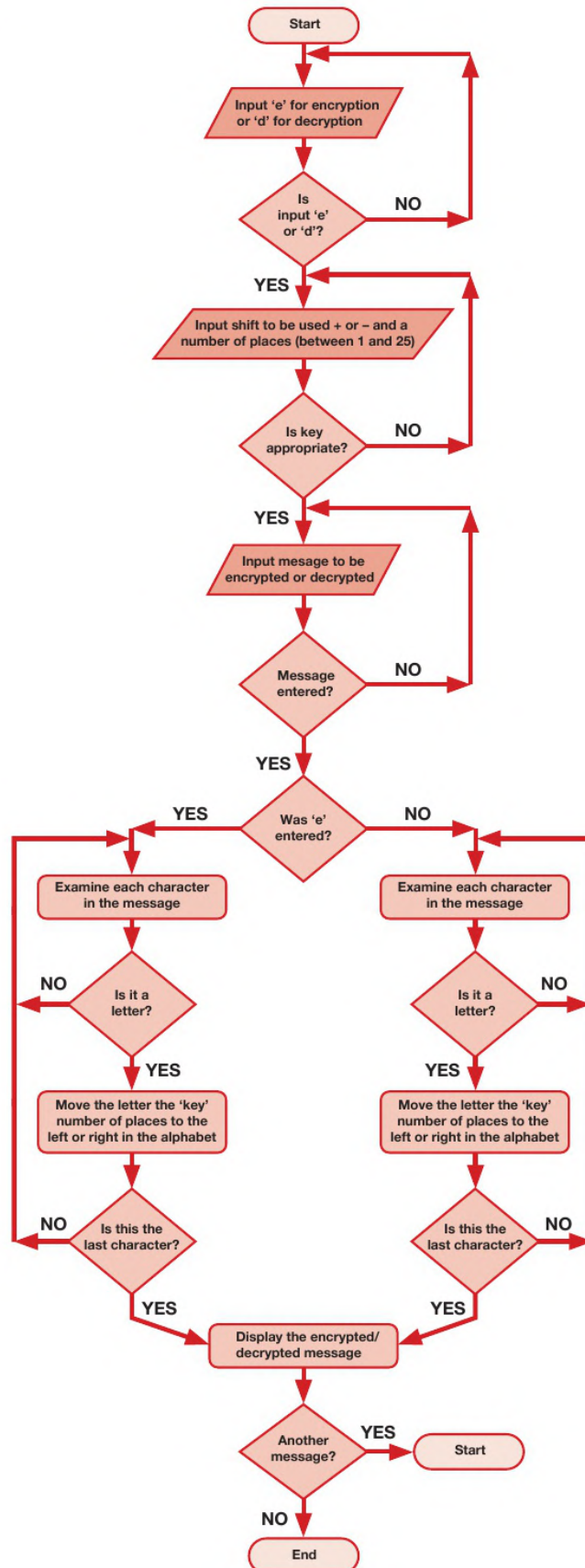
Here is an explanation of the problem:

- 1 Find out whether the message has to be encrypted or decrypted.
- 2 If the message is to be encrypted, then
- 3 ask the user to enter the message.
- 4 Ask the user for the key.
- 5 Analyse each character in the message and if it is a letter calculate the letter the 'key' number of places to the right if the shift is positive or to the left if it is negative.
- 6 If it is a punctuation mark, then do not change it.
- 7 Build up a new message using these new letters.
- 8 Output the new message.

If the message is to be decrypted, repeat the above, but reverse the shift.

Here is a flowchart of a possible solution.

► Figure 3.16 A flowchart showing a possible way of setting out a Caesar cipher



HINT

The ASCII codes for upper case characters run from 65 to 90. The code for 'X' is 88. With a key of +3 this would encrypt to 91, but this doesn't represent a letter of the alphabet.

It should encrypt to the letter 'A', which has the ASCII code of 65.

Therefore, 88 has to be changed to 65.

There are 26 letters in the alphabet, so you should be able to see how to get from 64 to 90 using the number of letters in the alphabet.

Don't forget it's the reverse if you are decrypting.

All this seems straightforward, but there are some more sub-problems.

- Your program will need to check that the user input – the key and the message – has actually been entered.
- The algorithm will have to start at the first character and move to the last (i.e. it must traverse the string), therefore your program will have to find the length of the message.
- Some of the letters may be entered in upper case and some in lower case. Your program should be able to cope with this and encode or decode them.
- Your program will have to leave any characters such as spaces, commas and full stops as they are.

THE 'BIGGEST PROBLEM'

Your program will have to cope with letters near the start and end of the alphabet.

For example, if the key is +3 and the letter to be encoded is 'X', then the encrypted version will be 'A' (because it has been shifted three places to the right).

Similarly, if the letter to be decrypted is 'B', the new version will be 'Y' (because a shift of 3 places to the right has been performed).

The algorithm must be able to move from the first letter of the alphabet to the last and vice versa.

STRUCTURED PROGRAMMING

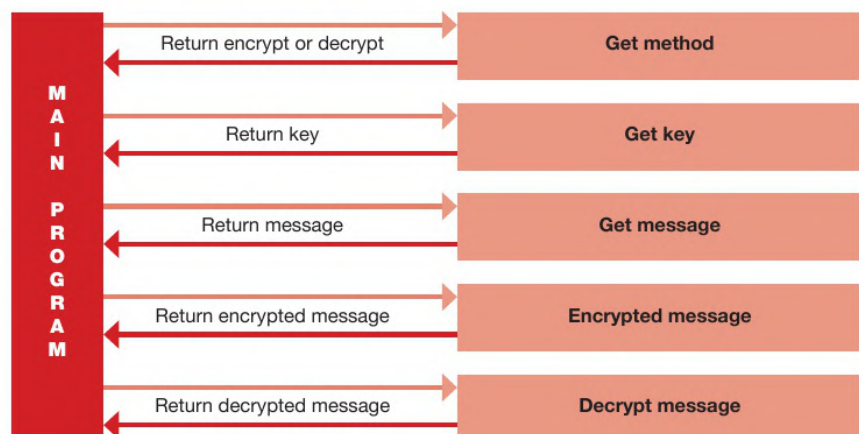
Creating the algorithm and coding the program will allow you to use structured programming using subprograms, and especially functions, which are called from the main program.

For example, functions are used to:

- ask the user if they want to encrypt or decrypt
- ask the user for the key
- ask the user for the message
- carry out the encryption or decryption.

Each of these functions will return values that are used by the main program.

► **Figure 3.17** An example of structured programming



Here is the main program in pseudocode – the subprograms are not included.

```

SET run TO 1
WHILE run = 1 DO

    SET method TO findMethod()

    SET key TO findKey()

    SET message TO getMessage()

    IF method == "e" THEN
        SET newMessage TO encrypt(message, key)
    ELSE
        SET newMessage TO decrypt(message, key)
    END IF

    SEND newMessage TO DISPLAY

    SEND "Please enter x to quit or any other key to continue" TO DISPLAY

    RECEIVE response FROM (CHARACTER) KEYBOARD

    IF response = "x" then

        SET run TO "x"
    ELSE
        SET run TO 1
    END IF
END WHILE

```

#A WHILE loop is started. It will continue until the variable 'run' does not contain the value 1.

#This calls the function for the user to ask for encryption or decryption.

#This calls the function for the user to enter the key.

#This calls the function for the user to enter the message.

#Functions to encrypt or decrypt are called.

#The encrypted or decrypted message is displayed.

#The user is asked if they want to encrypt or decrypt again. If not, the variable 'run' is changed from 1 and the loop will stop.

SKILLS

 REASONING,
PROBLEM SOLVING

ACTIVITY 18
CODING AN ENCRYPTION ALGORITHM

In the language you are studying, create and test a program to encrypt or decrypt a message using a key entered by a user.

GENERAL VOCABULARY

cyclically revolving or recurring regularly

THE VIGENÈRE CIPHER

This cipher uses polyalphabetic substitution. It is based on substitution using multiple alphabets. These alphabets are the 26 possible Caesar ciphers where each alphabet is shifted **cyclically** to the left producing a table.

There are 26 rows, each matching one of the 26 possible Caesar ciphers. The first alphabet goes from A to Z, the second from B to A, the third from C to B, and so on until Z to A.

► Figure 3.18 The Vigenère cipher

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

DID YOU KNOW?

The Vigenère cipher was first described in 1553 by Giovan Battista. However, in the 19th century, Blaise de Vigenère was mistakenly believed to be the inventor. Therefore, it should really be called the Battista cipher!

SUBJECT VOCABULARY

plaintext the text that is to be encrypted
keystream the characters that are combined with the plaintext to produce an encrypted message
keyword the text that is chosen to generate the keystream

ENCRYPTION

We will encrypt, 'This is a message'. The message is referred to as the **plaintext**.

PLAINTEXT	t	h	i	s	i	s	a	m	e	s	s	a	g	e
-----------	---	---	---	---	---	---	---	---	---	---	---	---	---	---

▲ Table 3.5 Plaintext table for 'This is a message'

So that each message will be encrypted differently, a **keyword** is chosen for each encryption. This is similar to the key chosen for the Caesar cipher, but there are far more, almost limitless options, making the cipher far more difficult to solve.

In this example we will use 'code' as the keyword. Using the keyword, the **keystream** is developed by repeating the keyword until it is the same length as the plaintext.

PLAINTEXT	t	h	i	s	i	s	a	m	e	s	s	a	g	e
KEYSTREAM	c	o	d	e	c	o	d	e	c	o	d	e	c	o

▲ Table 3.6 Using 'code' as the keyword to develop a keystream

To encrypt each letter of the plaintext, the intersection of the row with the plaintext letter and the column with the keystream letter is found. The first letter of the plaintext is 't' and for the keystream it is 'c'. The row starting with 't' and the column starting with 'c' intersect at the cell containing the letter 'v'.

► **Table 3.7** Using the Vigenère grid to encode the first letter

	A	B	C	D
A	A	B	C	D
B	B	C	D	E
C	C	D	E	F
D	D	E	F	G
E	E	F	G	H
F	F	G	H	I
G	G	H	I	J
H	H	I	J	K
I	I	J	K	L
K	J	K	L	M
K	K	L	M	N
L	L	M	N	O
M	M	N	O	P
N	N	O	P	Q
O	O	P	Q	R
P	P	Q	R	S
Q	Q	R	S	T
R	R	S	T	U
S	S	T	U	V
T	T	U	V	W
U	U	V	W	X

SUBJECT VOCABULARY

ciphertext the encrypted plaintext

The next letters are 'h' and 'o'. The row starting with 'h' and column with 'o' intersect at the letter 'v'.

In this way, the encoded plaintext (called the **ciphertext**) will be:

VVLW KW D QGGVEIS

DID YOU KNOW?

It was believed for a long time that the Vigenère cipher was unbreakable (i.e. couldn't be decrypted). Even though it was broken in 1863, it is still a very secure cipher.

PLAINTEXT	t	h	i	s	i	s	a	m	e	s	s	a	g	e
KEYSTREAM	c	o	d	e	c	o	d	e	c	o	d	e	c	o
CIPHERTEXT	V	V	L	W	K	W	D	Q	G	G	V	W	I	S

▲ **Table 3.8** The encoded message (ciphertext)

DECRYPTION

This is done in a similar way using the ciphertext and the keyword.

► Table 3.9 Decrypting the message using the ciphertext and keyword

CIPHERTEXT	V	V	L	W	K	W	D	Q	G	G	V	W	I	S
KEYSTREAM	c	o	d	e	c	o	d	e	c	o	d	e	c	o

The first letter of the keystream is 'C'. The user finds the column beginning with this letter and looks down it until they find the letter 'V', the first letter of the ciphertext. They then look along this row to the left-hand edge and find the letter 'T', the first letter of the plaintext. They repeat this process until all of the ciphertext has been decrypted.

SKILLS REASONING,
PROBLEM SOLVING

ACTIVITY 19

USING THE VIGENÈRE CIPHER

Encrypt the message 'computer science' using the Vigenère cipher and a keyword of 'edexcel'.

GENERAL VOCABULARY

transpose to change the order of two or more things

THE RAIL FENCE CIPHER

This is a simple cipher which **transposes** the characters following a simple rule based on a key.

'This is a message' would be encoded in the following way using a key of 3. The letters are set out on three rows or tracks following a zigzag diagonal path.

► Table 3.10 The rail fence cipher with a key of 3

t				i				e				g	
	h		s		s		m		s		a		e
		i				a				s			

To encrypt the message, we read each letter from left to right across each row in order, starting with the top row, which would be t → i → e → g. Therefore, the encrypted ciphertext would be:

tieghssmsaeias

With a key of 4, it would be:

taghsmaeiesss

► Table 3.11 The rail fence cipher with a key of 4

t						a						g	
	h				s		m				a		e
		i		i				e		s			
			s						s				

To decrypt a message a grid is created with as many rows as the key.

The message 'TWLIHSEBOGNWNENRSWNTEDAIH' has been encrypted with a key of 4. Therefore a grid with four rows is drawn and a dash placed in the cells in a diagonal path.

-						-						-					-						-				
	-				-		-				-		-			-		-				-		-			
		-		-				-		-			-		-			-		-				-			
			-						-					-						-						-	

▲ Table 3.12 A rail fence cipher grid with a key of 4 ready to decrypt the message

The letters, starting from the left, replace the rows, starting with the top row.

T					W					L					I					I					
	-				-		-			-		-			-		-			-		-			
		-		-				-		-			-		-			-		-				-	
			-					-					-					-							-
T					W					L					I					I					
	H				S		E			B		O			G		N			W		N			
		-		-				-		-			-		-			-		-				-	
			-					-					-					-							-
T					W					L					I					I					
	H				S		E			B		O			G		N			W		N			
		E		N				R		S			W		N			T		E				D	
			-					-					-					-							-
T					W					L					I					I					
	H				S		E			B		O			G		N			W		N			
		E		N				R		S			W		N			T		E				D	
			A					I					I					H							-

▲ Table 3.13 The stages of filling in the letters of the ciphertext to decrypt the message

Reading the letters in a zigzag, the message is:

THEANSWERISBLOWINGINTHEWIND

or

THE ANSWER IS BLOWING IN THE WIND.

SKILLS PROBLEM SOLVING

ACTIVITY 20

USING THE RAIL FENCE CIPHER

- 1 Encrypt the message 'see you tomorrow' using the rail fence cipher with a key of 3.
- 2 Decrypt the ciphertext 'ELTEDECUEINECOPRCCXMSE' using the rail fence cipher with a key of 4.

CHECKPOINT

Strengthen

S1 Encrypt the sentence below using the pigpen cipher, Caesar cipher with a key of +9, Vigenère cipher with a keyword of 'smile' and rail fence cipher with a key of 4: 'Data should be encrypted'.

S2 Why is the Vigenère cipher more secure than the Caesar cipher?

Challenge

C1 Use a diagram to demonstrate that a shift of +5 followed by a shift of -2 in the Caesar cipher is the same as a shift of +3 when applied to the word 'gold'.

SKILLS PROBLEM SOLVING

SKILLS DECISION MAKING

SKILLS PROBLEM SOLVING

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try the following activities again.

- For S1, try Activities 16, 17, 19 and 20.

SUMMARY

- Encryption is needed to ensure that sensitive data is secure and is read only by the intended recipient.
- Encryption keys are used to encrypt the data into ciphertext and decrypt it back into normal (plain) text.
- Asymmetric encryption uses two different keys and symmetric encryption uses the same one.
- In the pigpen cipher, letters of the alphabet are replaced by symbols.
- In the Caesar cipher algorithm, the letters of the alphabet are shifted a set number of places as defined by the key.
- In the Vigenère cipher the letters are replaced using a keyword and a grid containing 26 combinations of the alphabet.
- In the rail fence cipher, the letters are written in a zigzag pattern according to a given key.

UNIT QUESTIONS

SKILLS

REASONING,
PROBLEM SOLVING

A02

- 1 a Add together the following 8-bit numbers. (1)

```
0 1 0 1 1 0 0 1
1 1 1 0 0 1 1 1
```

- b Identify the problem that this addition has created. (1)

SKILLS

DECISION MAKING,
CRITICAL THINKING

A02

- 2 a Carry out a three-place logical right shift on the following binary number. 10010011 (2)
- b Explain the effect of performing a right shift on a binary number. (2)
- c Describe the steps needed to convert the binary number 11101110 into a hexadecimal one and show the result. (2)

HINT

- 1 a This is a straightforward question. Carry out the calculation and write the result in the space provided. Remember to carry over if the addition of each pair of digits is greater than 1.
- b This just requires a one- or two-word answer.
- 2 a Again, carry out the shift and write the result.
- b Here you have to explain what effect the shift will have. You could say that it is equivalent to multiplying the number by...
- c A longer answer is required. You should show stage by stage how the conversion is carried out. For example, you could start by saying what the 8-bit binary number is divided into. You should set out the explanation clearly and it could be in the form of a diagram.

SKILLS

CRITICAL THINKING

A01

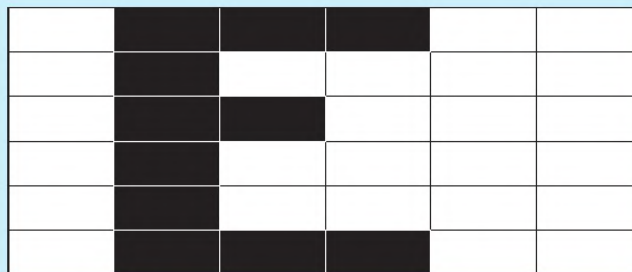
- 3 Explain what is meant by a 'pixel'. (2)

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING

A02

- 4 The following diagram shows a black and white image consisting of 36 pixels.
- a Explain why 36 bits are needed to represent the pixels in the image. (2)
- b Write the bit pattern needed to represent these pixels. (4)



▲ Figure 3.19 A 36-pixel image of the letter 'E'

- c State the number of bits per pixel that would be needed if the image was 16 colours rather than 2. (1)

SKILLS

CRITICAL THINKING,
DECISION MAKING

A01

- 5 Amir is using a computer program to record himself playing the guitar.

- a Describe how the program converts his music into a file. (2)
 b Describe two effects this will have on his recording. (2)
 c He eventually records the music in mono with a sample rate of 44.1 kHz and a bit depth of 16. If the recording lasts for 2 minutes, calculate the size of the file produced. Express your answer in bytes. (2)

A02

HINT

- 3 This simply requires a one-sentence definition.

- 4 a You need to explain why 1 bit will be needed to encode the colour information of each pixel in a black and white image.
 b You should show the bits required for each line of the image. Don't forget to state the type of bit used to represent each colour.
 c You could represent this in powers of 2.

- 5 a This requires a longer answer. You should start with describing how the sound reaches the recording device and then how it is converted into a digital format.
 b Two effects are needed – quality and size.
 c This requires you to apply the formula to calculate file size. Read the question carefully so that you use the correct figures in the calculation and remember that sample rate is defined as samples per second.

SKILLS

DECISION MAKING

A01

- 6 Arrange the following units from smallest to largest. (3)

GiB bit TiB byte nibble MiB

SKILLS

REASONING,
PROBLEM SOLVING

A02

- 7 Compress the following two-colour bitmap data using run-length encoding. (3)

111000110000001110001110

SKILLS

PROBLEM SOLVING

A02

- 8 Encrypt the following word using the Caesar cipher with a key of +6: Computer (3)

SKILLS

PROBLEM SOLVING

A02

- 9 Give the result of decrypting the following ciphertext using the Caesar cipher with a key of left 3: (3)

Pzfbkzb

SKILLS

PROBLEM SOLVING

A02

- 10 Give the result of decrypting the following ciphertext 'ZPSPNOXMOFAORMQDPUKZ' using the Vigenère cipher with a keyword 'giovani'. (3)

SKILLS

PROBLEM SOLVING

A02

- 11 Give the result of decrypting the ciphertext 'BEERMASIGHRNO' using the rail fence cipher with a key of 4. (3)

HINT

These questions do not require you to give an explanation or description. They are asking you to apply the algorithms to encrypt and decrypt.

SKILLS

REASONING,
PROBLEM SOLVING

A02

- 12 a Calculate the decimal equivalent of the binary number 10010101. (2)
 b Convert the binary equivalent of the decimal number 203. (2)



UNIT 4

COMPUTERS

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

This unit introduces the hardware and software that together make up the many different types of computer systems, from embedded microprocessors to distributed, cloud-based systems. You will learn how the processor works and the roles of internal memory and secondary storage devices. You will study the logic that the computer uses when processing instructions and how programs are written using low- and high-level languages.

16 MACHINES AND COMPUTATIONAL MODELS

In this section, you will look at how the data that is entered into a computer system is processed and the results are output for the user. You will study the different ways in which this processing occurs.

LEARNING OBJECTIVES

- Understand the input–process–output model
- Understand that there is a range of computational models (sequential, parallel and multi-agent)

THE INPUT–PROCESS–OUTPUT MODEL

How many computers do you have in your house? There are probably a lot more than you think. Let's find out why.



▲ How many of these things do you think of as computers?

GENERAL VOCABULARY

embedded fixed firmly into another object
format the way in which something is arranged or set out

► Figure 4.1 A simple input–process–output diagram



You are probably familiar with a desktop computer: the keyboard provides input and the screen provides output (which depends on what you type). In a similar way, the embedded computer in a washing machine has controls that provide input and a motor and heater as outputs; how fast they spin and how much heat they produce depends on the input setting.

SKILLS

CRITICAL THINKING,
INTERPRETATION,
COMMUNICATION

ACTIVITY 1

INPUTS AND OUTPUTS

- 1 Name any other inputs and outputs for:
 - a desktop computer
 - a washing machine.
- 2 Aliya is an air conditioning system installer. Research the computing devices she might use in her job.
- 3 Discuss in a group the inputs and outputs for the devices shown on the previous page. Think of some categories that you could use to group the input and output devices you come up with, for example, a pointing device or audio input device.

SUBJECT VOCABULARY

input to enter data into a computer

process to change the meaning or format of some data

output to display or output data that has been processed (or has been stored)

function machine a metaphor or diagram that represents a machine that takes an input. It applies a rule such as a set of operations and delivers the answer as an output

The clever bit in computer science is the activity that goes on between the inputs and outputs – the processing. Processing means performing a series of actions on the inputs, according to a given set of rules. You might have met this idea in maths already if you have studied ‘**function machines**’. You might also have processed ingredients according to a recipe to make a cake, or put together steps according to choreography to make a dance, or musical notes in harmonic combinations to make a song.

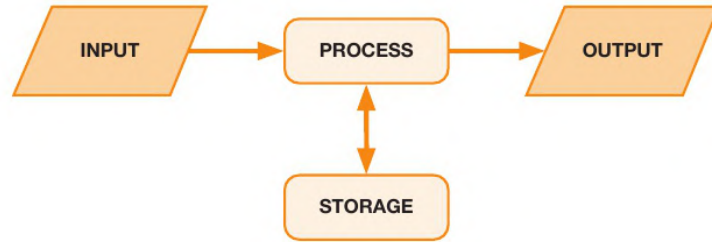
In Unit 1 (page 4) an algorithm was defined as a precise method for solving a problem, consisting of a sequence of step-by-step instructions. In order for processing to produce meaningful results, it has to follow an algorithm. So, you might also think of an algorithm as the sequence of steps that is used to process the input to produce the output.

► Figure 4.2 A simple maths function



If you try to follow a more complicated function machine than this one, you will probably need to keep track of your working. In a similar way, anything beyond the simplest algorithm needs temporary storage or memory. We should change our diagram of a computer to include this.

► **Figure 4.3** A simple function machine with storage



Indicates an input or output.



Indicates a process to be carried out.

▲ **Figure 4.4** Flowchart symbols for input/output and process

As your algorithms become more complicated, it can be useful to show the steps in a flowchart. These are the flowchart symbols that you met in Unit 1 (page 6).

EXTEND YOUR KNOWLEDGE

Make a table with the headings 'Input', 'Process' and 'Output' and with your school subjects down the side. See if you can find an example for each subject starting with the ones in the text, like this:

SUBJECT	INPUT	PROCESS	OUTPUT
Art	paints	instructions	watercolour

For each row, think about whether the rules or instructions for the processing are similar to or different from those we find in computer science. If they are different, in what ways are they different?

RANGE OF COMPUTATIONAL MODELS

Computational models describe how algorithms are executed by a machine, e.g. by processing data using inputs and outputs. There are different models of how this processing can be carried out.

SEQUENTIAL

In the sequential model, this is the way that we commonly imagine how computers work. This involves following instructions in an algorithm step by step, in order, from start to finish.

PARALLEL

In the **parallel** model, computer processes are distributed between two or more processors in a computer with two or more processors installed. Each separate part of the algorithm that each processor processes can be combined together. It requires an operating system capable of supporting two or more processors. It also requires software programs capable of distributing processes between them equally.

GENERAL VOCABULARY

computational connected to computers

parallel two parallel lines are at an equal distance from each other at all points; the term can be used to talk about two things that are happening at the same time as each other

GENERAL VOCABULARY

coordination organising different activities so that they all work together effectively

negotiation a discussion in order to reach an agreement

SUBJECT VOCABULARY

agent a computer system that can interpret its environment (through its sensors). It can act autonomously upon that environment (through its effectors)

autonomous self-directed and acts without requiring any help or guidance. Can choose its own goal and uses its experience to achieve it

effector something that produces an effect by carrying out an action

MULTI-AGENT

In the parallel model, one task is processed by several processors while in the multi-agent model, separate tasks or algorithms are processed by different systems (**agents**) to perform a particular function. Each agent acts independently and under its own control. Each agent is **autonomous**, but they cooperate with each other through **negotiation** and **coordination**. This is the basis of artificial intelligence and reflects our human society, where independent agents (people) interact, negotiate and cooperate to maintain our society and all that is needed for our well-being.

Searching the Internet for the answer to a specific query can be a long and boring process that could be done for us by an agent – a computer program – instead. We would give the query details of what we needed to find out. This would require the agent taking information from various sources. If we were organising a holiday, we could communicate our preferences and different agents could cooperate to search and compare different deals, negotiate for the best prices, plan transport and book airlines. They could even order sun tan lotion or book restaurants for us without our direct involvement!

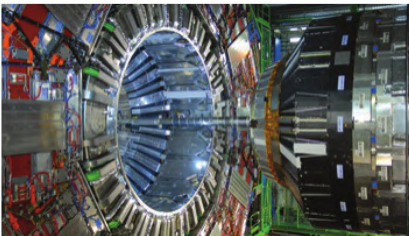
SKILLS PROBLEM SOLVING

ACTIVITY 2

FUNCTION MACHINES

- 1 Work with a partner to write a program to implement a function machine. Decide the mathematical steps required, then ask your partner to write the program. You should check that your partner's program produces the right answers with some example inputs. Do the activity again, but switch roles.
- 2 The sentence below is a written description of an algorithm that checks the strength of a password. Identify the inputs and outputs for this program and describe the processing that takes place.

Ask the user to enter their password. If the password is eight or more characters long and contains at least one upper case letter and at least one symbol then display the word 'Strong', otherwise display the word 'Weak'.



DID YOU KNOW?

There are many examples in history of people trying to discover the processing that a system is doing by changing the inputs and seeing what outputs are produced. For example, Gregor Mendel discovered ideas about genetics by breeding different varieties of pea plants and seeing what the characteristics of the new plants were, and scientists at the Large Hadron Collider are smashing particles together in different ways to help understand the laws of nature by analysing what comes out.

SKILLS

PROBLEM SOLVING,
REASONING

ACTIVITY 3

CODING A PASSWORD ALGORITHM

Code the password algorithm from Activity 2 in the high-level programming language you are learning. You could extend it to include a 'medium' output for passwords that meet only two of the criteria.

SKILLS

CRITICAL THINKING

SKILLS

REASONING

SKILLS

CRITICAL THINKING,
REASONING

SKILLS

PROBLEM SOLVING

CHECKPOINT

Strengthen

S1 What is meant by the term 'input-process-output model'?

S2 What are the inputs, outputs and processing involved in uploading an image to a social networking profile?

S3 In your own words, what are the sequential, parallel and multi-agent computational models?

Challenge

C1 Create a computer program that prompts the user to type in three numbers, adds them up and then displays the answer on the screen. Explain your program using the input-process-output model.

GENERAL VOCABULARY

Complexity when something is complicated or difficult

SUMMARY OF KEY POINTS

- Although computers come in all sorts of shapes and sizes, they all mainly do the same thing. They take in inputs, process them according to a given set of rules, and generate outputs.
- There are different models of different **complexity** for the processing of data by computer systems. Algorithms expressed in program code specify the processes a computer carries out.

17 HARDWARE

In this section you will look at the hardware components that make up a computer system. You will understand how they function and work together, and the need for different types of memory, data and storage.

LEARNING OBJECTIVES

- Understand the function of the hardware components of a computer system (central processing unit [CPU], main memory, secondary storage, input and output devices) and how they work together
- Understand the function of different types of main memory (random-access memory [RAM], read-only memory [ROM], cache, virtual memory)
- Understand the concept of a stored program and the role of components of the CPU – (control unit [CU], arithmetic/logic unit [ALU], registers, clock, address bus, data bus, control bus) – in the fetch–decode–execute cycle (the von Neumann model)
- Understand factors that affect the performance of the CPU (clock speed, number of processor cores, size of cache, types of cache)
- Understand how data is stored on physical devices (magnetic, optical, solid state)
- Understand the concept of storing data in the ‘cloud’ and other contemporary secondary storage
- Understand the need for embedded systems and their functions

STORED PROGRAMS: THE VON NEUMANN MODEL

HINT

Computers may come in all sorts of shapes and sizes, but it’s worth remembering that the four functions they all have in common are input, output, storage and processing.

SUBJECT VOCABULARY

von Neumann architecture
computer system design in which the program is stored in memory with the data

You probably have a calculator app on your smartphone, but your calculator almost certainly can’t play Candy Crush. Your calculator is designed to do just one job – sums. Your smartphone, on the other hand, is a general-purpose computer. It can be programmed to do a number of different jobs, from sums to web browsing to games.

The first computers, like your calculator, had to be redesigned – or at least rewired – to do a different job. The idea of creating a general-purpose computer developed in the 1930s and 1940s through the work of Alan Turing and John von Neumann. They proposed that the instructions for the processing could be held in storage with the input data.

A computer in which the processing instructions are stored in memory with the data is called a stored-program computer or a **von Neumann architecture** computer.

**DID YOU KNOW?**

One of the first electrical (rather than mechanical) computers was ENIAC. Like many technological developments, its original purpose was military. It was mainly used to work out where artillery shells were travelling to, but could be used in other ways. However, this required changing its internal wiring along with a large number of switches. This reprogramming work was given to secretaries, and this being the 1940s, the job was assigned to six women. They are now considered to be amongst the world's first modern computer programmers. The first computer programmer is recognised as Ada Lovelace, the daughter of the poet Lord Byron, but she worked on a mechanical machine designed by Charles Babbage.

SKILLS

RESEARCH,
CRITICAL THINKING,
CREATIVITY

ACTIVITY 4**LOVELACE AND BABBAGE**

With a partner, carry out research and produce a leaflet on the work of Ada Lovelace and Charles Babbage.

SKILLS

REASONING

ACTIVITY 5**THE FETCH–DECODE–EXECUTE CYCLE**

Look online for an animation of the fetch–decode–execute cycle. Try to describe the animation in a step-by-step way.

SKILLS

PROBLEM SOLVING,
REASONING

ACTIVITY 6**CREATING AN ANIMATION**

Use suitable software to produce your own similar animation.

SUBJECT VOCABULARY**central processing unit (CPU)**

hardware device that carries out the processing in a computer

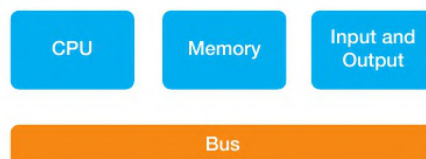
main memory/random-access memory (RAM) a temporary store for data and instructions (programs)

bus a group of connections between devices in a computer

You will remember from Unit 3 that all types of data (including pictures and sounds) are stored in memory as binary numbers. In the same way, the processing instructions are written as binary numbers. You will learn more about this on page 168.

In a von Neumann architecture computer, the hardware device that does the processing is called the **central processing unit (CPU)** and the storage is called **main memory/random-access memory (RAM)**. They are connected to each other, and to the input and output (I/O) devices, by a group of connecting wires called a **bus**.

► Figure 4.5 Buses used in a computer



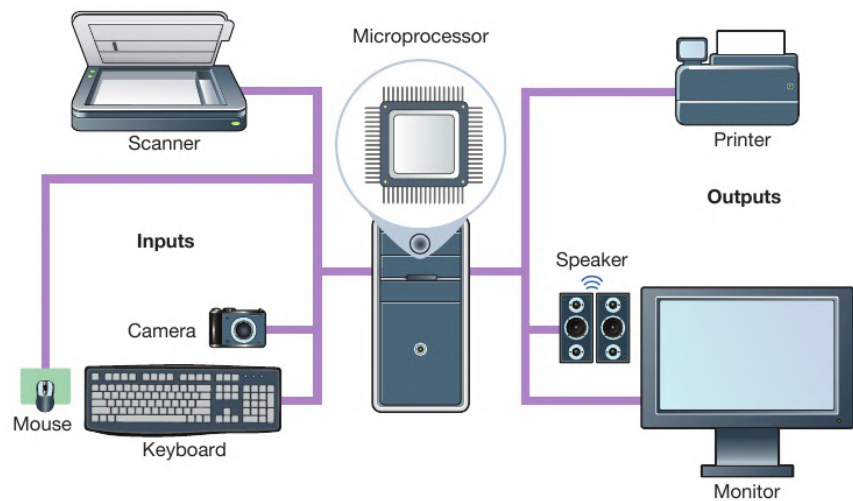
When a computer is turned on, the CPU fetches an instruction from the memory, carries out the instruction it receives, and then fetches the next, and so on. This sequence is called the **fetch–decode–execute cycle** and it carries on until the power is turned off.

INPUT AND OUTPUT DEVICES

Input devices such as keyboards, mice and light sensors provide the computer with data it can process.

Output devices such as monitors, printers and speakers are used by the computer to communicate the results of the processing.

► **Figure 4.6** A computer system with some common input and output devices



SKILLS CRITICAL THINKING

ACTIVITY 7

INPUT AND OUTPUT DEVICES

Make a list of 10 input and 10 output devices.

HARDWARE COMPONENTS OF A COMPUTER SYSTEM

SUBJECT VOCABULARY

fetch–decode–execute cycle

sequence of steps carried out repeatedly by a CPU

writing when the CPU sends data to memory to be stored at a given address

reading when the CPU retrieves the data stored at a given address

memory address a number that uniquely identifies a (memory) storage location

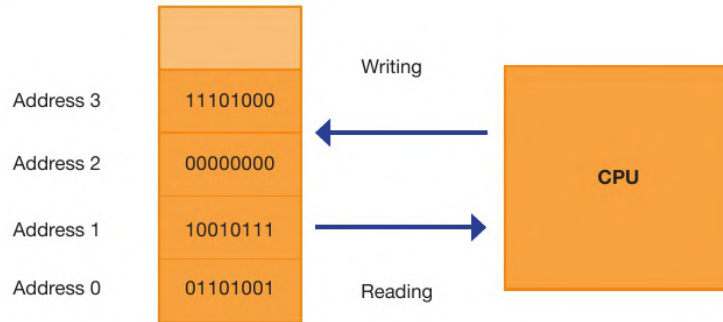
RAM AND ROM

If you have ever bought a desktop or laptop computer, you will have seen the main memory referred to as random-access memory (RAM). This is the temporary store that the CPU uses for data and instructions (programs).

Remember that both data and instructions are just binary numbers, usually in bytes (8 bits). RAM is typically measured in gigabytes (GB). See page 137.

When the CPU saves data into memory, this is called **writing** – the CPU uses the bus to tell the memory what data to save and where in the memory to save it. The reverse process is called **reading** and the CPU must specify which part of the memory to read from. Each memory location has a unique **memory address**, in the same way that your home is identified by an address. You can think of memory like a series of letter boxes each with space for 1 byte of data. The memory addresses are just numbers, starting from 0, labelling each mailbox.

► Figure 4.7 Memory addressing

**SUBJECT VOCABULARY**

volatile memory that is erased when the power is turned off

non-volatile memory that is not lost when the power is turned off

read-only memory (ROM) memory that cannot be altered and is not lost when the power is turned off

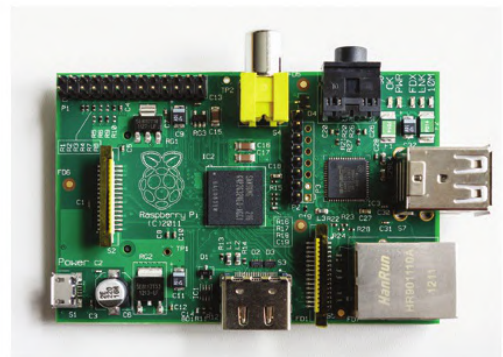
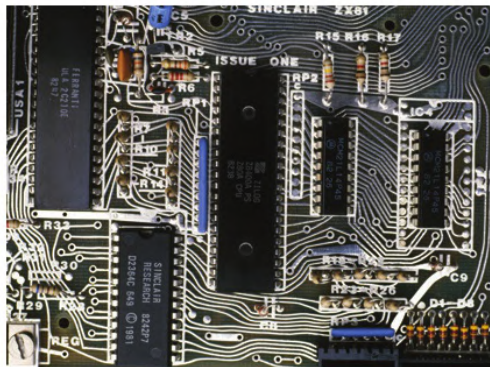
cache memory memory used to make up for the difference in speed between two internal components

RAM is described as **volatile**, which means that its contents are lost when the power is turned off. Because of this, computers also need **non-volatile** memory to store any programs that must run when the computer is first turned on – this memory is called **read-only memory (ROM)**. Programs permanently stored in ROM are known as firmware. These programs are small and carry out specific tasks, such as initialising the hardware components and starting the operating system when a computer is switched on. You might have heard of these programs being called the BIOS (Basic Input/Output System) or UEFI (Unified Extensible Firmware Interface). Nowadays, UEFI has replaced BIOS as the standard firmware used in desktop computers.

The photograph below on the left shows the main circuit board of a 1980s computer (the Sinclair ZX81). The ZX81 had either one or two RAM chips. Two are shown in the diagram, along with three other chips: CPU, ROM and a fourth chip that connects the input and output (I/O) devices. You can also see groups of parallel silver 'tracks'; these are the buses (the wires connecting the chips together).

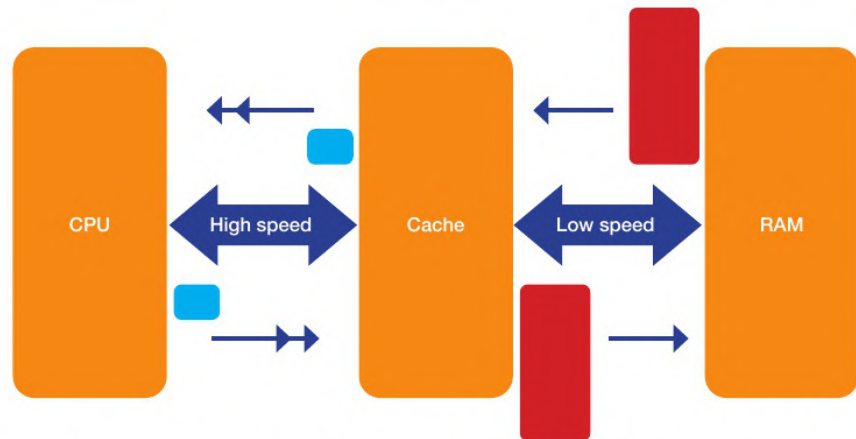
The photograph on the right shows a modern-day computer (the Raspberry Pi®). The chip in the centre contains both the CPU and RAM as well as handling some of the I/O, such as the screen. The role of the ROM is replaced by a plug-in memory card (not shown). The smaller chip to the right handles the slower I/O devices such as USB (Universal Serial Bus). Buses can still be seen, although a bit less clearly.

► Computer circuit boards:
Left – Sinclair ZX81;
Right – Raspberry Pi

**CACHE MEMORY**

Most computers require a third kind of main memory: this is called **cache memory** (pronounced like 'cash'). Cache memory is a small amount of fast, expensive memory that is used between two devices that communicate at different speeds, most often the CPU and RAM.

► Figure 4.8 Diagram of cache memory

**GENERAL VOCABULARY**

chunk quite a large amount of something

bottleneck where something is working at full capacity and cannot handle further demand

rate a measurement of how quickly something happens

SUBJECT VOCABULARY

cache miss when the data requested for processing by a component or application is not found in the cache memory

microprocessor the central unit that executes and manages the instructions passed to it

In a CPU cache, frequently used code or data is loaded in **chunks** from the slower RAM into the cache. The CPU accesses the cache memory at its own, faster speed. This means the CPU isn't slowed down by having to wait for data from the RAM.

DID YOU KNOW?

In the 1980s, CPUs and RAM worked at the same speed as one another, but CPU technology developed much faster than RAM. This meant that the RAM in newer computers could no longer keep up with the demands of the CPU. This caused a **bottleneck** that limited the performance of those computers. To solve the problem, engineers introduced cache memory, but this created a new challenge. If the data the CPU needs is not already in the cache (called a **cache miss**), then the CPU has to wait for the cache to reload with the correct data. There has been a lot of research into the ways of predicting which chunk of RAM to load into the cache at any time, in order to reduce cache misses.

SKILLSRESEARCH,
REASONING**ACTIVITY 8****COMPARING MICROPROCESSORS**

Research the two chip manufacturers Intel® and ARM®. Compare how their **microprocessors** are designed and where they are used.

DID YOU KNOW?

Most I/O devices have memory between themselves and the main system because they communicate at different speeds; this is often called a buffer (e.g. disk buffer, print buffer or frame buffer [for graphics]). Data is transferred at high speed between a processor and this memory. The CPU can then continue with other tasks while the I/O device takes data from its buffer at a slower **rate**.

SKILLSPROBLEM SOLVING,
COMMUNICATION**ACTIVITY 9****PRINT SPOOLING**

Produce a presentation to explain what is meant by **print spooling** and deliver your talk to the class.

VIRTUAL MEMORY

When a computer is running the operating system and several applications at the same time, the RAM often becomes full. Instead of closing some programs, the 'memory manager' program of the operating system will use 'pretend' or virtual memory to store some of the data, usually on the hard disk drive.

It works like this:

- A process running on the computer may need to store data in the physical memory.
- If there is no free memory, the memory manager will '**swap** out' some of the data stored in RAM to the swap area on the hard disk drive. Then, it will 'swap in' the requested data into the now free area.
- Usually, the least recently used stored data is swapped out.

If data is swapped out and then is needed again, it is swapped back in from the swap area, at the expense of other data.

There are disadvantages to using virtual memory:

- The read/write speed of a hard drive is much slower than RAM, and the technology of a hard drive is not made for accessing small pieces of data at a time.
- If the system has to rely too heavily on virtual memory, there will be a significant drop in performance.
- Often the operating system has to constantly swap information back and forth between RAM and the hard disk drive, which operates all of the time. You can hear the disk drive operating continuously. This is called '**disk thrashing**' and it slows down the execution of the programs.

The more RAM a computer has, the less virtual memory will be needed. This is why inserting more RAM will improve the performance of a computer. The size of the area on the hard disk drive to be used for virtual memory can be set by the user.

GENERAL VOCABULARY

swap to change one thing for another

SUBJECT VOCABULARY

disk thrashing a very high rate of hard disk access

SKILLS REASONING

ACTIVITY 10

TYPES OF MEMORY

- 1 What is cache memory used for?
- 2 What is the difference between RAM and ROM? Suggest a use for ROM.
- 3 Why would increasing the amount of RAM improve the performance of a computer?
- 4 Make a cartoon strip or storyboard showing the fetch–decode–execute cycle.
- 5 Look back at the photographs and descriptions of the ZX81 and Raspberry Pi. Working with a partner, identify the similarities and differences between them. After 35 years, what has changed and what is the same?

FETCH–DECODE–EXECUTE
IN DETAIL

There are many different kinds of CPU, but they share some features in common. To understand the fetch–decode–execute cycle in more detail, you need to know a bit more about what goes on inside the CPU.

SUBJECT VOCABULARY

arithmetic/logic unit (ALU) the part of the CPU that performs calculations and logic operations

register a storage location inside the CPU used to hold an instruction, an address or other single item of data

control unit the part of the CPU that organises the actions of the other parts of the CPU

clock an electronic device inside a CPU that 'ticks' at regular intervals and is used to synchronise the actions of the other parts of the CPU

gigahertz (GHz) a measure of frequency equivalent to 1000 million cycles per second

bus width the number of wires that make up a bus. This determines the range of binary numbers that can be communicated

The part of the CPU that does calculations and logic operations is called the **arithmetic/logic unit (ALU)**.

Inside the CPU are a number of memory locations called **registers**. These are extremely fast to access but there are usually only a relatively small number of them. Some registers play a specific role in the fetch–decode–execute cycle, for example, holding a memory address, an instruction or a piece of data, while others are general purpose. Some of the registers with a specific role that are found in most computers include the following.

COMMON REGISTERS

Accumulator: this holds the results of calculations performed by the ALU. All input and output from the CPU pass through the accumulator.

Program counter: this holds the memory address of the next instruction to be fetched.

Current instruction: this holds the instruction currently being executed.

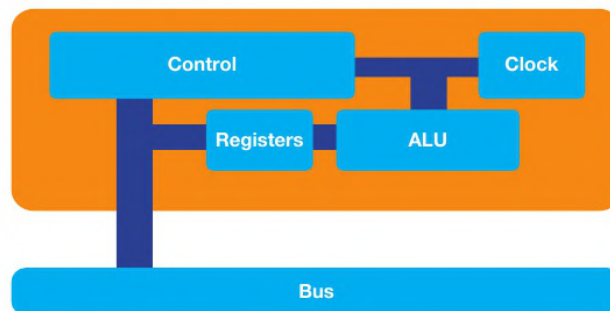
The steps in the cycle are controlled by a **control unit** and synchronised by an electronic **clock**. You have probably seen the advertised clock speeds of CPUs; for example, a 2.2 **gigahertz (GHz)** CPU means that its clock 'ticks' 2200 million times per second.

► Figure 4.9 Clock signal



The bus that connects the CPU to other devices in the computer is split into three parts: the address bus, the data bus and the control bus. The address bus carries memory addresses between the CPU and memory to identify a memory location; the data bus carries the value to be read from or written to memory; and the control bus carries the signals that, for example, determine whether to read or write the data, and when to do so.

► Figure 4.10 Inside a CPU



The number of connections on a bus is called the **bus width**. Since each connection represents a binary digit (a 1 or a 0), a greater bus width means larger number values can be communicated. For example, an 8-bit address bus can send values from 00000000 (0) to 11111111 (255), i.e. only 256 possible addresses. By contrast, a 32-bit computer can address up to 4 GB of RAM.

Now that you know a little more about the parts of a CPU, you can understand in more detail what happens at each step of the fetch–decode–execute cycle.

FETCH–DECODE–EXECUTE CYCLE

Fetch

The CPU control unit places the memory address of the next instruction on the address bus. It also sends a signal on the control bus requesting to read from memory. The memory receives the signal and looks up that memory location. The data in the memory is copied on to the data bus. The CPU copies this into a special register.

Decode

The control unit analyses the contents of the register and sends signals to the other parts of the CPU telling them what to do (e.g. add numbers, store data back into the memory).

Execute

The instruction is completed by the CPU.



DID YOU KNOW?

You are probably wondering how you get computer games just from copying, adding and storing binary numbers. First, remember that the I/O devices are also connected to the bus and they have their own addresses; storing numbers in their addresses can make pixels appear on a screen or sounds come from a speaker. Second, remember that modern CPUs can do billions of these simple instructions per second. What appears to be very complex behaviour is actually just a very large number of very simple operations.

FACTORS THAT AFFECT CPU PERFORMANCE

Everyone wants their computers to work faster and faster. Manufacturers have continued to increase the speeds at which computers work, but there are many factors that affect CPU performance.

CLOCK SPEED

The rate at which instructions are processed by the CPU is controlled by the clock speed. The faster the clock speed, the faster the rate of processing. Or, to put it another way, a new instruction will start processing on a clock tick: the more clock ticks there are, the faster the instructions will be processed. Clock speeds have increased, and a rate of 3 GHz is common in modern computer processors. However, increasing the clock speed to increase processing speed has disadvantages:

- The instructions are processed by transistors and the rate at which they operate is limited.
- The processor generates a large amount of heat and this increases as the clock speed increases. A fan and heat sink help to get rid of excess heat and prevent it from breaking or even melting. However, there are limits to the rate of cooling.
- Processors with clock speeds of 9 GHz require cooling by **liquid nitrogen**.

GENERAL VOCABULARY

liquid nitrogen nitrogen gas that has been cooled down to below -195.79°C so that it is a liquid

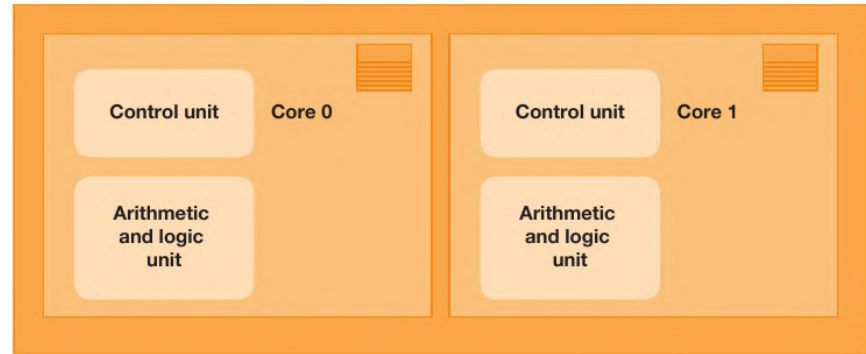
GENERAL VOCABULARY

core the part of something that is central to its existence or function

NUMBER OF PROCESSOR CORES

Manufacturers introduced **multicore processors** in 2006 to increase processing speed. A multicore processor has more than one processor **core** (but only one CPU). The following diagram illustrates the structure of a **dual-core processor**.

► **Figure 4.11** The structure of a dual-core processor



The following table shows the names given to processors having different numbers of cores.

NUMBER OF CORES	COMMON NAME
1	Single-core
2	Dual-core
4	Quad-core
5	Penta-core
8	Octa(o)-core
10	Deca-core

SUBJECT VOCABULARY

multicore processor a processor with many processor cores

dual-core processor a processor with two processor cores

single-core processor a processor with only one processor core

The advantages of multicore processors over **single-core processors** are that the cores can work:

- together on the same program – this is called parallel processing
- on different programs at the same time – this is called multitasking.

However, not all programs will run at twice the speed with a dual-core processor. The tasks required might not be able to be carried out in parallel. They might be sequential, so that one task requires output from a previous task. This means the second task cannot start until the first has finished.

SIZE OF CACHE

Bottlenecks occur when one component cannot work as fast as other components and so slows down progress. In the fetch–execute cycle, this bottleneck is caused by the main memory that is used to store the

SUBJECT VOCABULARY

cache a temporary data store so that the data can be accessed very quickly when needed

static RAM (SRAM) memory that retains data bits in its memory as long as power is being supplied and does not have to be refreshed

instructions and data. Although it is far quicker to fetch the instructions from the main memory than from a hard disk drive, the RAM is still far slower than the CPU. The instructions then have to be transferred through the data bus. The speed of processing will be limited by the RAM, which supplies the instructions regardless of how much clock speed is increased or how many cores are used.

The solution to this bottleneck problem is to use faster memory very close to, or even within, the CPU. This memory is used to store recently used data and data likely to be frequently used (which is called a **cache**).

Most CPUs have independent instruction and data caches. The data caches have to be read and written to, but the instruction caches just have to be read by the CPU. The use of caches allows the CPU to check the fast cache for the data it needs. It does not have to wait for it to be fetched from the slower main memory. The faster **static RAM (SRAM)** is used for the cache.

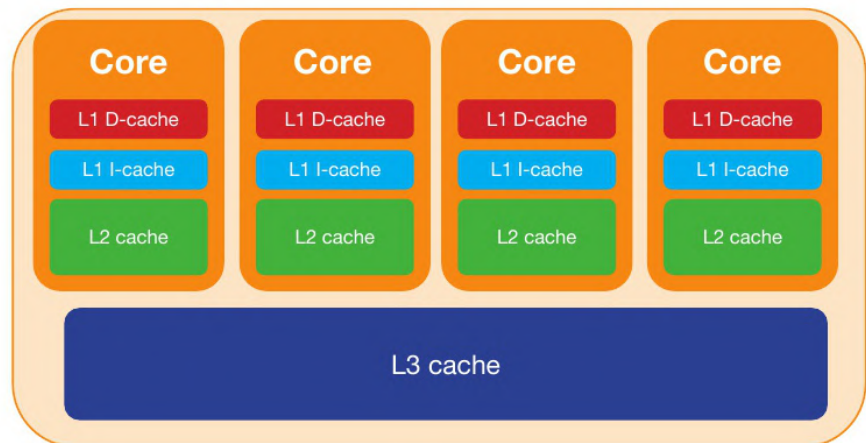
The caches are located on the processor chip. The fastest is the Level 1 cache and is smaller than the Level 2 and Level 3 caches.

The Level 1 cache is checked first, followed by the Level 2 and then Level 3 caches. In a multicore processor, the cores have their own L1 and L2 caches while the Last Level cache is usually shared by all the cores.

With a larger cache there is a greater probability that the instruction or data item to be fetched is in the cache and so the RAM will not need to be accessed. This will speed up processing.

The sizes and positions of the caches are shown in the following diagram.

► **Figure 4.12** The size and positions of caches



SKILLS

REASONING,
CRITICAL THINKING,
COMMUNICATION

ACTIVITY 11

IMPROVING CPU PERFORMANCE

In groups of three, stage a debate on the best way to improve CPU performance. One person should research and speak on behalf of clock speed, another cache size and the third on the type and size of RAM.

SKILLS REASONING

ACTIVITY 12

QUAD-CORE

A computer is advertised as being 'quad-core'. What is meant by 'quad-core'? How does it improve the performance of the computer? How does using cache memory also improve the performance?

SECONDARY STORAGE

SUBJECT VOCABULARY

secondary storage any kind of permanent storage to which the contents of ROM/RAM are copied (usually a hard disk, optical or solid-state device)

magnetic storage secondary storage that works by making parts of a substance behave like a magnet, with north and south poles to represent binary 1s and 0s

optical storage secondary storage that works using differences in light reflection from a material

solid-state storage secondary storage that works by storing charge (electrons)

platters rigid, rapidly rotating, magnetically coated disks within a hard disk drive

latency refers to any kind of delay that data travelling through a network might encounter

GENERAL VOCABULARY

reflective used to describe a surface that sends back (reflects) light or heat

In most situations, you need to be able to store your data and programs after the power is turned off. You have learned that RAM is volatile. This means most computers need to be able to copy the contents of their RAM to another kind of storage that is not volatile – a type that doesn't lose its contents when there is no power. This more permanent storage is called **secondary storage**. Secondary storage is non-volatile and, compared to RAM, is slower to access, cheaper and has much higher storage capacity. Typical capacities are now in the terabytes, i.e. millions of megabytes.

You remember that all data is stored as binary – 1s and 0s. There are three physical ways of recording 1s and 0s that do not require power.

Magnetic storage uses the fact that magnets have north and south poles. By making something behave like a magnet, the north and south poles can represent the 1s and 0s of your data. This is used in hard disks and magnetic tape storage.

Optical storage is used by CDs and DVDs. Shinier or more **reflective** parts of the disk represent the 1s or 0s.

Solid-state storage or 'flash' memory (such as USB memory sticks or SD cards) represents the 1s and 0s with little pools of trapped electrons on a microchip.

MAGNETIC SECONDARY STORAGE – HARD DISKS

Inside a hard disk drive is a stack of disks called **platters** with a magnetic coating on each surface. Tiny magnetic recording heads on the end of an arm float a millionth of a centimetre above the disk spinning at 110 km/h underneath.

Data is recorded on each disk along circular tracks, each split into smaller parts called sectors. When data is read:

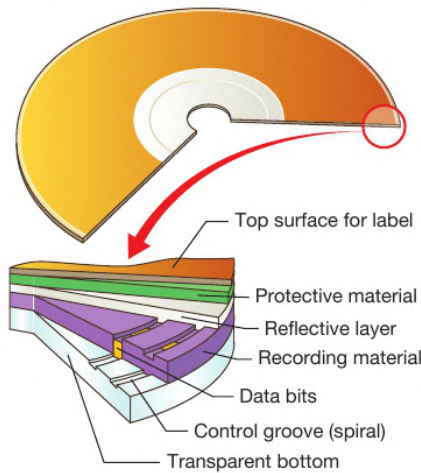
- the arm moves across to be above the right track
- the required sector comes around under the head
- the surface behaving like a magnet causes a tiny current in the head
- the disk controller translates this into 1s and 0s.

Because of these steps, data does not come from the disk immediately. Each step takes some time. The time for step 1 is called the seek time; for step 2 it is called the **latency**.

OPTICAL SECONDARY STORAGE – DVDS

A CD, DVD or Blu-Ray™ disk is made of several layers as shown in Figure 4.13. Data is written along a single track that moves out from the centre of the disk in a spiral.





▲ Figure 4.13 The structure of a DVD

The player uses two lasers: one to write data on to the disk and the other to read data.

When writing, one laser heats the recording material. This creates non-reflective pits on the surface of the disk, where the surface becomes lower than the parts around it. This is slow because the heating and cooling of the surface takes time for each bit of data.

If a CD is already written when it is manufactured, it has pits in the recording layer that are less reflective than the flat parts, called lands.

When reading, the laser reflects differently off the surface and this is detected by a light sensor.

The spiral track is longer on the outside so when reading data from the outside edge the disk has to reduce its rotation speed so that the data passes under the laser at a constant speed.

When data is read:

- 1 The disk spins in the drive to ensure all data can be read.
- 2 The tracking mechanism moves the laser into the correct position over the disk.
- 3 The laser shines on to the disk and is reflected back on to a light sensor.
- 4 Signals from the sensor are translated into 1s and 0s.

ELECTRICAL SECONDARY STORAGE – FLASH DRIVES

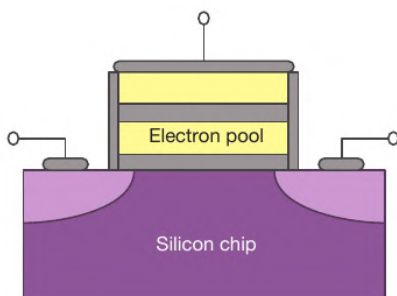
Solid-state storage uses chips (called NAND flash) made of special kinds of transistors that can trap electrons in a 'pool'. Electrons in a pool represent the 0s of the data, while empty pools represent 1s.

Because billions of transistors will fit on a chip, solid-state devices are small and have high capacity (often several gigabytes), as you may know from any memory sticks you have. They also give much faster access to data than hard disks.

When data is read from the chip:

- 1 Control signals identify which bit is to be read out and apply a small voltage.
- 2 If the electron pool is empty the transistor turns on and a 1 is read out.
- 3 If the electron pool is full the transistor doesn't turn on and a 0 is read out.
- 4 The control signals are changed to read other bits.
- 5 When data is written to the chip, control signals identify which bit is to be written and apply a higher voltage.
- 6 This pulls electrons into the pools of those transistors, recording the 1s and 0s.

Erasing data also requires higher voltages to remove electrons from the pools. Because of this, both erasing and writing cause the transistor to break down slowly. This means that flash drives can only be rewritten around 1 million times before eventually failing.



▲ Figure 4.14 Flash drive

Both magnetic hard disks and solid-state flash memory are developing rapidly and will continue to increase in capacity and decrease in size. The largest computer companies are currently researching some brand-new technologies that might become important in the future. For example, HP® is testing a new type of electrical component called a memristor to create super-fast, high-capacity, non-volatile memory chips. IBM® is developing racetrack memory – a type of magnetic storage that stores information as magnetic patterns on tiny wires, providing much faster data access speeds than current hard drives or flash disk technologies are capable of. You might also have heard of quantum computing. Quantum computers store data as qubits. Qubits can be individual atoms and each one is capable of storing (and processing) more than 1 bit at once.

SKILLS

REASONING,
CRITICAL THINKING,
DECISION MAKING,
COMMUNICATION

ACTIVITY 13

SECONDARY STORAGE

- 1 What is the purpose of secondary storage?
- 2 List one kind of magnetic storage and one kind of solid-state storage.
- 3 Mohammad runs his own gardening business. Which kind of secondary storage would you advise Mohammad to use for:
 - a storing his garden design software?
 - b taking garden designs to show potential customers?
 - c backing up all his financial records for seven years?
- 4 Carry out research into recent and future developments in data storage.
- 5 If you have an old hard disk you no longer need, try taking it apart to look at some of the components you have learned about. Your tutor might have one to show you or you could ask your school's IT technician.

CLOUD STORAGE



Sometimes it can be useful to have your secondary storage in a different place from your computer. For example, you might want to have a back-up copy of your data in a secure location or share your data with users of other computers; or you might want to be able to access your data when you are not sitting at your usual computer.

In Unit 5, you will learn more about how networks can be used to share computing resources. Computers that share their secondary storage across a network are called file servers. You are probably used to using the secondary storage on a server on your school network (your home drive or a learner shared area). Some hard-disk drives are able to connect directly to a network (called NAS or network-attached storage). Others connect to several servers using their own storage area network or SAN. This type of arrangement is useful when it's important not to lose access to your data, for example in a banking system. If one server or hard disk fails, another can take over its job without users of the system even noticing.

SKILLS

RESEARCH,
CRITICAL THINKING

SUBJECT VOCABULARY

cloud storage secondary storage, often belonging to a third party, that is accessed via a network, usually the Internet, and so is not in the same physical place as the machine's RAM/ROM. Files stored 'in the cloud' can be accessed from anywhere via an Internet connection

virtualisation any process that hides the true physical nature of a computing resource, making it look different, usually to simplify the way it is accessed

ACTIVITY 14

RESEARCHING YOUR NETWORK

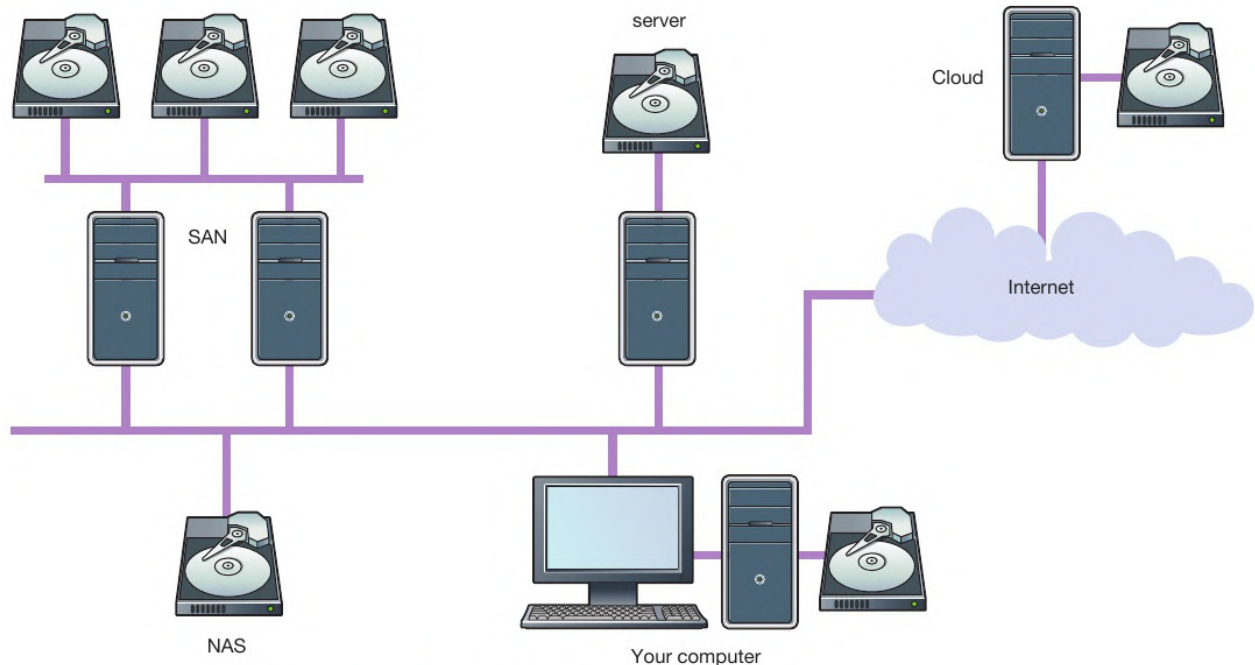
Find out about the storage systems used on your school network. Research, for example, what 'drives' are available in My Computer. Does your school use a SAN or NAS? Where are the servers located? How does this compare to using a service such as Google Drive®? Write a short report giving the advantages and disadvantages of different systems.

Sometimes storage is accessed via the Internet using Internet services such as Dropbox® or Google Drive. In this situation it is usually called **cloud storage**. This is an example of **virtualisation**. You seem to have access to a single, virtual hard drive when, in fact, your storage might be spread across several servers around the world, with software dealing with the details of where each part of each file is actually stored.

Storing data on servers accessed via the Internet has the following advantages:

- You can access the data from anywhere on many devices using a web browser.
- The data is securely backed up by the company providing the storage service.
- You don't need to transfer your data if you get a new computer.

Some people will be concerned about security: you might, for example, have read about high-profile cases of private photographs being accessed and released on the Internet. Other people might be uncomfortable with not being able to have their data 'in their hand'. They might be worried about losing access if they lost their Internet connection. See Unit 5, pages 225–226, for more on cloud security.



▲ Figure 4.15 Cloud storage diagram

EMBEDDED SYSTEMS

An embedded computer system is one that is designed to do a specific job. It is usually part of a bigger system that might have other electrical or mechanical parts, such as a washing machine. This makes them distinct from general-purpose computers like the PC or laptop you use every day. Embedded systems are all around us in every part of daily life, for example in your cable or satellite TV box, car, games console, traffic lights, toaster or toys. They might be so simple that you don't even think of them as computers.

Embedded computers are often cheap, low-power devices using simple technology and limited memory. They might run just one program, which is very rarely updated, and so is stored permanently in firmware (ROM or flash memory). They often use sensors of some kind.

SKILLS CRITICAL THINKING

ACTIVITY 15

EMBEDDED SYSTEMS

- 1 Modern cars contain an engine management system. This is an example of an embedded computer system.
 - a Suggest two inputs to and two outputs from this system.
 - b Outline one function of the engine management system.
- 2 Make a table listing five other embedded systems you might see in your daily life. You could look back to page 160 at the beginning of this unit to get some initial ideas.

GENERAL VOCABULARY

converge move towards the same point

SUBJECT VOCABULARY

Internet of things (IoT) the interconnection of digital devices embedded in everyday objects

In recent years, wireless network technology, the Internet and embedded systems have come together (**converged**). Many devices can now communicate with each other and be accessed and controlled remotely via the Internet. This has become known as the '**Internet of things**'. For example:

- your printer in school might automatically order new toner when it is getting low
- your electricity meter might send readings straight to the supplier via the Internet
- you might be able to control your central heating from your smartphone.

Imagine a future where your fridge knows when you're running out of milk and sends you a text telling you to pick up some more on the way home.

SKILLS ADAPTABILITY, CRITICAL THINKING, PROBLEM SOLVING

ACTIVITY 16

HOW EMBEDDED SYSTEMS WORK

- 1 Simulate a traffic light control system by writing a computer program. It must cycle through the normal sequence of lights, printing out the current state. Your tutor might give you the opportunity to use your program to control a real light output.
- 2 Draw a flowchart showing the processing required by a chip-and-PIN credit card reader. This is used when you pay for something in a shop. It collects data about the purchases from the till, gets your card details, checks your PIN, sends encrypted data to the bank and waits for a reply. It might print a receipt if the purchase is allowed and informs the till of the result.

CHECKPOINT

Strengthen

- S1** Outline in detail the fetch–decode–execute cycle. Include the roles of the memory, buses and special registers.
- S2** What are the three factors that affect the performance of the processor?
- S3** What are the advantages and disadvantages of different kinds of secondary storage?
- S4** What is meant by the term 'Internet of things'?

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS DECISION MAKING

SKILLS CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** REASONING**SKILLS** CRITICAL THINKING,
REASONING**SKILLS** REASONING**Challenge**

- C1** Can you describe the operation of a von Neumann architecture computer?
- C2** Can you compare the features of CPU registers, cache memory and RAM?
- C3** How does cache memory affect the performance of the processor?
- C4** What is the purpose of secondary storage? Justify an example of a situation where optical storage is the best choice.
- C5** Conduct your own research into the use of embedded systems to produce 'driverless cars'.

How confident are you in your answers to these questions? If you are not sure that you answered them well, go back and review the fetch–decode–execute section on pages 168–170. Try to explain this to a family member at home. This will help you to understand it yourself.

SUMMARY

- Von Neumann computers process data using a stored program.
- A CPU runs a program by accessing main memory using the fetch–decode–execute cycle.
- Main memory can be volatile RAM or non-volatile ROM.
- Cache memory is used between devices that communicate at different speeds.
- Virtual memory is memory that appears to exist as main storage although it is held in secondary storage, transfer between the two being made automatically as required.
- CPU, memory and I/O devices are connected using a bus.
- Inside the CPU, registers provide storage and the ALU performs calculations.
- The performance of the CPU is affected by the clock speed, the number of processor cores, the size of the cache and the type of cache.
- Secondary storage can be magnetic, optical or solid state.
- Secondary storage can be located away from your computer in the cloud.
- Embedded systems are cheap, low-power computers that are dedicated to a specific task.

18 LOGIC

Algorithms and programs are based on logic, which consists of rules that should be used when thinking about ideas. You have already met some logic statements such as **IF...THEN** and operators such as **AND** and **OR**. In this section, you will also look at truth tables to evaluate the output expected from different inputs.

LEARNING OBJECTIVES

- Construct and interpret truth tables for a given logic statement (AND, OR, NOT)
- Produce logic statements for a given problem

TRUTH TABLES

You learned in Unit 1 (pages 12–13) about selection – determining which parts of a program run, depending on certain conditions – usually expressed using **IF** statements.

```
IF health <= 0 THEN
    SET game_over TO true
END IF
```

In general, this kind of statement has the form:

```
IF CONDITION THEN COMMANDS
```

GENERAL VOCABULARY

condition something that must happen before something else can happen

command an order to a computer

When this code is translated the logic statement or **condition** is calculated. If the result is TRUE then the **commands** will run.

Here is a more complicated condition.

```
IF health <= 0 AND lives = 0 THEN
    SET game_over TO true
END IF
```



You can probably work out from this code when the game is over because you will have played games and understand what 'health' and 'lives' mean. But how does a computer do this? A computer needs to be able to calculate **AND** to work out if the condition is TRUE or FALSE.

In the example, the game is only over if both parts of the condition are true. We can show all the possibilities for this condition in a table (see Table 4.1).

► Table 4.1 Truth table for the AND operator

HEALTH <= 0	LIVES = 0	GAME OVER?
No	No	No
No	Yes	No
Yes	No	No
Yes	Yes	Yes

SUBJECT VOCABULARY

truth table a table showing all possible combinations of the inputs and outputs of an operator

Boolean something that can take only the values True or False; named after English mathematician George Boole

This kind of table is called a **truth table**. It is the truth table for the **AND** operator. The outcome is only true if both of the conditions are true. (Usually you will see a 'Yes' written as a '1' and a 'No' written as a '0'.)

Perhaps the game doesn't have lives and we only want to end the game if you run out of 'health' or you pass the winning score of 1 million.

```
IF health <= 0 OR score > 1000000 THEN
    SET game_over TO true
END IF
```

Here are the possibilities – this time it is the truth table for the **OR** operator (Table 4.2).

► Table 4.2 Truth table for the OR operator

HEALTH <= 0	SCORE > 1 000 000	GAME OVER?
0	0	0
0	1	1
1	0	1
1	1	1

In this case the outcome is true if either of the conditions is true.

INPUT	OUTPUT
0	1
1	0

There is one more logical operator (or **Boolean** operator) called **NOT** (Table 4.3). This operator just swaps 1s and 0s. Notice that, unlike **AND** and **OR**, **NOT** has only one input.

We could use this in our 'game over' example as follows.

▲ Table 4.3 Truth table for the NOT operator

```
IF (NOT god_mode AND health <= 0) OR score > 1000000 THEN
    SET game_over TO true
END IF
```

We can list all the possible combinations in a truth table as in Table 4.4. (Like in maths, brackets are evaluated first.)

GOD_MODE	NOT GOD_MODE	HEALTH <= 0	NOT GOD_MODE AND HEALTH <= 0	SCORE > 1000000	GAME OVER?
0	1	0	0	0	0
0	1	0	0	1	1
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	1	1
1	0	1	0	0	0
1	0	1	0	1	1

▲ Table 4.4 Truth table showing all the possible combinations

The three logic operations **AND**, **OR** and **NOT** are very easy to build as electronic circuits known as **logic circuits**.

SKILLS

REASONING,
CRITICAL THINKING,
PROBLEM SOLVING

ACTIVITY 17

LOGIC AND CONDITIONS

- 1 In words, write the condition for the game to be over in the final example in the table. (You will see why Boolean logic is clearer and easier to use than English when the situation gets complicated.)
- 2 Which part of the CPU works out the logic for conditions?
- 3 Create a computer program that asks for your year group, grade and target and then prints out whether you are invited to revision class or not. Start by assuming revision classes are only for Year 11 students whose grade is below target, then you can invent your own criteria for invitations to revision class.
- 4 Look at the statement below and write the truth table. This time there are three parts to the condition. Remember: as in maths, brackets are worked out first.

```
IF year = 11 AND (grade < target OR target > 7) THEN
    SET revision_class TO true
END IF
```

SUBJECT VOCABULARY

logic circuit an electronic circuit that has inputs and outputs that follow one of the Boolean operators

bug an error or flaw in a computer program

WRITING LOGIC STATEMENTS

When you write programs, you will often have to work out the correct logic statements for a given situation so that the right code runs. Mistakes in the logic (logic errors) are a frequent cause of **bugs**. So, it is important to check that the logic is right and that the code matches the logic you have worked out.



WORKED EXAMPLE

Mohammad has decided to expand his gardening business to give online advice. He is writing a web page that advises customers which fertiliser to buy from his online shop. The web page will ask customers about their plants and then give relevant advice using these rules:

- If plants have yellow leaves mainly near the soil, then they need a nitrogen fertiliser.
- If plants have brown leaves that are small, then they need a phosphorous fertiliser.

To program this, you first need to work out what are the important features in the rules – these will be your variables. In this case, the colour of the leaves is important. We can write the rules in pseudocode like this.

```
SET advice TO 'nothing needed'
IF colour = 'yellow' THEN
    SET advice TO 'nitrogen'
ELSE
    IF colour = 'brown' THEN
        SET advice TO 'phosphorous'
    END IF
END IF
```

Notice that we start off by assuming the plants are growing normally and only change the advice if certain conditions are met.

Mohammad points out that if leaves go brown, but are normal sized, then the customer needs potassium fertiliser. Now, the size of the leaf is also important.

```
SET advice TO 'nothing needed'
IF colour = 'yellow' THEN
    SET advice TO 'nitrogen'
ELSE
    IF colour = 'brown' AND size = 'small' THEN
        SET advice TO 'phosphorous'
    ELSE
        IF colour = 'brown' AND size = 'normal' THEN
            SET advice TO 'potassium'
        END IF
    END IF
END IF
```

SKILLS

CRITICAL THINKING,
PROBLEM SOLVING

ACTIVITY 18

ABSTRACTION

- 1 Add the following rules to Mohammad's website logic in the example.
 - If plants have cracked leaves or leaves that are the wrong shape, then they need a calcium fertiliser.
 - If plants have leaves that are yellow mainly on the tips then they need a magnesium fertiliser.
- 2 Why is Mohammad's advice an example of abstraction?



HINT

Note that we have used brackets in this example. You have learned about BIDMAS in your maths lessons and in Unit 1 (page 28). It tells you the order to do the sums. It is the same with logic operators: brackets go first, then **NOT**, then **AND** and finally **OR**. This is called **operator precedence**.

SKILLS

REASONING,
CRITICAL THINKING,
PROBLEM SOLVING

SUBJECT VOCABULARY

operator precedence the order in which you apply the operators (including logical operators) in a mathematical equation

WORKED EXAMPLE

Nadia has built her own programmable alarm clock. She wants to program the alarm to wake her up at the correct time depending on the day. She gets up:

- at 7.30 a.m. if it is a school day
- at 9.00 a.m. if it is the weekend or a school holiday, except on Saturdays in term time when she gets up at 8.00 a.m. to play hockey.

Let's work out the important features of these rules. The alarm times depend on whether it's term time or holidays and also on the day of the week. So, term time and day will be the variables for your conditions.

```
SET alarm TO '7:30 am'
IF term_time = false OR (day = 'Saturday' OR day = 'Sunday')
THEN
    SET alarm TO '9:00 am'
ELSE
    IF term_time = true AND day = 'Saturday' THEN
        SET alarm TO '8:00 am'
    END IF
END IF
```

ACTIVITY 19

CHANGING CONDITIONS

- 1 Why would the code for Nadia not work if the last two lines were swapped around?
- 2 How can you change the conditions so that it works with the code either way round?

SKILLS

REASONING,
CRITICAL THINKING

ACTIVITY 20

UNDERSTANDING LOGIC STATEMENTS

- 1 Work out what happens for the following statement.

```
IF gender = 'female' AND (subject = 'computer science'
OR subject = 'physics') THEN
    SEND "Superstar!" TO DISPLAY
END IF
```

- a for a girl studying French
 - b for a girl studying computer science
 - c for a boy studying physics.
- 2 Which answer would be different if the programmer had forgotten the brackets?

SKILLS PROBLEM SOLVING,
CRITICAL THINKING

SKILLS PROBLEM SOLVING,
CRITICAL THINKING

SKILLS REASONING,
CRITICAL THINKING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING,
REASONING

CHECKPOINT

Strengthen

S1 Create a truth table for the **AND** operator.

S2 Produce a logic statement for the operation of headlamps on a car – the headlamps come on if the light sensor reading is dark or if the headlamp switch is turned on.

S3 Produce a revision summary of this chapter.

Challenge

C1 Create a truth table for the expression $P = (A \text{ AND } B) \text{ OR } C$.

C2 What combination of logic operations would produce an output of 1 only if the two inputs were different?

How confident are you in your answers to these questions? If you are not sure that you answered them well, go back and review the examples on pages 182–183.

SUMMARY

- **AND**, **OR** and **NOT** are called logical or Boolean operators.
- They are used in selection statements such as **IF**.
- A truth table shows all possible combinations of inputs and the output produced.
- The order of precedence is: brackets, **NOT**, **AND**, **OR**.

19 SOFTWARE

Computer hardware consists of items that we can see and touch such as keyboards, disk drives and monitors. Software consists of the programs that make the hardware function and also carry out tasks for the users.

SUBJECT VOCABULARY

software the set of programs run by a computer system

application software software that performs a task that would otherwise be done by hand, perhaps with pen and paper

operating system software designed for particular hardware and which manages other programs' access to the hardware

utility software software that does a useful job for the user that is not essential to the operating system and not the reason for using a computer in the first place

system software operating system and utility software

LEARNING OBJECTIVES

- Know what an operating system is and how it manages files, processes, hardware and the user interface
- Understand the purpose and functions of utility software (managing, repairing and converting files; compression; defragmentation; backing up; antivirus, anti-spyware)
- Understand how software can be used to simulate and model aspects of the real world
- Understand what is meant by system software and application software

Computer systems consist of both hardware and **software**. Software is the set of programs that run on a computer system. You can usefully divide software into two types. The first type is what you normally think of as programs, those that do a job for us that would otherwise have to be done by hand, such as writing a document or creating and using a spreadsheet. This is **application software**. The other kind of software is there to help us use the computer and the application programs, such as providing the user interface and tools to manage the hardware. This is **system software** and it is usually divided into **operating systems** and **utility software**.

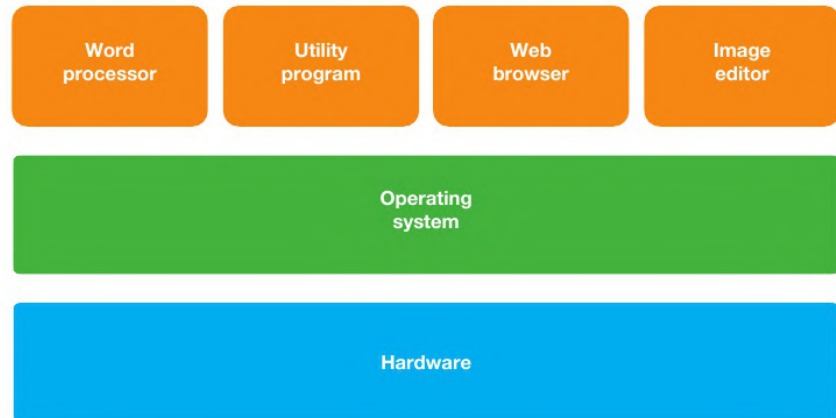
OPERATING SYSTEMS

The operating system (OS) is usually loaded when the computer starts up. You will probably already know several operating systems, for example Microsoft Windows®, Linux®, Android, iOS®, etc.

Why do you need this software? Unlike embedded systems, a general-purpose computer can be used to do a range of jobs. Software written for general-purpose computers needs to be able to run on a range of different hardware. For example, your favourite web browser runs on your laptop or desktop computer and also on your mobile phone or tablet. One of the jobs of an operating system is to manage whatever input and output devices the computer has. A programmer writing an application uses simple commands, e.g. `OpenFile`, and the operating system deals with exactly how to open a file. This will depend on what sort of secondary storage the file happens to be on. You might have realised this is yet another example of abstraction.

This system means applications have to be written for a particular operating system but will work on any hardware. You will probably already be familiar with this idea because you know, for example, that Android apps run on any Android device. Many different models of smartphone or tablet will run the app, but the same apps won't run on your friend's iOS phone.

► **Figure 4.16** The operating system sits between the hardware and the programs



SUBJECT VOCABULARY

application programming interface (API) code that allows two programs to communicate with each other
scheduling the algorithm that the OS uses to allow each running process to use the CPU
paging the algorithm the OS uses to move programs from RAM to disk and back again when needed once main memory is full

GENERAL VOCABULARY

illusion pretence

DID YOU KNOW?

The operating system provides developers of application programs like word processors with a set of commands such as `OpenFile` or `DisplayJPEG`. This set of commands is usually called an **application programming interface** or API. Using the operating system API rather than accessing hardware directly means that an application program will run on the operating system. It doesn't matter what hardware it is installed on. It also means that the operating system controls what application programs are allowed to do. This helps to reduce the risk of programs crashing and to maintain security.

In general, managing files and the directory structure, and input/output, is one of the important jobs of the operating system. An operating system also shares access to the hardware among the different programs that are running (and on some operating systems among the different users who are logged on to the computer). Two important pieces of hardware that have to be shared are the CPU and RAM.

- In the chapter on hardware, you learned that a von Neumann system does only one thing at a time. When you have several applications open, they can't really run at the same time. The operating system keeps up this **illusion** by allowing each program to use the CPU for a short time before switching to the next program. This is called **scheduling**.
- It is possible to load many applications all at the same time and each uses some memory. What if you keep loading applications until RAM is full? In this case the operating system creates another illusion, by moving programs from RAM to disk and back again when needed. This is called **paging**, swapping or virtual memory. You might notice this as a lot of disk activity. Your computer might run more slowly when lots of applications are open.

As well as the applications you are running, the operating system is also running other programs in the background. They include various parts of the operating system itself such as the programs that manage the network, the scheduler and also any hardware drivers and utility programs such as anti-virus. Each task that is running is called a process. To see all the processes that are running, including the invisible ones:

- on Microsoft Windows, press CTRL-ALT-DEL and choose Task Manager
- on Linux, type `ps aux` at the command prompt
- on OS X, run the Activity Monitor from Applications/Utilities.

SUBJECT VOCABULARY

concurrent processes that run at the same time are described as being concurrent

authentication the process of proving to a computer system who you are (e.g. using a username and password)

user interface the way the user interacts with the operating system

Information about how much the hardware is used by each process is displayed, including:

- the proportion of the CPU time spent on this process
- the amount of memory used by this process
- the amount of available memory (real memory and virtual memory, VM).

These processes are running one at a time, being given their share of the CPU time by the scheduler. They are all running alongside one another, but not really at the same time, so we say they are running **concurrently**. An operating system that allows processes to run concurrently is called a multitasking operating system. Almost all modern operating systems work this way.

As well as allowing multitasking, some operating systems allow different users to use the hardware. In some cases, only one user can use the hardware at once (single user). In others, several users can be sharing the hardware (multi-user). Your smartphone is an example of a single-user system, while your school network servers will be running a multi-user operating system. This is so that many of you can log on at the same time. In a multi-user system, each user has to identify themselves (with a username, fingerprint, etc.) and provide a password (or other login) to prove they are who they say they are. This is called **authentication**. Authority to use the facilities of the computer system is based on who you are. For example, learner users might not be able to print, while tutor users can. In such systems there is usually also a super-user (sometimes called administrator or root) who can access all resources and also control the access rights of other users.

The operating system also provides and runs the **user interface**. Sometimes this is a graphical user interface (GUI) with windows, icons, menus and pointers (WIMP) like Microsoft Windows or MacOS. Sometimes it is just a text, command-line interface (CLI) like MS DOS.

The choice of user interface depends on the applications a computer is expected to run. An embedded system might not have an operating system as such: therefore, it does not have a user interface. A computer acting as a web server might only need a command-line interface because it is intended only ever to be managed by technicians. A computer used for video editing will almost certainly need a graphical user interface.

DID YOU KNOW?

In common with many school subjects, as you learn more about computer science you will discover that some details were missed out so that you could understand the key ideas first. At least in computer science you are used to the idea of abstraction. One such detail is the idea of a driver. The driver for a device is the software that really communicates with the I/O device on the computer's bus. It transfers data between the hardware device and the operating system in an agreed format – also known as the application programming interface (API). To find out more about drivers and operating systems search online for 'NDIS' – this is the API that is used for Microsoft Windows network drivers – or 'ALSA', the API for sound drivers on Linux.

An API is yet another example of abstraction. A software developer deals with the essential tasks like `OpenFile`, but the detail of how data is taken from secondary storage is hidden, managed by the operating system.

SKILLS

RESEARCH,
REASONING,
CRITICAL THINKING

ACTIVITY 21

OPERATING SYSTEMS AND SCHEDULING

- 1 Run/open the program that lists the running processes on an operating system you are familiar with. Make a table listing the application software, utility software and operating system processes that are running.
- 2 There are several different algorithms that operating systems can use for scheduling. The one described on page 189 is called 'round-robin' scheduling. Research and describe at least one other method used for scheduling.
- 3 Take one of the programs you have already written. Add a feature to save or load data using the operating system API.

UTILITY SOFTWARE

ACTIVITY 22

SKILLS

CRITICAL
THINKING

UTILITY SOFTWARE

Make a list of all the utility software you can find on your computer. Try to organise your list into groups or categories such as 'file management tools'. Pair with a partner and share your list. Make a 'problems' list of software that you cannot agree a category for. Finally, pair with another couple of learners and compare their problem list with yours.

Utility software is software that does a useful job for the user. It is:

- not essential to the operating system
- not the reason for using a computer in the first place (i.e. it is not application software).

Utility software, sometimes called tools, can be split into three areas:

- basic tools
- file management
- security.

BASIC TOOLS

Basic tools include things like a simple text editor (e.g. Notepad or nano), calculator, command prompt, hex editor and software for accessibility, such as for producing large print. They are usually included as part of the operating system.

FILE MANAGEMENT TOOLS

No matter how much storage you have, somehow you always manage to fill up the space with files. This brings problems such as not being able to find your favourite photos, losing files if there is a fault and things simply running slower on your computer.

File management tools include software to keep your data secure by making a **back-up** copy of your files in another location. Some programs can recover files that have been deleted or repair files that might be damaged or have errors.

Another useful file management tool is for converting files between different formats. This is especially useful for **multimedia** files where there are a large number of common formats (e.g. jpg, gif, png).

SUBJECT VOCABULARY

back-up a copy of files in another location so that they are still available if the original copy is damaged or lost. Backing up is the process of making a back-up copy

GENERAL VOCABULARY

multimedia consisting of more than one medium, e.g. sound, text, images, animation, film

HINT

Be careful not to confuse compression with encryption.

SUBJECT VOCABULARY

defragmenter a utility that moves file clusters on a disk so they are closer to each other in order to speed up disk access

virus software that is designed to make and distribute copies of itself. This is usually for a **malicious** purpose

spyware software, possibly a virus, that is designed to be installed secretly on a computer. It records private information as the user enters it

firewall a utility that controls program access to the network, both incoming and outgoing

GENERAL VOCABULARY

cluster a group of things together that are of the same type

malicious intentionally harmful

Another important file management tool is a **defragmenter** which is used to speed up access to data stored on magnetic hard disk drives. Files are not stored on disk in one large chunk. They are stored in smaller pieces called sectors or **clusters** and a single file's clusters can be spread across the disk. Accessing these fragmented files can be slow, so a defragmenter moves the bits of files around, so they are closer together, speeding up disk access (at least in theory). Note that, unlike magnetic disk drives, solid state/flash drives do not require defragmentation (i.e. what is done by a defragmenter) as access times are the same across the entire volume.

You can also use a file management tool to compress the files you don't use very often. This frees up space on the disk, but those compressed files are then slower to access as they have to be uncompressed to open them.

SECURITY TOOLS

Security tools include anti-malware programs such as antivirus and anti-**spyware** software. Computer viruses and spyware can be used to capture private information, to introduce a process on your computer to do a job for the attacker, such as send out spam email, or to crash an important machine such as a web server. Some are simply written to annoy users. This software is called a virus because it copies itself and spreads: antivirus tools try to identify code that copies itself and prevent those processes from running.

Firewall software controls your connection to a network, deciding which network data can flow to and from your computer. This can help to prevent infection by viruses.

Encryption software is used to make files unreadable without a password. Data on your tutor's laptop might be encrypted so that it is not understandable if the laptop is stolen.

SIMULATION AND MODELLING

In Unit 1 you learned about how you can use abstraction to model the real world. By now you will be getting used to the idea of abstraction as a central part of computer science. It means hiding complexity, only exposing the key features of a situation and leaving out the details.

Computer science allows us to do experiments that can't (or shouldn't be) done in the real world. For example:

- you can't increase the average temperature of the Earth to see what happens to our weather
- you shouldn't stop one of a pair of twins from exercising to see if it causes depression.

You can, however, write a computer program to model or *simulate* these situations and see what results you get. Scientists, engineers, economists and other experts can ask '**what if?**' **questions** using computer models. For example, what if the bridge were twice as long?

There are two big problems with this approach:

- The model or simulation includes assumptions. It is not reality, so the answers might not be right.
- The real world is far too complicated to allow for every possible factor in your model. So, you have to use abstraction to simplify it. This makes it even less likely that you will get a completely right answer.

SUBJECT VOCABULARY

'what if?' question running a computer model with a given set of inputs to see what the model produces as an output or prediction

SUBJECT VOCABULARY

heuristic a type of algorithm capable of finding a solution to a problem quickly and easily. This is done through **trial and error** and educated guesswork to cut corners and remove less likely alternatives. Heuristic algorithms don't always find the best solution, but they will usually find one that works

Monte Carlo method carrying out a statistical analysis of a number of random samples to get approximate solutions to a problem. The larger the number of samples used, the more accurate the result is likely to be

neural networks process information in a similar way to human brains and learn and adapt over time. This makes them useful tools for recognising faces, identifying illnesses and quality control, for example. Computers are normally not very good at these things

GENERAL VOCABULARY

atmospheric of the atmosphere surrounding the earth

trial and error a way of solving a problem by trying different methods and learning from mistakes

SKILLS REASONING,
PROBLEM SOLVING

However, computer models can reveal some interesting things about how the world works or about how humans work with machines.

EXAMPLES OF COMPUTER MODELS

A flight simulator allows pilots to train to fly aircraft safely. They can make mistakes in the simulator without endangering their own lives or the lives of passengers. Commercial flight simulators use a moveable cabin and real flight controls so pilots can see, hear and feel things just as if they were flying a real aircraft.

If you ever watch the weather forecast, you will sometimes hear the presenter talking about their computer and sometimes even about computer models. The **atmospheric** models used to forecast the weather are some of the most complex pieces of software that exist. They split the atmosphere up into small pieces and keep track of the temperature, pressure, humidity and other factors in each piece. They start the model with real measurements, then they let it calculate the effect each piece of atmosphere has on its neighbours and see how the weather patterns develop.

Another example of a computer model is used for earthquake prediction. Unlike the previous examples, we don't know the mathematics behind how earthquakes work. Models like these have to use information from past examples and probability to come up with best guesses as to what might happen and when. Some common techniques used in such models are **heuristics**, **Monte Carlo methods** and **neural networks**. There is a whole field of computer science devoted to finding patterns in data and using them to predict the future: this is machine learning.

ACTIVITY 23

CREATING A SIMULATION

- 1 Using a programming language that you are learning:
 - a create a simulation of rolling a die. The die must come up with 1 to 6 with equal probability.
 - b adapt your program so that the die is biased to give 6 more often than it should.
- 2 In a group, research techniques used in machine learning and illustrate your findings using presentation software. Bring your talks together and present them back to the class.

CHECKPOINT

Strengthen

- S1** What is the difference between systems software and application software?
- S2** Can you identify one operating system? List the main functions it performs.
- S3** What are the differences between command-line and graphical user interfaces?

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS REASONING

SKILLS REASONING

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS PROBLEM SOLVING

S4 Why is antivirus software utility software?

S5 Computer models suggest that if we continue burning fossil fuels at the current rate, average sea levels will rise 25 cm by 2050. What are the advantages and disadvantages of using a computer model for this problem?

Challenge

C1 What is meant by a process? What is the purpose of scheduling?

C2 Why might some users prefer a command-line interface to a graphical one?

C3 Develop a program to model the populations of rabbits and foxes in a habitat. Use the following information to help you.

- Foxes eat rabbits.
- Foxes die of starvation if they don't eat enough rabbits.
- Rabbits and foxes both reproduce, but rabbits reproduce faster.
- Rabbits and foxes both die of other causes.

How confident are you in your answers to these questions? If you are not sure that you answered them well, make a revision resource by researching operating systems, utility software and computer models online before attempting the questions again.

SUMMARY

- Application software is used to perform specific jobs for a user that they would perform by other means, e.g. word-processing software to write a report, spreadsheet software to manage finances.
- An operating system (OS) is software that:
 - manages a hardware device
 - manages users' access to the I/O devices, including files on secondary storage
 - provides the user interface
 - shares the computing resources such as CPU and memory between processes.
- Processes are tasks running on the computer, including application programs.
- Utility software does useful jobs for the user, such as file management.
- Computer modelling enables experts to ask 'what if?' questions without affecting the real world.
- Computer models are very useful but contain assumptions and simplifications so you can't rely completely on their results.

20 PROGRAMMING LANGUAGES

An algorithm can be written in any language that a human can understand. Unfortunately, computers can only understand instructions written as a series of 1s and 0s in a language called machine code. Machine code is too difficult for humans to use. So, we code programs in other languages, which can then be translated into machine code for the computer.

LEARNING OBJECTIVES

- Understand what is meant by high-level and low-level programming languages and understand their suitability for a particular task
- Understand what is meant by an assembler, a compiler and an interpreter when translating programming languages and know the advantages and disadvantages of each

LOW-LEVEL PROGRAMMING LANGUAGE

SUBJECT VOCABULARY

instruction set the list of all possible commands a particular CPU knows how to carry out

machine code the binary codes representing each of the instructions in the instruction set

translator a program that converts source code into machine code

source code the text of the program that a programmer writes

assembler translates the mnemonics of assembly language programs into machine language instructions for the microprocessor to carry out

assembly language a low-level language written using mnemonics

low-level programming language a programming language that is closely related to the CPU's machine code

You learned earlier that a von Neumann architecture computer uses the stored program concept. The instructions that the computer carries out, the program, are stored in memory just like any other kind of data. This means that programs are also just binary numbers – that's all you can store in memory. So how can binary numbers represent a program?

One other thing you learned is that the CPU can only do a few simple things. It only 'understands' a few very simple instructions. Complex things like playing games are possible because a really large number of those simple things happen very fast. Those few simple instructions that a CPU knows how to do are called the **instruction set** for that type of CPU. Each instruction is given a binary code. It is these codes that make up your programs. The binary codes representing a program are called **machine code**.

A machine code program would look something like the diagram below. You can probably see why programming in machine code would be completely impractical.

0110100100100101010101011111010101100101011010110101010

▲ Figure 4.17 An example of a machine code program

Not only would it be incredibly difficult to work out the sequence of 1s and 0s you need, but also your program would only work on that kind of CPU. This is because different CPUs have different instruction sets. Instead, you need a **translator** which converts your programs into the CPU's machine code language.

Translators are also programs. Their input is the text of your program – this is called your **source code**. Their output is machine code which the CPU can run (execute).

The simplest translator is called an **assembler**. It converts **assembly language** to machine code. Assembly language is called a **low-level programming language**.

SUBJECT VOCABULARY

mnemonic a short, simple acronym that represents each of the instructions in a CPU's instruction set, e.g. LDR (load register), STR (store) and CMP (compare)

emulator hardware or software that allows one type of computer system to behave like another

reboot to start a device again, e.g. turn off the computer and turn it on again

This is because the programmer is working at the lowest level of detail, the level of the CPU hardware itself. Each instruction in assembly language is the same as a machine code instruction.

Assembly language is a bit easier to work with than machine code because each instruction is written as a short, memorable keyword called a **mnemonic**. For example, the instruction for the CPU to add numbers together might be 01101001 in machine code but is just **ADD** in assembly language. The assembler replaces each mnemonic with the appropriate binary machine code and data. Here is an example of assembly code:

```
LDR R3, [R1]
MOV R2, #0
CMP R3, R0
MOVT R2, #1
```

DID YOU KNOW?

The 1980s space-trading game Elite, which featured 3D graphics and a galaxy of hundreds of planets to explore, was written entirely in assembly language and occupied just 22 KB. To understand how amazing that programming was, look at the file size of a modern 3D graphics game. To learn more, you could play an authentic version of the game on an **emulator**. Search online for 'BeebEm'.

Writing programs in assembly language is challenging for three reasons.

- A very limited range of instructions is available. Every task, even the simplest, has to be built up from the smallest steps. Some older CPUs could not even multiply numbers.
- You have to manage all your data. There are no strings, integers or real numbers, just binary, so you have to decide how to represent your data. You also have to decide and manage where it is stored in memory.
- Debugging is very difficult. When the assembled program runs, any bugs usually just make the machine crash and you have to **reboot** to try again.

Nowadays, almost no programs are written in assembly language. The exceptions are some hardware drivers and programs designed to run on embedded systems that you learned about earlier in the chapter. This is because writing in the CPU's home language can be useful if you want your program to be very small and very fast or to directly control the I/O hardware.

SKILLS REASONING

ACTIVITY 24

ASSEMBLY LANGUAGE

Make a list of tasks for which assembly language is better suited than a high-level programming language.

EXTEND YOUR KNOWLEDGE

Research the Little Man Computer simulation. Make a list of the machine code instructions that Little Man Computer 'understands'.

HIGH-LEVEL LANGUAGES

SUBJECT VOCABULARY

compiler a translator that converts high-level language source code into object code, often machine code. The source code is translated all at once and saved to be executed later

interpreter a translator that converts high-level language source code into object code, often machine code. The source code is translated and executed one line at a time

object code the translated source code. Often this will be machine code, but might also be an intermediate code, which has to be further translated before it can be executed

It is much more common to write software in high-level programming languages such as Python, C#, Java or Visual Basic. For these programs to run on the CPU, the source code has to be translated into machine code. This can be done all at once, and the finished machine code program saved and run later, one line at a time.

COMPILERS AND INTERPRETERS

A translator that translates the whole program in one go is called a **compiler**.

A translator that translates and runs your program one line at a time is called an **interpreter**.

A compiler converts high-level language to machine code and saves the output as a machine code program, sometimes called **object code**. This usually has the .exe file extension, short for 'executable', because the machine code can be executed or run.

An interpreter translates your source code one line at a time then runs it. There is no object code file, only your original source code.

► Table 4.5 Using compilers or interpreters – a comparison

INTERPRETER	COMPILER
Every computer that will run your program needs the interpreter software installed.	The output from a compiler will run on its own on any similar computer.
Interpreters find errors when they happen and can often tell you what has gone wrong.	A compiler cannot produce any object code unless the whole program is correct – they tend to report a lot of errors initially, making it harder to debug your program.
Programs tend to run slower using an interpreter because the interpreter has to translate the source code while the program is running.	It can be easier to protect your code from being altered or copied if it has been compiled because you only give people the object code (machine code), which is hard to understand.

SKILLS

PROBLEM SOLVING,
REASONING,
CRITICAL THINKING

ACTIVITY 25

GRAPHICAL VERSUS TEXTUAL LANGUAGES

Implement one of the algorithms you designed in Unit 1 in a graphical programming language (e.g. Scratch®), and a textual language (e.g. Python). Which language did you find most intuitive, i.e. which was easiest to use? Which one would you recommend for someone starting to program?

SKILLS CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** REASONING**SKILLS** CRITICAL THINKING**SKILLS** REASONING**CHECKPOINT****Strengthen****S1** What is meant by 'machine code'?**S2** What is the function of an assembler?**S3** What is the job of a translator?**S4** Momina is a C# programmer and uses a compiler. Denis is a Python programmer and uses an interpreter. What challenges will Momina face in developing software compared with Denis?**Challenge****C1** What is the difference between high-level and low-level programming languages?**C2** Evaluate the choice of translator between compilers and interpreters.

How confident are you in your answers to these questions? If you are not sure that you answered them well, you could review the chapter by searching online.

SUMMARY

- CPUs can only run programs written in machine code.
- A translator is a program that converts source code into machine code.
- Humans program computers mostly using a range of high-level languages, but sometimes in assembly language (a low-level language).
- An assembler is used to translate assembly language mnemonics into machine code.
- Compilers and interpreters are types of translator, each having their own advantages and disadvantages.

UNIT QUESTIONS

SKILLS CRITICAL THINKING A01

- 1 Name **three** output and **three** input devices of a smartphone. (6)

HINT

A touch screen is an input and output device so could appear twice in your answer.

SKILLS REASONING A01

- 2 Compare and contrast the sequential, parallel and multi-agent models of computation. (6)

HINT

Compare means to look for similarities and contrast means to emphasise the differences. You should therefore state the similarities and differences between these computational models.

SKILLS REASONING A01

- 3 The performance of a CPU can be improved by increasing its clock speed. Explain why performance cannot be increased indefinitely by increasing the clock speed. (2)

SKILLS REASONING, CRITICAL THINKING A01

- 4 Explain why sequential programs might not run quicker with multicore processors. (2)

SKILLS REASONING A01

- 5 State what is meant by the term 'volatile memory'. (1)

SKILLS REASONING, CRITICAL THINKING A01

- 6 Explain why having cache memory can improve the performance of the CPU. (2)

HINT

In Questions 4 and 6, remember that 'explain' means you have to give a reason as well as your answer.

SKILLS REASONING A01

- 7 Describe the steps in the fetch part of the fetch–decode–execute cycle. (2)

SKILLS REASONING, PROBLEM SOLVING A02

- 8 Yusuf is developing a logic circuit for an automatic door system.
- The output from the sensor on the street (S) is 1 when someone approaches the door from outside.
 - The output from the sensor in the building (B) is 1 when someone approaches the door from inside.
 - The opening mechanism works only if the manual switch (M) is off (0).
- Develop a suitable **IF** statement in pseudocode to represent this logic. (2)

SKILLS REASONING, PROBLEM SOLVING A03

- 9 Develop a pseudocode representing the logic for a cinema ticket pricing application. (3)
- Standard pricing is £5
 - Children under 4 are free.
 - Children under 16 and people over 65 are half price.

HINT

The operator precedence for **NOT**, **AND** and **OR** is likely to be tested in the exam question, so you need to know it. This means you should always check carefully whether you need brackets.

SKILLS REASONING A01

- 10 One of the functions of an operating system is scheduling.
- a Describe the purpose of scheduling. (2)
 - b Give three other functions of an operating system. (3)

SKILLS

REASONING,
DECISION MAKING

A01

A02

- 11 Manjit is using a high-level programming language to develop a computer game for children.

- a Explain why the program code she writes will need to be translated before it can be executed by a computer. (2)
b Compare using a compiler with using an interpreter to translate the code. (2)
c Explain why Manjit would not use an assembler. (2)

HINT

The command word in Question 11b is 'compare', so you need to talk about similarities and differences between a compiler and an interpreter. Make sure your answer relates to both.

SKILLS

REASONING

A01

- 12 Describe what is meant by the input–processing–output model. (2)

SKILLS

REASONING

A01

- 13 A user has to log in to a network. Describe this process in terms of the input–processing–output model. (2)

SKILLS

REASONING

A01

- 14 Omar has bought a laptop with a 2.2 GHz central processing unit.
a State the purpose of the CPU. (1)
b Describe what is meant by a 2.2 GHz CPU. (2)

SKILLS

REASONING

A01

- 15 List two registers of the CPU with specific functions in the fetch–decode–execute cycle. (2)

SKILLS

DECISION MAKING

A01

- 16 The performance of a CPU can be improved by increasing its clock speed. Explain why performance cannot be increased indefinitely by increasing the clock speed. (2)

SKILLS

REASONING,
DECISION MAKING

A01

- 17 a Describe the role of programs stored in ROM. (2)
b Describe the difference between volatile and non-volatile memory. (2)

SKILLS

REASONING

A01

- 18 Explain why secondary storage devices are needed. (3)

SKILLS

DECISION MAKING

A01

- 19 Explain why cloud storage may be preferable to local storage. (2)

SKILLS

PROBLEM SOLVING

A02

- 20 Complete the truth table below for the Boolean expression
 $P = \text{NOT}(A \text{ AND } B)$ (4)

A	B	P

SKILLS

CRITICAL THINKING

A01

- 21 A compiler or an interpreter can be used to translate a high-level language into machine code. State one advantage and one disadvantage of using each of these translators. (2)



UNIT 5

COMMUNICATION

AND THE INTERNET

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

In this unit you will learn about different types of networks, their importance, and how computers communicate with each other. You will also learn about the security threats to users' data and how they can be protected. The Internet, a huge network with billions of users is also considered, in addition to the hardware and software users need to connect to it.

21 NETWORKS

The networking of digital devices has had a profound impact on our lives and the ways in which we work, communicate with each other, buy products and services and find out information in our daily lives. In this section you will look at the different types of networks and the components required for their construction.

SUBJECT VOCABULARY

network an arrangement of computers and other devices connected together to share resources and data

network service an app running on a server which provides facilities or operations such as data storage, printing or communications

LEARNING OBJECTIVES

- Understand why computers are connected in a network
- Understand the different types of networks – local area network (LAN), wide area network (WAN), personal area network (PAN) – and usage models (client–server, peer-to-peer)
- Understand wired and wireless connectivity
- Understand that network data speeds are measured in bits per second (Mbps, Gbps)
- Understand the role and need for network protocols – Ethernet®, Wi-Fi™, TCP/IP, HTTP, HTTPS, FTP, email (POP3, SMTP, IMAP)
- Understand that data can be transmitted in packets using layered protocol stacks and the 4-layer TCP/IP model (application, transport, network, data link)
- Understand the characteristics of network topologies (bus, ring, star, mesh)
- Understand the different mobile communication standards (3G, 4G and subsequent generations)

COMPUTER NETWORKS AND THEIR USE

DID YOU KNOW?

Internet-connected fridges are already available. They can provide details of what food is inside them. They can also send messages to your smartphone or tablet when you need to replace items.

A **network** is a collection of two or more computers that are connected together for the purpose of sharing resources and data. All types of computer (e.g. desktops, laptops, tablets, e-readers, gaming systems, shop tills and even Internet-enabled fridges) can be linked in networks.

Many networks include servers. A server is a powerful computer that provides the network with **services**, such as storing files or sending/receiving emails. A small network might have one server, whereas a large business network could have tens or hundreds of servers.

WHY ARE NETWORKS USED?

A network can support multiple users accessing multiple services at the same time.

Almost all organisations (small or large), including schools, have a computer network that provides multiple services to its users.

The network allows multiple users:

- to read/write personal files on a central server – this provides extra storage space to that on a single computer and also gives a back-up facility
- to access shared files among several users
- to download data or updates to computer programs
- to send data to a shared printer
- to access the Internet
- to communicate with each other – perhaps through email or video, or to play interactive games.

SKILLS

RESEARCH,
CRITICAL THINKING,
COMMUNICATION

ACTIVITY 1

NETWORK BASICS

Carry out the necessary research, then create a leaflet that explains the following to new learners:

- what a network is and what kinds of devices can be connected to it
- what services your school network provides and how to access them
- what services a home network typically provides.

DIFFERENT TYPES OF NETWORK

SUBJECT VOCABULARY

local area network (LAN) a network that covers a relatively small geographical area, often a single site

wireless local area network (WLAN) a local area network in which connected devices use high-frequency radio waves to communicate

wide area network (WAN) a network that covers a large geographical area. It connects together two or more LANs and is usually under shared ownership. The largest wide area network is the Internet

personal area network (PAN) network used for data transmission over short distances by computer devices, such as laptops, mobile phones, tablets, media players, speakers and printers

Bluetooth™ a protocol for the short-range wireless interconnection of mobile phones, computers and other electronic devices

LAN, WAN AND PAN

A **local area network (LAN)** is a network that covers a relatively small **geographical** area. This is often a single site, such as a home, a hospital or a factory. The hardware (e.g. cables, routers, etc.) that connects the computers, servers and other hardware devices are usually owned by the organisation that the network belongs to.

Many people have a **wireless local area network (WLAN)** at home. A wireless router allows all the computer devices in a household to access the Internet and share devices such as printers and external hard drives.

A **wide area network (WAN)** covers a large geographical area, usually across several sites of an organisation. Each site has one or more LANs, and they are all connected together to make a WAN. The WAN allows employees on different sites to communicate and share data.

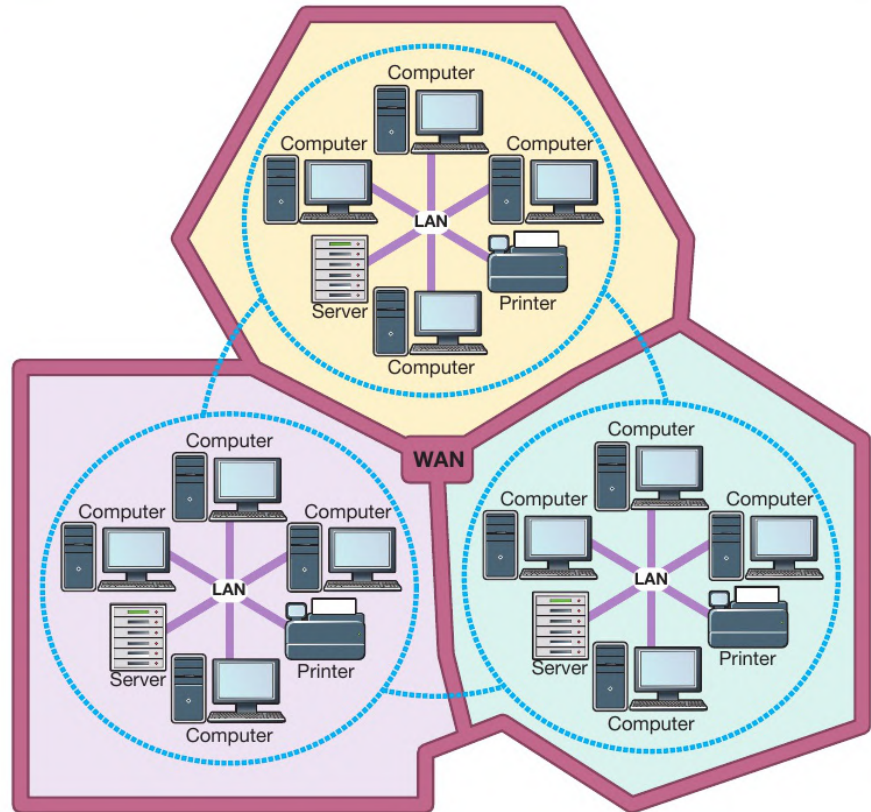
A **personal area network (PAN)** is a network communicating between computer devices, such as laptops, mobile phones, tablets, media players, speakers and printers. They may be devices belonging to one person or to several.

The PAN may just connect the local devices or allow them to connect to higher-type networks such as the Internet. The devices usually communicate wirelessly over distances of up to 10 metres by radio waves using a technology called **Bluetooth™**.

GENERAL VOCABULARY

geographical connected to a particular area or place

► Figure 5.1 LANs may be connected together to form a WAN



GENERAL VOCABULARY

peer someone who is in the same social or hierarchical position as somebody else

SUBJECT VOCABULARY

client-server network a network that has at least one server to provide services to the client computers

CLIENT-SERVER AND PEER-TO-PEER NETWORKS

There are two main models relating to computer networks: client-server and **peer-to-peer**.

CLIENT-SERVER

In a **client-server network** there is at least one server, which is a powerful computer that provides a service or services to the network. Individual computers (such as the ones in a typical school computer room) are referred to as the client computers. The server will authenticate the user (explained later in the unit) and then give the user access to the files that he/she has been given permission to access. The server provides services to the clients as required.

A client accesses data or files from a server using the following process:

- 1 A client will make a connection to the server using its address. The server will know the address of the client because this will be included in the request for a connection.
- 2 Once the connection has been made, the client will make a service request to the server.
- 3 If the request is valid, the server will send the requested data to the client using the address identified in step 1.

Sometimes additional steps, such as authentication, are needed, depending on what the client is trying to access. For example, when Jamala wants to check her web-based email from her laptop, she will use a web browser (the client) to access the website, which is hosted on a server. The website will serve a web page that asks for her username (her email address) and her password.

She will enter her username/password and then submit them to the server. If the details are correct, she will be authenticated, and the server serves back the web page that displays her emails and allows her to manage her email.

SUBJECT VOCABULARY

peer-to-peer network a network that doesn't have any dedicated servers. Each computer in the network can act as client and server

network topology describes how the devices on a network are connected together

GENERAL VOCABULARY

dedicated committed to one particular function

recipient the person receiving something

pirated stolen

PEER-TO-PEER

Unlike in a client-server network, there aren't any **dedicated** servers in a **peer-to-peer network**. Each computer can act as both a client and a server. Each computer in a peer-to-peer network can provide a service, such as share some files or provide access to a printer. Each computer can also request services from any other computer that has been configured to provide that service.

Some instant messaging systems use a peer-to-peer service to allow users to chat in real time. There is no central server that you sign on to and your messages are sent directly to the **recipient**.

DID YOU KNOW?

Some peer-to-peer networks connect using the Internet. These particular peer-to-peer networks are often used for sharing files, especially **pirated** movies, music and computer games. It is difficult to find out who is sharing files when they are distributed using peer-to-peer networks. This is because no records of who downloaded what are normally kept.

There are also legal uses of Internet-based peer-to-peer file sharing. For example, some versions of Linux can only be downloaded using peer-to-peer file sharing. The lack of records also benefits those people worried about privacy.

SKILLS

CRITICAL THINKING,
RESEARCHING,
COMMUNICATION

ACTIVITY 2

CLIENT-SERVER VERSUS PEER-TO-PEER NETWORKS

- 1 Create a table comparing a client-server network with a peer-to-peer network. Use one column for the client-server network and another for the peer-to-peer network.
- 2 Research why Internet-based peer-to-peer networks are a popular way of distributing pirated material such as movies or music. Present your findings as a short report.

NETWORK TOPOLOGIES

There are several different ways that the connections between networked devices can be arranged. The arrangement of these connections is referred to as the **network topology**.

There are four main network topologies – bus, ring, star and mesh.

BUS

A bus network consists of a single cable to which each networked device is connected. Messages are sent along the cable in the form of electronic signals. At each end of the cable is a terminator.

GENERAL VOCABULARY

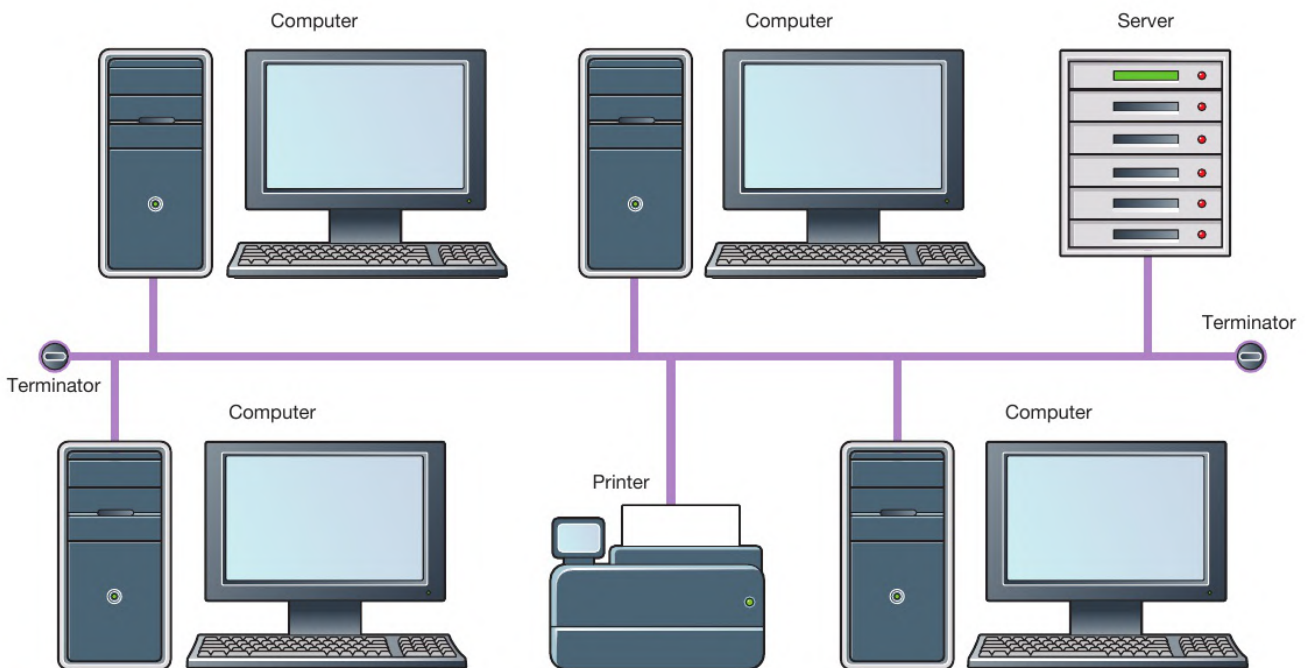
interference where signals collide with each other and cause both to be altered

collision when two or more things crash into each other

unreadable cannot be read

The purpose of a terminator is to absorb signals that have reached the end of the cable. This prevents them from bouncing back and causing **interference**.

As a bus network only has one cable (the bus), only one message can be sent at any one time. A bus network needs a system to deal with multiple devices sending a message at the same time. The most common method is called Carrier Sense Multiple Access with Collision Detection (which is shortened to CSMA/CD). A **collision** occurs when two or more network devices send a message at the same time, making all the messages **unreadable**.



▲ Figure 5.2 Bus network topology

DID YOU KNOW?

The bus topology isn't widely used in business computer networks any more. However, it is used in many modern cars and some aircraft to allow all the different systems within the vehicle to communicate. The most well-known standard is called CAN (Controller Area Network) bus.

In simple terms, a CSMA/CD sending algorithm works like this.

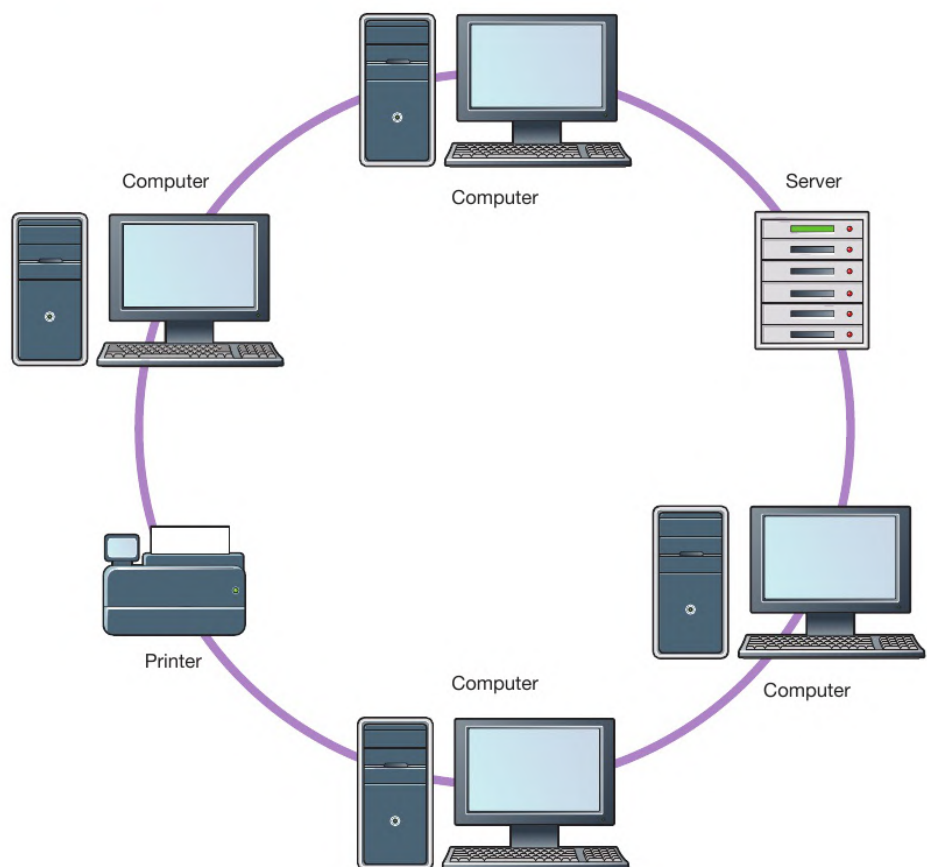
- 1 Check if bus is busy.
- 2 If not busy then send message, else go to step 1.
- 3 Listen to see if message received correctly.
- 4 If message not received correctly then go to step 5, else go back to listening for messages.
- 5 Wait any amount of time, go to step 1 to retry sending message.

Making each device wait any amount of time (e.g. a small number of milliseconds) before checking whether it is safe to send, prevents the different senders from repeatedly sending at the same time.

ADVANTAGES AND DISADVANTAGES OF THE BUS TOPOLOGY

ADVANTAGES	DISADVANTAGES
Relatively cheap to install since only one cable is needed	Whole network will fail if the cable is cut or damaged
Easy to add extra network devices	Can be difficult to identify where a fault is on the cable
	The more devices that are added to a bus network the slower they run. This is due to only one message being able to be sent at once and because more collisions happen
	All data sent is received by all devices on the network; this is a security risk

RING



► Figure 5.3 Ring network topology

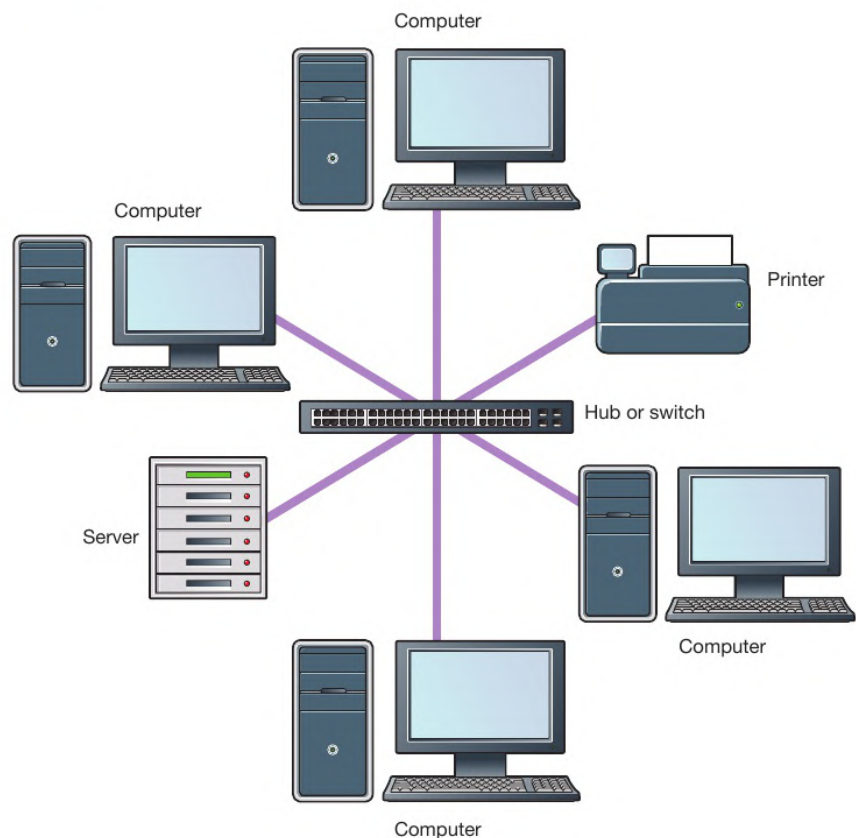
A ring network is a network in which the cable connects one network device to another in a closed loop, or ring. Each network device has what can be thought of as an 'in' and an 'out' connection.

Messages sent on a ring network all travel in the same direction and, unlike a bus network, there are no collisions. Data is passed from one device to the next around the ring until it reaches its destination.

ADVANTAGES AND DISADVANTAGES OF THE RING TOPOLOGY

ADVANTAGES	DISADVANTAGES
Adding extra devices does not affect the performance of the network	Whole network will fail if the cable is cut or damaged or a device on the network fails
Easy to add extra network devices	Because all the devices in the network are connected in a closed loop, adding or removing a device involves shutting down the network temporarily
	Can be difficult to identify where a fault is on the network
	More expensive to install than a bus network as it requires more cable to complete the ring

STAR



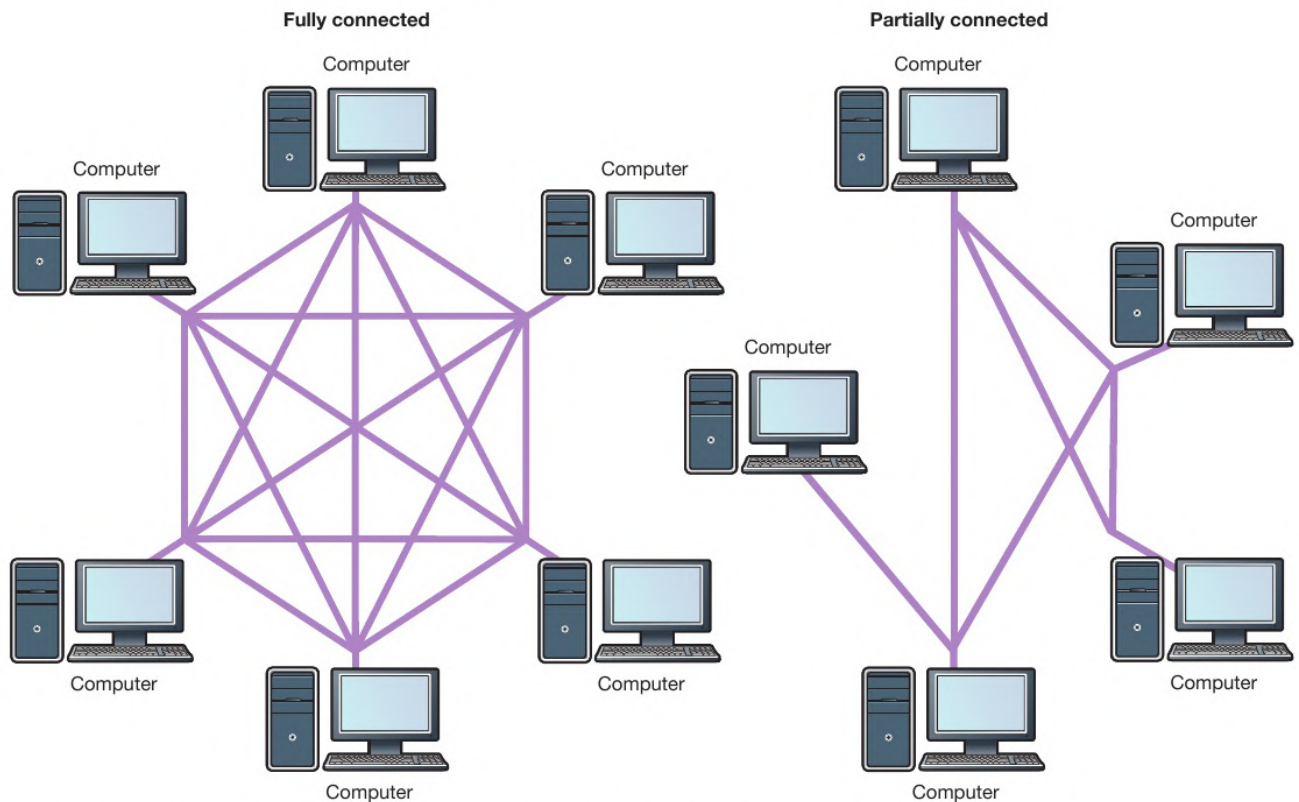
► Figure 5.4 Star network topology

In a star topology, each network device is connected to a central point. This central point will be either a hub or a switch. A central hub receives and directs messages to the correct recipients. The star topology is the most widely used, but it does require a lot of cabling. This is because each device is connected to the central point.

ADVANTAGES AND DISADVANTAGES OF THE STAR TOPOLOGY

ADVANTAGES	DISADVANTAGES
A damaged cable will not stop the whole network from working, just the network device connected to it	If the hub or switch fails, then the whole network will fail
If a switch is used, then the network is efficient as messages are only sent to devices needing them. This also adds to security as not all devices will see a message	Expensive to install due to amount of cable needed and the hub or switch
Easy to locate faults because they will normally only involve one device	
A new device can be added or removed without having to close the network down	

MESH



▲ Figure 5.5 Mesh network topology showing fully connected and partially connected types

GENERAL VOCABULARY

partially not entirely**routed** sent in a particular direction, on a particular path**sensor** a device which detects or measures a physical property**optimum** best possible outcome

SUBJECT VOCABULARY

Internet a worldwide system of interconnected networks that enables information to be exchanged and shared**communication media** the means by which data is transmitted between devices on a network. Coaxial cable, fibre-optic cable and microwaves are all forms of communication media

SKILLS

CRITICAL THINKING,
RESEARCH,
DECISION MAKING

There are two main types of mesh topology – fully connected and **partially** connected. In a fully connected mesh network, every network device is connected to every other network device. In a partially connected mesh network, some network devices may be connected to multiple other devices. However, others might only be connected to one other device. Each device in a mesh network will pass messages on to other devices within the network.

Mesh networks can be wired or wireless. In a wired network, the number of connections needed for a fully connected network soon becomes very expensive and difficult to implement. This is because the number of network devices increases. Wireless mesh networks don't have the same problems.

Mesh networks are very fault tolerant. If a device or connection fails, messages are **routed** around it. In a fully connected mesh network, a large number of failures could happen, and the rest of the network devices would still be able to communicate. The ability of a partially connected mesh network to carry on working correctly will depend on how many connections to each device are available.

The largest mesh network of all is the **Internet**.

Mesh networks are used to communicate between **sensors** in the 'Internet of things', especially if the devices are mobile, because the **optimum** network paths can be used no matter where a sensor is located.

ADVANTAGES AND DISADVANTAGES OF THE MESH TOPOLOGY

ADVANTAGES	DISADVANTAGES
Very fault tolerant, especially in the case of a fully connected mesh network – if one device fails, messages can be re-routed	Difficult and expensive to install wired mesh networks
Very high performance because each network device is likely to be connected to multiple other devices	Can be difficult to manage due to number of connections within the network
In a wireless mesh network each node extends the range of the network	

ACTIVITY 3

NETWORK TOPOLOGIES

- 1 Close this book and draw an annotated diagram for each of the following topologies: ring, star, bus and mesh.
- 2 From memory, list the advantages and disadvantages for each network topology you drew in Question 1.
- 3 Research which network topology in Question 1 would be most suitable for providing local access to the Internet in a country that doesn't have reliable wired communication links. Write a paragraph explaining your decision and how the system could work.

COMMUNICATION MEDIA

There are many different ways that computers can be connected together to form a network. Networks are either wired or wireless according to the transmission or **communication media** used.

DID YOU KNOW?

Many TV remote controls use infra-red light to send signals to control the volume or change the channel.

**WIRED**

Wired connection methods involve a physical connection between the computer and the network. Most wired connections are made of copper wire. They could also be fibre-optic cable, which is made of either glass or plastic. Copper wire carries electrical signals, while fibre-optic cable carries light signals. Fibre-optic cables transmit data far faster and allow signals to travel over greater distances than copper wire.

WIRELESS

Wireless connectivity does not require a physical connection between devices. Most wireless connections transmit and receive radio signals, but other connection methods such as infra-red light can be used over limited distances.

Important methods of wireless communication are the mobile phone network, Bluetooth and Wi-Fi™.

SUBJECT VOCABULARY

protocol a set of rules that govern how communications on a network should be formatted and what data they should include

eavesdrop having unauthorised sight of data being sent from one computer to another over a network (this is covered more fully later in the unit in the section on technical weaknesses)

DID YOU KNOW?

The 'Internet of things' is a term that covers the growing range of physical items that can be connected to the Internet. These physical items (the 'things') can report details about themselves and be controlled over the Internet. So, your central heating could report the temperature of your house and allow you to change the thermostat from any Internet-connected device, including your mobile phone.

ZigBee is an open-source **protocol** for allowing devices to be controlled by using low-power digital radio transmitters/receivers that are embedded within the device.

GENERAL VOCABULARY

susceptible likely to be influenced or harmed

WIRED OR WIRELESS CONNECTIVITY?

There are advantages and disadvantages to wired and wireless connection methods.

ADVANTAGES	DISADVANTAGES
Wired connectivity	
Faster than wireless connectivity	Expensive to install and reconfigure
Not easy to intercept or eavesdrop on data	Requires many cables at a premises
Less susceptible to interference than wireless connectivity	
Wireless connectivity	
No need for a cable to connect devices or to the Internet	Data transmission speeds can be slower than wired connectivity
Allows users to use their own device	Interference from other wireless devices can adversely affect performance
A wider range of devices can communicate with each other/a network because it is not dependent on having the correct cable	Walls and other physical objects can adversely affect performance
	Data needs to be encrypted (see Unit 3) to prevent eavesdropping or interception

NETWORK DATA SPEED

Being able to measure the speed of a network allows you to:

- discover whether an equipment upgrade is necessary
- estimate how long it will take to download a file
- discover whether actual performance lives up to the promises made by the service provider.

The speed that data can be transmitted through a communications medium is measured in bits per second (this can be shortened to bps, b/s or bit/s). Modern high-speed networks can transmit billions of bits per second.

The units listed have similar names to the units used to measure storage (such as the capacity of a hard drive or the amount of RAM a computer has). But they are different, and you should make sure not to confuse them. For example, you might measure storage in kilobytes (see Unit 3, page 137); however, you measure the speed of a network connection in kilobits per second.

CALCULATING THE TIME NEEDED TO TRANSMIT A FILE

WORKED EXAMPLE

If you have a network connection of 12 Mbps and you want to download a 50 MB file, how long would it take?

To find out:

- 1 Convert the file size from megabytes to bits.
 $\text{File size in megabytes} \times 1024 \times 1024 \times 8$
 $50 \text{ MB} \times 1024 \times 1024 \times 8 = 419\,430\,400 \text{ bits}$
- 2 Convert the transmission speed from Mbps to bits per second.
 $\text{Transmission speed in Mbps} \times 1000 \times 1000$
 $12 \times 1000 \times 1000 = 12\,000\,000 \text{ bps}$
- 3 Divide bits by bits per second.
 $419\,430\,400 / 12\,000\,000$

Time = 34.95 seconds

These times are only theoretical as transmission speeds are not constant and other factors affect the speed of data transmission.

SUBJECT VOCABULARY

latency the delay encountered by data travelling through a network

The amount of data that can be carried from one point to another on a network in a given period of time is referred to as its bandwidth. Bandwidth is usually expressed as bits per second (bps). Another important factor is the **latency** of the network connection. Latency refers to any kind of delay that data travelling through a network might encounter. A low latency network connection is one where any delays are small and a high latency network is one where data suffers from long delays.

SKILLS

RESEARCH,
PROBLEM SOLVING

ACTIVITY 4

INTERNET SPEED

- 1 Find out how fast your school, home and mobile phone Internet connections are. Compare these, ensuring you use the correct units (Kbps, Mbps or Gbps). Make a note of your findings.
- 2 Work out how long it would take to download a 300 MB file using your home and school networks.

PROTOCOLS

In relation to computer systems and networks, a protocol is a set of rules that control how communications between devices are formatted and how these communications will be sent/received. Without protocols, different computers and other hardware wouldn't be able to communicate with each other because they would essentially be speaking different languages.

A protocol might contain details of:

- how each computer will be identified (its address)
- what route the data will take to get to its destination (routing information)
- how errors will be found and dealt with (error checking)
- whether each part of a message should be acknowledged as received correctly
- what to do if data isn't received correctly
- how the data is to be formatted
- how the data is to be sequenced (i.e. does it need to be sent in order or can it be put in its correct order later?)
- how the speed of the sender and receiver can be **synchronised**.

SUBJECT VOCABULARY

synchronised occur at the same time or operate at the same rate

DID YOU KNOW?

In the 1980s, manufacturers of home computers each developed their own set of protocols, rather than all using a standard set of protocols. This meant that if you bought one manufacturer's computer, you would have to buy a printer especially designed for that system. If you later bought another manufacturer's computer, you would also have to change your printer and most other hardware.

There are many different protocols for different purposes. The main ones in use relating to networks and the Internet are detailed below.

EMAIL PROTOCOLS

Emails are sent and received using a set of standard protocols. This means that when you send an email it doesn't matter what email provider the recipient uses or what type of computer system they have.

There are three main email protocols in use.

PROTOCOL	DESCRIPTION
SMTP	Simple Mail Transfer Protocol. This protocol is used when sending email through the Internet. It details the format that messages are sent in, what commands email servers should understand and how they should respond to them.
POP3	Post Office Protocol, Version 3. POP3 is the current version of the Post Office Protocol that is used for retrieving email from an email server. Normally email clients using POP3 will connect to the mail server, download any messages and then delete the messages from the server.
IMAP	Internet Message Access Protocol. IMAP allows emails to be accessed using multiple email clients. For example, you might access your email using an email client on your computer, your tablet and your mobile phone. IMAP leaves messages on the server until you delete them. This means that no matter which email client you use, you should see an up-to-date list of your email messages.

NETWORK PROTOCOLS

ETHERNET

Ethernet is a family of protocols that are used in wired LANs. They cover everything from the physical parts of a network, such as type of cable or optical fibre and type of connector to be used, to the logical parts, such as



SUBJECT VOCABULARY

checksum a technique for finding errors. A mathematical formula is applied to the data and the resulting number value is transmitted with the data. The recipient computer applies the same formula to the received data. It then compares the checksum sent with the data to the calculated checksum. If the checksums don't match, the data is likely to have been corrupted. So, the recipient computer requests the data again

how data is sent and checked for errors and the speed that data can be transmitted.

WI-FI

Wi-Fi is a digital communications protocol that sets out how data is transmitted on wireless LANs. Wi-Fi is a trademarked term that is owned by Wi-Fi Alliance.

TCP

The Transmission Control Protocol provides a reliable connection between computers. Here, 'reliable' means that the receiving computer can be certain that it has received all the data it should have (none of it is missing) and the data received is identical to the data sent (the data is correct).

TCP does this by:

- specifying that the receiving computer sends acknowledgements that each section of the data sent has been received
- using **checksums** to ensure that the data received is accurate
- allowing the receiving computer to tell the sending computer to slow down transmission. This means the receiving computer has time to process the received data (this is called flow control)
- ensuring that data sent up to the application layer contains no duplicates and is in the correct order.

TCP is used when you access web pages, send/receive email or upload/download files.

TCP/IP

TCP/IP stands for Transmission Control Protocol/Internet Protocol. TCP/IP is a protocol stack, which means that it is a collection of protocols that work together. It is named after the two most important protocols used in the stack.

LAYER	DESCRIPTION
Application	This is the top layer of the stack. It is the layer which interacts with the user to provide access to services and data that is sent/received over a network. Examples of protocols that work at this layer are HTTP, FTP and email protocols.
Transport	This layer manages end-to-end communication over a network. There are two main protocols that operate at this layer – TCP and UDP. (You only need to know about TCP for iGCSE Computer Science.)
Internet	This layer deals with sending data across multiple networks (possibly the Internet), from the source network to the destination network. This is known as routing and is the role of Internet protocol (IP).
Link	This layer controls the transmission and reception of data to/from a local network.

▲ Table 5.1 The four layers of the TCP/IP protocol stack

SUBJECT VOCABULARY

packet a small quantity of data being sent through a network. The packet is labelled with the sender's address (source), the recipient's address (destination), how many packets are being transmitted and the position of this packet in the complete message

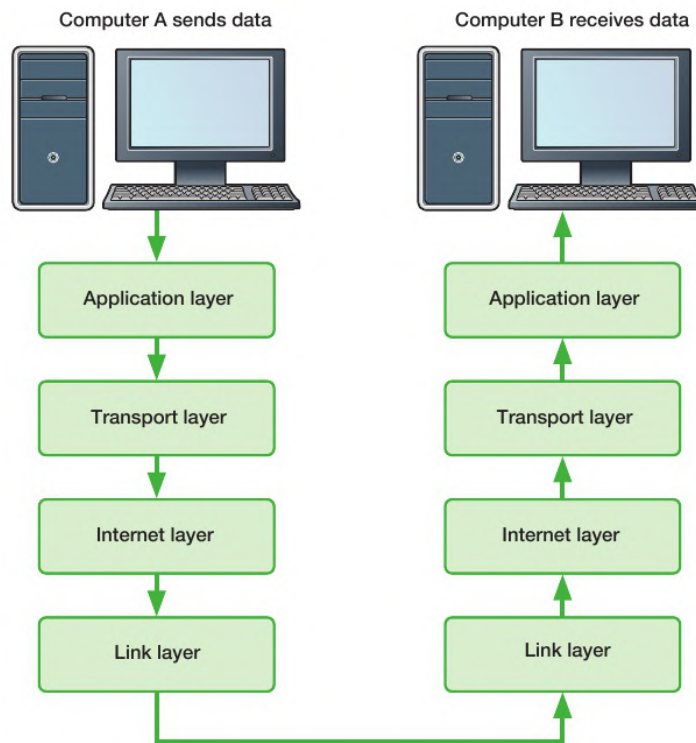
Data sent using TCP/IP is broken up into **packets**. Each packet of data consists of a small section of the data being sent along with a packet header. A packet header contains details of:

- the sending computer
- the recipient computer
- how many packets the data has been split into
- the number of this particular packet.

Once the recipient computer receives all the packets, it will use the information in the header to reconstruct the data into its original format.

A network that moves data in packets is called a packet switched network. (More details of how this works can be found later in the unit.)

► **Figure 5.6** Layers within the TCP/IP suite pass data down the stack when sending and back up the stack when receiving

**DID YOU KNOW?**

You will sometimes see TCP/IP described as a 'protocol suite' rather than a 'protocol stack'. These two terms are often used as if they are the same. However, some sources state that the suite is the definition of the protocols and that stack is the software implementation of the protocols.

EXTEND YOUR KNOWLEDGE

Another protocol that plays a major part in the TCP/IP protocol suite is called UDP. Research this and create a table explaining the differences between TCP and UDP.

SUBJECT VOCABULARY

web server powerful computer system that stores web pages and any multimedia that the pages might contain

HTTP

The HyperText Transfer Protocol (HTTP) is used when sending and receiving data between web browsers and **web servers**. The HTTP protocol covers how data should be formatted, what commands the web server and web browser should understand and how they should react to each command.

For example, if you request a web page that doesn't exist on the web server, then the web server will send your web browser a **404-error** message. Your web browser will display this error and a short description of the error.

HTTPS

The HyperText Transfer Protocol Secure (HTTPS) is the secure version of HTTP. This means that the data sent between your web browser and the web server is encrypted. It should prevent the data being sent/received from being read by a third party. HTTPS also helps the web browser and user know that they are communicating with the intended web server rather than a fake that is trying to steal sensitive data, such as passwords or bank details.

SKILLS

RESEARCH,
CRITICAL THINKING

ACTIVITY 5

ADVANTAGES OF HTTPS

When you use online banking, your data will be transferred using HTTPS. In a group, research why HTTPS is used rather than HTTP.

What advantages does HTTPS give the bank customer and the bank itself? In a group, create a poster summarising this information.

FTP

File Transfer Protocol is used to transfer files over a network that uses the TCP protocol (see above), such as the Internet.

FTP is often used when sending web pages and other associated files that have been created on a web developer's computer to the web server. Once the files are on the web server, other computers connected to the Internet will be able to view them.

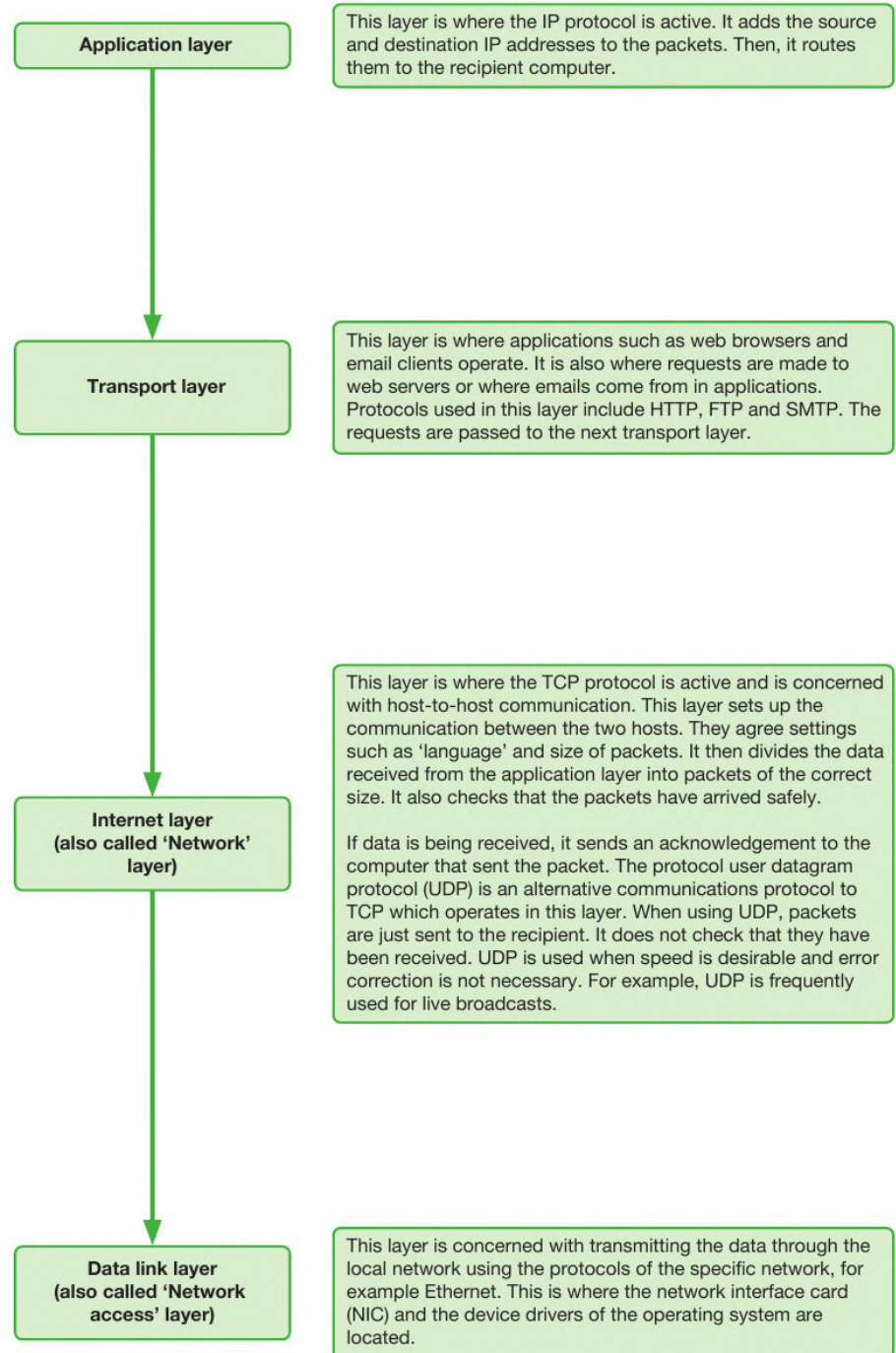
INTERNET PROTOCOLS

At the Internet layer, the Internet Protocol (IP) deals with:

- the addressing system to identify individual computers/servers on the network (usually the Internet)
- splitting data into packets and adding the packet header with details such as the sender/receiver addresses.

Each device linked directly to the Internet has a unique IP address assigned to it to allow data to be routed to it.

All of these protocols are in the TCP/IP suite, arranged in four layers. Data to be transmitted (or to be received) must pass through the layers where packaging data are added to or read from the packets. There are four layers in the 4-layer TCP/IP model.



▲ Figure 5.7 The 4-layer TCP/IP model

HINT

Remembering a list of items where the order is important (such as the TCP/IP layers) can be difficult. Why not try using a mnemonic like 'Another Task Is Done', where the initial letter of each word matches the initial letter of the layer name?

PROTOCOLS OF THE APPLICATION LAYER

Table 5.2 shows some of the protocols of the application layer.

FTP	File Transfer Protocol: the rules that must be followed when files are being transmitted between computers
HTTP	HyperText Transfer Protocol: the rules to be followed by a web server and a web browser when requesting and supplying information. HTTP is used for sending requests from a web client (a browser) to a web server and returning web content from the server back to the client
HTTPS	Secure HTTP: allows for communications between a host and client to be secure. It ensures that all communication between them is encrypted
SMTP	Simple Mail Transfer Protocol: the protocol for sending email messages from client to server and then from server to server until it reaches its destination
POP	Post Office Protocol: used by a client to retrieve emails from a mail server. All of the emails are downloaded when there is a connection between client and server
IMAP	Internet Message Access protocol: unlike POP, the messages do not have to be downloaded. They can be read and stored on the message server. This is better for users with many different devices. This is because they can be read from each other, rather than being downloaded to just one.

▲ Table 5.2 Some of the protocols of the application layer

SKILLS

REASONING,
CRITICAL THINKING

ACTIVITY 6

PROTOCOLS

Create a revision sheet that includes the following:

- 1 a definition of what a protocol is
- 2 a table that includes the following protocols: SMTP, POP3, IMAP, Ethernet, Wi-Fi, HTTP, HTTPS and FTP, along with a brief description of what each one is used for
- 3 a diagram showing the TCP/IP protocol layers – you should annotate this to show what each layer does and which protocols run at each layer.

BENEFITS OF USING NETWORKING LAYERS

Networking technologies are separated into layers, each one containing specific hardware and software protocols. Each layer performs specific tasks and interacts with the **adjacent** layers in the 'network model'.

The benefits of this approach are:

- It makes the overall model easier to understand by dividing it into functional parts.
- Each layer is specialised to perform a particular function.
- The different layers can be combined in different ways.

GENERAL VOCABULARY

adjacent next to each other

- One layer can be developed or changed without affecting the other layers.
- It makes it easier to identify and correct networking errors and problems.
- It provides a universal standard for hardware and software manufacturers to follow. This is so that they will be able to communicate with each other.

MOBILE COMMUNICATIONS



A mobile or cellular network is also a wireless network distributed through cells where each cell includes a fixed location transceiver known as a base station. These cells together provide wireless coverage over a large geographical area. Each cell offers coverage of between 9 and 21 miles.

When a user moves out of range, the signal falls and the base station makes a request to transfer control to another base station that is receiving the strongest signals without notifying the subscriber. This is called handover.

The European Telecommunications Standards Institute developed a standard for mobile communications called the Global System for Mobile Communications (GSM) to describe the protocols to be used. As these develop, new versions are called 'generations', as in 2G, 3G, 4G and 5G.

- 2G was the first to use digital communications. It enabled text messages to be sent and introduced a multimedia messaging service (MMS).
- 3G increased data transmission speeds to 2mbit/s. This gave wireless access to the Internet, so it enabled video calls and downloading and streaming.
- 4G provided much higher speeds and gave rise to the popularity of mobile gaming. A film that would have taken more than five hours to download on 3G could take less than eight minutes! 4G has a much higher capacity, meaning it can support a greater number of users at the same time. It also gives a more immediate response to a user's commands, useful when playing online games.
- 3G and 4G both use IP-based protocols for data but 4G also uses IP, even for voice calls.
- The new generation, 5G, is smarter, faster and more efficient than 4G. It achieves peak speeds of 100 Gbps, 100 times faster than 4G and faster than most home broadband networks. It also has a far lower latency than 4G. 5G will be able to handle current devices and emerging technologies, such as driverless cars and connected home products.

CHECKPOINT

Strengthen

- S1** What is the difference between a LAN, a WAN and a PAN?
- S2** What are the advantages of wireless networks compared with wired networks?
- S3** What are the three major protocols used in sending and receiving email?
- S4** What is a protocol stack?
- S5** Create a diagram of the 4-layer TCP/IP model.
- S6** From memory, create a diagram showing the bus, ring and star topologies.
- S7** Can you name one difference in communications standards between 3G and 4G?

SKILLS CRITICAL THINKING

SKILLS REASONING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS PROBLEM SOLVING

SKILLS PROBLEM SOLVING

SKILLS CRITICAL THINKING

SKILLS REASONING**SKILLS** CRITICAL THINKING**SKILLS** REASONING**SKILLS** REASONING**SKILLS** CRITICAL THINKING**Challenge****C1** Why might a chain of supermarkets use a WAN to connect all its stores?**C2** What are the advantages of a peer-to-peer network in terms of privacy when sharing files?**C3** Most networks in use in schools and offices today are based on the star topology. Why do you think this topology is used?**C4** IMAP is now used much more than POP3. Can you explain the link between this and the popularity of smartphones?**C5** What are the functions of the protocols in the application layer?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try drawing a concept map showing how the information in this section links together. Your concept map can have brief definitions on it and some people use colours/pictures to help them memorise the contents.

SUMMARY

- A network is a number of computers and other devices connected together to allow them to communicate and share data.
- A PAN is the interconnection of digital devices within the vicinity of an individual user.
- A LAN covers a relatively small area. A WAN connects LANs together and spans a large geographical area.
- Client-server and peer-to-peer networks are two different ways of setting up a network. In a client-server network, a central computer fulfils service requests from client computers, whereas in a peer-to-peer network each computer can act as a client and a server.
- Network devices can communicate using wired or wireless connections.
- A protocol is a set of rules governing communications on a network. They specify the format for communications and how they must be sent/received.
- There are many different protocols, each serving a different purpose.
- Data is transmitted in packets using layered protocol stacks.
- A network topology is how the connections between devices in a network are configured.
- The developing standards provided by the European Telecommunications Standards Institute are referred to as 2G, 3G, 4G and 5G.

22 NETWORK SECURITY

It is essential that all data stored on networks is kept confidential and secure. In this section you will learn about the threats to network security from criminals, either directly or by creating malicious software, and how networks can be kept secure.

LEARNING OBJECTIVES

- Understand the importance of network security and be able to use appropriate validation and authentication techniques (access control, physical security and firewalls)
- Understand security issues associated with the 'cloud' and other contemporary storage
- Understand different forms of cyber attack (based on technical weaknesses and behaviour), including social engineering (phishing, shoulder surfing, pharming), unpatched software, USB devices, digital devices and eavesdropping
- Understand methods of identifying vulnerabilities, including penetration testing, ethical hacking, commercial analysis tools and review of network and user policies
- Understand how to protect software systems from cyber attacks, including considerations at the software (application) design stage, audit trails, securing operating systems, secure coding, code reviews to remove code vulnerabilities in programming languages and bad programming practices, modular testing and effective network security provision

NETWORK SECURITY AND ITS IMPORTANCE

SUBJECT VOCABULARY

network security activities designed to protect a network and its data from threats such as viruses, hacker attacks, denial of service attacks, data interception and theft, and equipment failure

GENERAL VOCABULARY

confidentiality a situation where you would expect something to be kept secret

Network security covers a wide range of activities that protect data from threats to its **confidentiality**, correctness (integrity) and availability.

CONFIDENTIALITY

An organisation's computer system often holds data about its people (employees and customers), products or financial performance. It could not run efficiently, or sometimes at all, without a network and this data. It is important (and in the case of personal data, the law) that the network is not hacked so data is not intercepted or stolen by criminals or competitors who might use the data for illegal purposes or business advantage.

Some ways to protect data confidentiality include:

- ensuring only authorised users can access the parts of a network and its resources that they have a reason to require, such as its storage (data), printers or Internet connection. For example, the managing director of a company is able to access the financial records, while a checkout employee is not (see also the later section 'Other ways to secure a network')
- stopping misuse – even users who have been given permission to access a network might deliberately or accidentally access data they have not been given permission to read

- encrypting data – if a criminal or other unauthorised person gains access to data and it is encrypted, they won't be able to read it without the encryption key.

CORRECTNESS

Data is useless unless it is correct. Use of the network to communicate and store data must not change the data or allow the data to be changed without authorisation. Imagine the seriousness of an error on your health records or in a manufacturing control system.

AVAILABILITY

A network is useless if data cannot be accessed when it is needed. Complex systems are likely to fail at some time (e.g. a piece of hardware might stop working and have to be replaced, or program code might become corrupted), but it is also important that protection is put in place to prevent failures caused by criminals. Virus or **denial of service (DoS)** attacks can, for example:

- slow down network performance or stop it working altogether
- delete data
- allow data to be stolen or eavesdropped on
- alter data or program code.

SUBJECT VOCABULARY

denial of service (DoS) an attack on a network that attempts to prevent legitimate users from accessing its services

REASONS WHY SECURITY IS IMPORTANT

Here are some more specific examples of the importance of network security. The data stored on the network could be:

GENERAL VOCABULARY

bankrupt declared in law as unable to pay debts

confidential secret

sue to take legal action against a person or organisation

HINT

Articles relating to network security can often be found on news websites under the technology section. Reading these articles will broaden your knowledge about the latest security threats. You can then look for further details on specialist information technology websites. You will often find details of what happened and how to protect against that kind of threat.

- **required for the running of the organisation** – The business might have details of its customers, stock and outstanding orders saved on the network. If the data were lost and the business failed to fulfil any remaining orders, this could mean losing the trust of customers who then go elsewhere. This could lead to the business going **bankrupt**. This has happened to a number of businesses, some of them very large. Even a school would struggle to run effectively without its network and the data stored on it. Data, such as contact details, registers, timetables, medical conditions, handouts for learners, presentations to be shown by tutors, are stored on the network. They could become inaccessible if the network fails for some reason.
- **private and confidential** – There are many types of private and **confidential** data that people or businesses wouldn't want to make public. For example, a hospital stores patient details on its network, including medical conditions and what drugs have been prescribed. There is a legal requirement to keep personal data secure. If the network isn't secure and the confidential data is leaked, then the hospital could be **sued**. Many potential patients would be put off using the hospital due to a lack of trust in the ability of the hospital to maintain patient confidentiality. This could result in the hospital being closed down.
- **financially valuable** – Many types of data stored on business networks might be financially valuable. For example, imagine a business was planning a huge sale to increase its revenue and attract new customers. If a competitor obtained the details of the planned price reductions before the sale started, it could launch its own sale beforehand. It would undercut the business. This tactic would reduce the chance of the sale increasing revenue or attracting new customers.

AUTHENTICATION AND VALIDATION

Authentication is the process of checking the identity of a user of a computer system or network. This is often done by validating a username and password against details stored on a central server. However, other systems of authentication exist, such as using a PIN (personal identification number) or

fingerprint recognition. If the details you provide match those on record, then you are allowed access to the network and its resources.

SUBJECT VOCABULARY

two-factor authentication a security check where users have to type in the code from a portable hardware device called a 'secure token' or from an SMS message sent to their mobile phone

access control this decides which users have access to which data, and what they are allowed to do with it

SKILLS

REASONING,
PROBLEM SOLVING

EXTEND YOUR KNOWLEDGE

Passwords have many weaknesses. For example, people write them down or use guessable passwords (like a pet's name). Programs exist that repetitively try guessing passwords. To improve the security that passwords provide, an additional check, such as having to type in the code from a portable hardware device called a 'secure token' or from an SMS message sent to your mobile phone, is being introduced by some businesses. This is called **two-factor authentication**. How does this improve security? Try researching some different two-factor authentication systems and think about which ones are likely to be the most secure.

ACTIVITY 7

RESTRICTING ACCESS

Think about how you would write a program so that only approved users who enter their password correctly can get beyond the login screen.

- 1 Develop the pseudocode for the username and password checking part of the program. Then, write your program.
- 2 How secure do you think your implementation is? Could it be improved?

OTHER WAYS TO SECURE A NETWORK

There are many different ways to help secure a network, authentication alone is not enough. Three others you need to know about are: access control, firewalls and physical security.

GENERAL VOCABULARY

modify to change something in order to make it better

DID YOU KNOW?

A system administrator (often shortened to sysadmin) is a person whose job is to manage and maintain the network system. This includes ensuring the network is working correctly, setting up users, dealing with security issues, installing software and many other tasks.

ACCESS CONTROL

Once a user has been authenticated, they gain access to the network; however, you wouldn't want every user on a network to have access to every file. **Access control** is the method that controls whether a particular user will gain access to a particular file. Access control also decides if that user gets:

- read-only access – in this case, the user can open the file and read its contents, but not modify the contents or delete the file
- read and write access (modify access) – in this case, the user can read the file, alter the contents and then save the changes.

In some network systems you might need further permission if you want to delete a file – this is sometimes referred to as full control of the file.

Controlling the level of access a user has to a file is useful for many reasons. For example, you might want employees of a shop to be able to open a price list, but not accidentally or deliberately **modify** any prices.

Access controls are set up by an organisation's system administrators according to the management's requirements.

FIREWALL

A firewall is a network security system that monitors and controls data that is moving from one network to another. Often one of these networks is the Internet, in which case the firewall sits between the Internet and the local internal network.

A firewall inspects incoming and outgoing data and uses a set of rules (often known as a firewall policy). These rules decide whether to allow the data to move from one network to another or not. The rules in a firewall are designed to secure the internal network from a range of potential threats. The organisation that owns the firewall can customise the rules so that the firewall is suitable for their particular circumstances. Some examples of what the rules can do:

- stop certain protocols from being used (for example FTP) to prevent the organisation's data from being potentially copied to an external server
- block data coming from or going to certain network addresses. The blocked addresses could be of particular computers, servers or websites, or even a range of network addresses that belong to particular countries or organisations
- stop attempts at **hacking** the internal network's servers by disallowing data that matches the pattern an attacker would use.

SUBJECT VOCABULARY

hacking the act of gaining unauthorised access to a computer system and the data it contains

throughput allow more data to pass through them

physical security controlling access to critical parts of a network using physical methods (such as locked doors) rather than software

malware short for 'malicious software'. It is used as a generic term for any kind of software that is designed to disrupt the use of a computer system

Firewalls can be software or hardware based. Individual computers (such as your home computer or laptop) are likely to have a software firewall installed with some default rules to protect your computer from common threats. A business with a LAN and Internet connection is likely to have a hardware-based firewall. This is because these have much more flexibility in terms of the rules that can be applied and allow faster **throughput** of data.

PHYSICAL SECURITY

Physical security ensures that critical parts of the network can only be physically accessed by authorised people, such as the network technicians or a systems administrator. It also includes protecting against theft of equipment. It might involve installing a burglar alarm, security tagging and physically locking down equipment, etc.

The starting point for physically securing a network is having the servers located in a locked room. This is normally referred to as the server room. Electronic lock systems have many advantages over traditional lock-and-key systems. Entry/exit times can be recorded, and individual cards/fobs can be deactivated if lost or if an employee leaves the company.

Physical security is very important because anyone with physical access to the server can more easily **bypass** the security provided by the authentication system or access control system. Having done so they could copy, modify or delete any data on the network, or install **malware** that would allow them to gain remote access whenever they wanted.

GENERAL VOCABULARY

bypass go around, rather than through

SKILLS

CRITICAL THINKING,
REASONING



ACTIVITY 8

SECURING A NETWORK

A business has a network comprising 50 computers, several printers and an Internet connection. The business employs two technicians to manage the network; has several departments, including sales, accounts and human resources; and is run by a managing director, with a manager in charge of each department and around 60 other employees. In pairs:

- 1 Create a poster explaining to the employees what authentication is and what makes a secure password.
- 2 Where should the servers in this business be located? Justify your answer and explain who should be able to get physical access to the servers.

CLOUD STORAGE AND SECURITY

The concept of the cloud is discussed in Unit 4, on page 177. This section covers the security issues associated with cloud storage.

Many of the advantages of cloud storage relate to securing the availability of data.

- The cloud storage provider is responsible for the hardware your data is stored on. They will need to ensure that the data is accessible and that they maintain the hardware and software needed to make the data available.
- The amount of storage available to an organisation can easily be changed as and when required – the cloud storage provider can normally make extra storage available in minutes. Doing this physically would mean ordering more hard drives and employing technicians to install them and set them up. This would take a lot longer than a few minutes.
- Having data stored off-site (not on the organisation's premises) means that it is protected from loss due to fire, theft of computers/servers, electrical failure, and so on. As the data is still available the business may be carried on from elsewhere. This could be the difference between staying in business or going bankrupt.
- Many cloud storage systems also manage the back-up of your data. They may take hourly or daily snapshots of your data and be able to restore from these in the case of hardware failure or accidental deletion of files.

However, cloud storage also has a number of security disadvantages.

GENERAL VOCABULARY

catastrophic dangerous, harmful, disastrous

- You are relying on a third-party storage provider to keep your organisation running. What happens if the cloud storage provider goes bankrupt, has a **catastrophic** failure or has its servers stolen? In any of these cases, it is likely you would be unable to access your data, at least temporarily, but possibly permanently.
- Data stored anywhere accessible via the Internet carries the risk of other people gaining access to it. This might happen through a deliberate attack by hackers, or as a result of an accidental error in the way it is set up that results in the data becoming available to the public. Also, the Internet isn't a particularly secure network. Data may be accessed as it is being sent to/from the cloud storage provider. The loss of data, particularly of a confidential nature, could be devastating for an organisation.
- Users of cloud storage have to assume that the people providing the service are trustworthy and that their data is being held safely and securely.
- The cloud storage provider might have its servers located in a different country (or countries) to your organisation. Some types of data can only legally be stored in countries that have similar data protection laws to our own.
- Access to cloud storage is dependent on having a reliable, high-speed Internet connection available. How will a business operate if the Internet becomes unavailable for some reason?

The decision to use cloud storage involves carefully weighing up the advantages against the disadvantages. Many of the disadvantages present risks to the operations of an organisation, but the impact of some of these risks can be reduced. For example:

- Many cloud storage providers will store your data at multiple locations so that fire, flood, hardware or electrical failure should have no major impact on accessing your data.

- You can encrypt your data so that it is much more difficult to read while it is either being transferred to/from the cloud storage provider or stored on the provider's servers.

SKILLS CRITICAL THINKING

ACTIVITY 9

ADVANTAGES AND DISADVANTAGES OF CLOUD STORAGE

In pairs, discuss the advantages and disadvantages of moving your school's files/data to cloud storage. Focus on the security and privacy issue that the school will need to address.

CONTEMPORARY STORAGE AND SECURITY

The types of storage systems available are constantly evolving. One recently developed system is network-attached storage (NAS).

NAS is a hardware device that is connected to a network to provide file storage for any device connected to that network. A typical NAS device designed for home use could consist of a single hard drive and associated network hardware, while an organisation's NAS device might consist of many hard drives and the associated network hardware.

NAS devices often include a wide range of additional features, such as allowing access over the Internet, and specialist apps to allow smartphones and tablets to access the files stored on the NAS easily.

Often NAS devices are designed for ease of use rather than being secure. Once a NAS is connected to the Internet it becomes possible for it to be hacked remotely. Home users often make mistakes such as:

- not changing a device's default password. This gives access to the stored files to anyone who looks up the default password on the Internet
- not updating the software running on the NAS. Manufacturers tend to release updated software to fix security weaknesses as they are found. If the software isn't kept up to date, these weaknesses might be used to gain access to the stored data. The security weaknesses are often published on the Internet for criminals to read.

There are ways to minimise the security risk to data when using a NAS or similar device. Firstly, decide if the data does indeed have to be accessible over the Internet and if NAS is the correct storage device for the data. If the data needs to be available over the Internet, then ensure you use a complex password. Remove any default passwords and ensure that all software patches (updates) are applied.

Another widely used storage device is the **USB** flash drive. USB flash drives are easy to transport, relatively cheap for the amount of storage available and very convenient to use.

The disadvantage is that people can carry large amounts of possibly sensitive information around with them on a small drive that is easily lost. Anyone can find a lost flash drive and access the information. To overcome this, encrypted USB flash drives are available. These are unreadable until a password or number code is entered on to a very small keypad on the body of the drive itself. (See Unit 3, page 144 for more on encryption.)

USB storage devices also present an ideal opportunity for an employee to steal data, perhaps to sell to somebody else.



SUBJECT VOCABULARY

USB Universal Serial Bus sockets found on most modern computer systems

SKILLS

CRITICAL THINKING,
RESEARCH,
COMMUNICATION

ACTIVITY 10

CLOUD STORAGE FOR INDIVIDUALS

Many providers offer cloud storage aimed at individuals.

- 1 Create a table to compare what three of these providers offer. Headings for the table should include: Amount of free storage space, Cost for more storage, Security measures.
- 2 Research and write a brief report that explains what commitments these providers make to keep your data secure. Relate this to the security measures you found out about in step 1 of this activity.

CYBER ATTACKS

A **cyber attack** is any kind of electronic attack on a computer system, server, network or other IT device. These attacks might be designed to:

- gain access to data contained within the system
- delete or modify information
- make the system unavailable for use
- physically damage a device connected to the network (usually by overriding safety limits).

TYPES OF CYBER ATTACK

There are many different forms of cyber attack and these are constantly changing as technology evolves. Most cyber attacks can be classed as exploiting either human behaviour or technical weaknesses.

SOCIAL ENGINEERING

Attacks that rely on exploiting human behaviour are often referred to as **social engineering**. This is because the attacker will 'engineer' a situation where the target person (or group of people) gives away confidential information or gives access to the confidential information.

Three common forms of social engineering are **phishing**, **shoulder surfing** and **pharming**.

PHISHING

A phishing attack is an attempt to get sensitive, confidential information from the user of a computer system or service. Often the phishing attack is targeting usernames and passwords or financial details such as bank account or credit card details.

There are various forms of phishing. The most common types are through email or fake websites that look legitimate, but phishing can also happen via phone calls or instant messaging.

A typical phishing attack might start with an email that asks the user to update details at a bank, online payment system, online auction website or social network. Sometimes the reason given for this request is that there has been a fraud attempt on the user's account.

Once the user clicks on a link within the email, a website that looks and acts like the real website will open. However, when the user enters his or her login details, they are passed to the attacker, who will use them for financial gain.

Phishing takes advantage of people trusting the contents of emails to be genuine. Many people don't bother to look at the web address they have been taken to or think about whether their bank is really going to email them to request a security update.

SUBJECT VOCABULARY

cyber attack any kind of malicious attack on a network-connected device

social engineering any kind of attack on a computer system or network that takes advantage of how people behave and respond to certain situations

phishing attempting to get confidential information by sending a user a message that appears to come from a trustworthy person or organisation

shoulder surfing gaining access to information by watching someone enter it into a computer system

pharming directing Internet users to a fake website that looks like a real one, in order to obtain personal information such as passwords, account numbers, etc.

SHOULDER SURFING

Shoulder surfing means gaining access to confidential information by directly observing a user, possibly literally looking over their shoulder, as they complete a task. Often shoulder surfing is used to get a person's username/password or PIN.

Often shoulder surfing happens in busy places where the target is concentrating on completing a task quickly. A classic example is at a cash machine on a street, where the shoulder surfer stands near the cash machine and sees the user enter his or her PIN on the keypad. Once the PIN has been seen, the card is stolen. The card and PIN can then be used to withdraw cash.

PHARMING

To make it easier for us to use the Internet, domain names are used to represent IP addresses. These are translated back into the IP address by the domain name service (DNS) when we enter the name into a browser. If we have visited the site before, then this IP address will be stored on our computer (in the DNS cache). The browser will connect without using the DNS. We might receive malware in an email that can change the IP address of the domain name to a bogus one. This means that you will visit that site instead. Malware can also infect the DNS servers themselves, so that everybody is directed to the bogus site!

To prevent pharming, users should:

- check that the http address of the site is the one you intended to visit
- check that there is a secure connection (https) if you have to enter sensitive information
- check the site's security certificate
- install the latest security updates
- install antivirus software.

SKILLSRESEARCH,
CRITICAL THINKING**ACTIVITY 11****CYBER ATTACKS**

- 1 Research recent cyber attacks. What kinds of organisations were targeted? How successful were the attacks? What information was taken?
- 2 Some organisations have been known to deny that a cyber attack successfully happened or downplay the effects of such an attack. Why might they do this? What are the risks to a business of lying about cyber attacks?

TECHNICAL WEAKNESSES

Other forms of cyber attack rely on technical weaknesses (vulnerabilities) in the system being attacked. Some common examples are described below.

UNPATCHED SOFTWARE

Software is very complicated and usually security issues are only found as it is used in the real world. The maker of the software will normally provide updates (referred to as patches) to fix security issues as they are found.

Often these security issues are discussed on the Internet. Some people will use this knowledge to attack **unpatched software** to gain unauthorised access to information.

The patches to fix the security issues often have to be manually installed by a technician. Sometimes these patches get forgotten about or the organisation doesn't have any technicians to install the patch. So, the software remains **vulnerable**.

SUBJECT VOCABULARY

unpatched software software that hasn't had the latest security updates applied to it, making it vulnerable to attack

GENERAL VOCABULARY

vulnerable easy to hurt

USB DEVICES

You read earlier about the weaknesses associated with USB flash drives, but any USB device can potentially be a security threat because it might contain malware that could be transferred to your system or copy data to the attacker via the Internet.

EAVESDROPPING

One form of attack mentioned earlier is an eavesdropping attack. In computer security terminology, eavesdropping means intercepting data being sent to/from another computer system. In a similar way that a person can eavesdrop on a conversation without the speakers knowing about it, eavesdropping on a network is simply reading data without actually copying or stealing it. The owner of the data might not know that the data has been read until it is used by the criminal.

Security weaknesses such as unpatched software or a USB device might allow malware to be installed on the network that allows an eavesdropping attack to be carried out, but specialised hardware might also be used.

PROTECTING AGAINST SECURITY WEAKNESSES

There are many ways to strengthen computer systems and networks from attacks, starting from when software is designed and written, to deciding on what kind of operating system should be used and how network security provision will be implemented.

DESIGN AND IMPLEMENTATION OF SOFTWARE

Security must be considered at the system and software or application design stage if a piece of software is to be as safe as possible against cyber attack. It should be a key concern when the system is being set up.

The software and system designers will need to consider the following, for example:

- What kind of authentication is needed?
- Do different users need different levels of access to the information stored within the system?
- Should warnings be issued before allowing users to carry out potentially risky operations such as copying or deleting large amounts of information?
- Does stored data need to be encrypted?
- What threats will the software face? Will the software be run on computers that are connected to the Internet? How do they guard against phishing, malware and other hacking attempts?

Once the designers have considered security, they need to look at the implementation (writing the code) phase.

One major issue at this point is guarding against bad programming practice. Usually a software project will have many different programmers working on the code. Some programmers write poor quality code and don't consider how safe or secure their code is. Sometimes time pressure might make even a good programmer take shortcuts to meet deadlines.

Another major issue is that some methods used in programming might lead to **code vulnerabilities**. A code vulnerability is when the code does the task intended but has been written in such a way that it creates a potential security issue.

More details about the kinds of issues and their impact that need to be considered when creating software can be found in Unit 6.

SUBJECT VOCABULARY

code vulnerability a computer program (the code) that has been written in such a way that it creates a security issue that may be taken advantage of to gain access to the computer system or data within it

GENERAL VOCABULARY

labour intensive requires a lot of time and effort

chronological the order in which they occurred

SUBJECT VOCABULARY

modular testing testing each block of code as it is completed to ensure the code works as expected

audit trail a record of activities that have taken place on a computer system. This record is generated automatically and will record what has happened and who or what made the change

Issues like these can be minimised by having regular code reviews. There are two main types of code review.

- Review by another programmer, usually someone who is more senior and has more experience of writing secure code. The reviewer will look at the code produced. They will check to see if any bad programming practices have been used or if any code vulnerabilities are present. This is fairly **labour intensive** and therefore expensive.
- An automated review. Here, a specialist piece of software is used to examine the code. The software will highlight potential issues such as common vulnerabilities in the programming language or obvious bad programming practice. This software can't find every issue and is also fairly expensive.

Sometimes a combination of both types will be used. If issues are found, then the original programmer will be asked to improve the code and another code review will be scheduled.

Modular testing is important to security because if small problems remain, they might be used by hackers to gain access to the system or the data it contains. For example, failing to validate input correctly could lead to a hacker being able to crash the program. They could then gain access to the whole system, which could allow valuable data to be obtained.

OTHER SECURITY MEASURES TO PROTECT FROM CYBER ATTACKS

Here are some other methods to reduce the chance of cyber attacks succeeding.

- Use an **audit trail**. An audit trail is a record of activities that have taken place on a computer system, and which cannot be changed. The audit trail is automatically generated and likely to be in **chronological** order. Ordinary users of a system shouldn't be able to read an audit trail. The amount of detail in an audit trail varies but is likely to contain date and time of change, what change happened, who or what (as in a user or other software that is part of the system) made the change. Audit trails allow technicians to figure out what happened during a cyber attack; if the attack was successful; what sections of the system were accessed; and if data was copied or modified.
- Use secure operating systems. Different operating systems are designed for different purposes. Some are designed with security in mind and these are likely to be much harder to attack successfully. This is due to the way they have been written and the features they offer. When implementing a computer system that will contain sensitive or confidential data, the choice of operating system must be considered carefully.
- Provide effective network security. Keeping a network secure requires effective management, monitoring and training of its users. Well-educated technical staff, with up-to-date training, are needed to keep software patched correctly, implement policies that reduce the chance of an attack being successful, train users in best practice in terms of security and monitor the systems to ensure there is no unauthorised access.

SKILLS

RESEARCH,
CRITICAL THINKING

ACTIVITY 12

UPDATING SOFTWARE

A patch has become available for a major web browser. Research the risks that the user faces by not updating his/her software and add details of the potential risks to your revision notes.

IDENTIFYING VULNERABILITIES

Ethical hacking is the branch of computer science that relates to cybersecurity and preventing cyber attacks from being successful. Ethical hacking is essentially 'good' hacking – it is looking for weaknesses in software and systems so that they can be improved – while hacking is seen as 'bad' if someone is trying to gain access to a system to steal data or cause damage.

There are many ways that vulnerabilities can be identified and some of these are explained below. Once a vulnerability has been found, steps to remove or reduce its impact can be taken.

PENETRATION TESTING

Penetration testing (often shortened to 'pen testing') is where the IT systems of an organisation are deliberately attacked to find any weaknesses. These attacks are authorised by the organisation and are therefore legal. The attacks might be run by employees of the organisation or by a business that has been contracted to run the tests.

The pen tester is likely to try to gain access to all the systems that the organisation has and the attacks will often include looking for technical weaknesses and trying social engineering methods. Once the pen testing has been completed, a report is usually presented to a senior manager within the organisation explaining what issues were found and the likely impact of them.

COMMERCIAL ANALYSIS TOOLS

It is possible to use software tools to scan a system for vulnerabilities. These commercial analysis tools or vulnerability scanners can be either purchased or hired. The tools look for common issues and alert the user to them.

These tools can only identify already known vulnerabilities and must be kept up to date to be effective. They can be used to scan the network from within (internally) or from outside (externally). Both of these scans can be valuable. An internal scan can show up issues that could be exploited by a rogue employee or a hacker who managed to physically get into the building or gain access to the network in some other way. The external scan might show vulnerabilities that a hacker could exploit from outside the company network.

One negative aspect is that these analysis tools aren't really restricted in terms of who can purchase them, so they could be used by a potential hacker to find security vulnerabilities.

REVIEWS OF NETWORK AND USER POLICIES

All networks should have written policies that document:

- who is authorised to carry out various activities on the network
- how and when patches to software should be applied
- access controls
- password requirements, including how complex passwords should be and how often they should be changed
- how security is set up and maintained on the network
- what data audit trails should collect and how long they should be kept for
- anything else relevant to the security and maintenance of the network.

Before an employee is given access to the network, they should be given a copy of the network user policy to read and sign to show they accept the organisation's rules on the use of the network.

This user policy is likely to contain details of:

- what use of the network is allowed or not allowed
- what will happen to the user if they do something unacceptable (depending on the severity of what they do wrong, this could include losing their job)
- how to report faults, problems and security issues
- security information, such as good practice when choosing and using passwords.

Over time the hardware, software and usage of a network are likely to change. Any policies relating to the network should be regularly reviewed. This is to ensure that they are up to date and suitable to help maintain network security.

SKILLS RESEARCH,
CRITICAL THINKING

ACTIVITY 13

UPDATING SOFTWARE

In a group, find your school's Network User Policy (you might need to ask your tutor for a copy).

- 1 Review the policy. Has it been updated recently? Is it easy to understand? Is it detailed enough?
- 2 Using your school's policy and any others you can find on the Internet, try to redraft the policy to make it easy to understand for non-specialist users of IT.

CHECKPOINT

Strengthen

- S1** What is the most common authentication method in use?
- S2** What is access control?
- S3** What is social engineering?

Challenge

- C1** In your own words, summarise the different types of cyber attack.
- C2** The number of cyber attacks happening is increasing each year. Why do you think this is the case?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try writing a list of questions then work through this section, adding relevant points as you find them.

SUMMARY

- Authentication is the process of checking the identity of someone trying to use a computer system.
- Cloud storage is storing data using a third party, usually in a system connected to the Internet.
- A cyber attack is any kind of electronic attack on a computer system, server, network or other IT device.
- There are many different types of cyber attack; most exploit technical weaknesses or human behaviour.
- Many different measures need to be taken to prevent cyber attacks from being successful. These start at the design stage of the system and include keeping systems up to date, training staff and proactively testing the security of the network regularly.

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING

SKILLS CRITICAL THINKING,
DECISION MAKING

23 THE INTERNET AND THE WORLD WIDE WEB

The Internet is a wide area network spanning the globe, incorporating networks in governments, educational establishments and businesses across the world. One of the services provided by the Internet is the World Wide Web (WWW), so widely used that people often refer to it as the Internet. The WWW has changed the ways in which people work, shop and communicate.

LEARNING OBJECTIVES

- Understand what is meant by the Internet and how the Internet is structured – IP addressing, domain name service (DNS)
- Understand what is meant by the World Wide Web (WWW) and components of the WWW – web server URLs, ISP, HTTP, HTTPS, HTML
- Understand the need for IP addressing standards and the formats of IPv4 and IPv6
- Understand the role of components used to access the Internet – modem, router, switch, wireless access point (WAP) – and how these are combined

THE INTERNET

The word 'Internet' is a shortened form of the words 'inter' and 'network', which together means 'interconnected networks'. It's a good idea to think of the Internet as a network of networks (i.e. the biggest WAN of all). Your school, many organisations and many people's homes all have a network that is a part of the Internet.

The reach of the Internet is global – all countries have multiple cables, fibre-optic links and satellites that connect them to other countries. These links, plus others within each country, can be thought of as the Internet backbone – the connections that link all the networks together.

HOW THE INTERNET WORKS

DOMAIN NAMES

Domain names are used to identify one or more IP addresses. They are more convenient to use and easier to remember than the four octets of binary numbers. For example, the IP address of bbc.co.uk is 212.58.244.27.

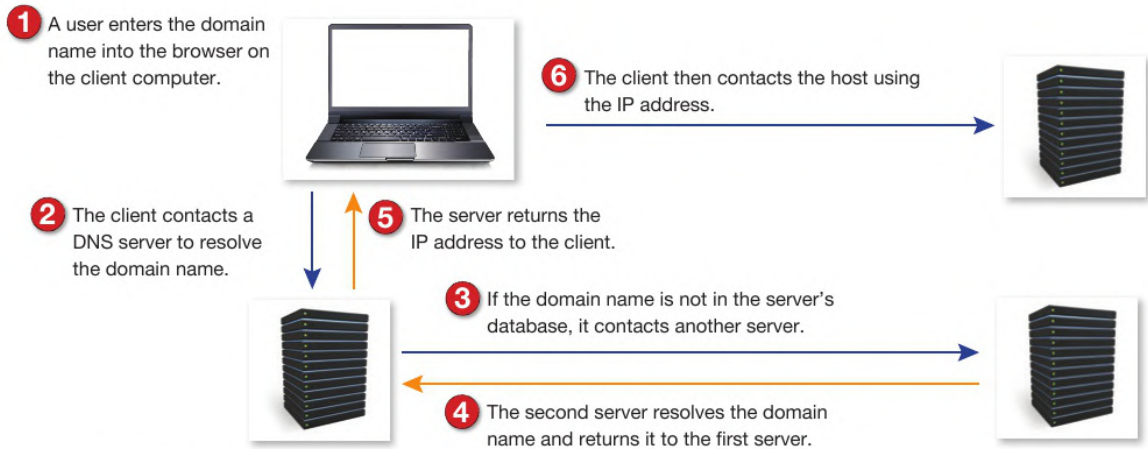
DOMAIN NAME SERVICE (DNS)

This is an Internet service that translates domain names into IP addresses. Because domain names are alphabetical, they're easier to remember. The Internet, however, is based on IP addresses.

Every time a user enters a domain name, a domain name service (DNS) must translate the name into the corresponding IP address. The DNS system is a network of servers. If one DNS server doesn't know how to translate a particular domain name, it asks another one, and so on, until the correct IP address is returned to the user.

SUBJECT VOCABULARY

domain name this is part of the URL for a resource on the Internet. When the domain name is used, it will be converted to the correct IP address by the domain name service (DNS) and the contact will take place



▲ Figure 5.8 Domain name service (DNS)

When a browser requests access to a host using its domain name, the client computer contacts one of the servers of the DNS. The DNS server contains a database of domain names which allows it to look up the domain name and return the IP address. This is known as 'resolving the domain name'. If it hasn't got that domain name in its database it will ask another server.

SUBJECT VOCABULARY

Internet service provider (ISP) an organisation that provides its customers with a connection to the Internet

router a piece of networking hardware used to forward packets of data from one network to another

Your computer or network is likely to be connected to the Internet using an **Internet service provider (ISP)**. An ISP is an organisation that provides Internet connections. This connection to the Internet can be provided in a number of ways, but the most common is through telephone lines.

The Internet uses the TCP/IP protocol stack to allow communication between all the different networks. More detail about protocols can be found in the section on Internet protocols earlier in the unit.

The Internet Protocol (IP) part of the stack provides the technical rules on addressing; that is, it provides each device or network connected to the Internet with a unique address to send data to/from.

As discussed earlier, two versions of IP, Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6), are currently in use.

The networks that are a part of the Internet are linked together using routers. A **router** is a piece of networking hardware that forwards packets between networks. A router has a routing table that is essentially a list of rules stating where to send packets for different destinations. When a router receives a packet it looks in the packet header for the destination address and then uses the rules within the routing table to decide where to send it. It is likely that a packet will need to be forwarded between several routers before it reaches its destination.

When an Internet-connected computer wants to send data to another whose IP address it already knows the following happens.

- 1 The sending computer splits the data into packets.
- 2 Each packet has a header that contains the sender's address, the destination address, the current packet number and the total number of packets that make up the data.

DID YOU KNOW?

The World Wide Web was invented in 1989 by a British scientist called Tim Berners-Lee. Can you find out when the Internet was invented?

- 3 Each packet is sent to your ISP.
- 4 Your ISP will have a router. This router will inspect the packet header and decide where to send the packet depending on its destination IP address.
- 5 The packet is likely to end up at another router, which will again look at the destination IP address and forward it on. This can happen many times before the packet reaches its destination network and intended recipient.
- 6 Once a packet reaches its destination, the receiving computer will put the data back together from the packets. Depending on the protocol being used, the packets might arrive in the wrong order and have to be put back in order using the information in the packet header.

The Internet, like any network, has many different services running on top of it. Email and the World Wide Web are two of the most commonly used services to which the Internet provides access.

SKILLS**RESEARCH,
CRITICAL THINKING****ACTIVITY 14****SERVICES USING THE INTERNET**

Find out what services other than the World Wide Web and email use the Internet. Create a list with a brief definition of each service, what protocols it uses and how it uses the Internet.

THE WORLD WIDE WEB

The World Wide Web (often shortened to WWW or 'the Web') is a service that runs on the Internet. It provides access to web pages (a type of document), which are linked together using a hypertext system. Hypertext is simply text that contains links (hyperlinks) that can be selected to go to another document/web page. These documents can contain text, graphics and multimedia.

ACCESSING THE WORLD WIDE WEB

You access the World Wide Web using a program called a web browser. There are many different web browsers to choose from, but they all use the same protocols and use the Internet to transfer information. The web browser's job is to convert the data received from a web server to a human-readable format. When a user accesses a website using the client-server system mentioned earlier in the unit the web browser is the client and the web server provides the service to the web browser.

HOW THE WORLD WIDE WEB WORKS

The World Wide Web uses the Internet to transfer data from one computer system to another. The computer being used to access the World Wide Web connects to the Internet using an ISP and will be running a web browser.

Getting a web page or other file using the World Wide Web involves a large number of steps that use different protocols and standards:

- 1 The user of a computer enters the web address of the information he or she wants to look at. A web address is also known as a URL (uniform resource locator). A URL may be for a web page or an individual file. For example, a URL could be just for a particular picture file rather than a web page.



SUBJECT VOCABULARY

HyperText Markup Language

(HTML) essentially a text document that contains any text to be displayed along with:

- details of how the text should be formatted (e.g. font size, colour, etc.)
- details of any hyperlinks and where they link to using a URL
- details of any objects such as pictures or videos that should be shown within the web page when it is displayed

► **Figure 5.9** How a web browser display of a web page is generated from HTML

- 2 The computer uses a system called the domain name service (DNS) to find the IP address of the required web server.
- 3 The web browser connects to the web server using the IP address and requests the relevant web page or other object (picture, sound or video file).
- 4 A web page is transferred from one computer (normally a web server) to another using HTTP or HTTPS, which were covered in 'Network protocols' earlier in the unit.
- 5 Data sent from a web server to a web browser is in **HyperText Markup Language (HTML)** format. The web browser displays the web page as described by the HTML.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

This is a Heading

This is a paragraph.

If the user selects a hyperlink, then the URL that the link points to will be loaded using the method outlined above. The URL might point to a web page on the same web server or on a different web server. This is where the 'web' part of World Wide Web comes from. Different pages and web servers are linked together in a complex pattern.

SKILLS

CRITICAL THINKING,
RESEARCH

ACTIVITY 15

INTERNET DATA TRANSMISSION

- 1 Show how a packet gets from one computer to another through a number of routers on the Internet by drawing an annotated diagram.
- 2 The Internet is capable of continuing to work even when some parts of the physical network fail. Research and discuss how the Internet deals with faults of this kind.

THE NEED FOR IP ADDRESSING
STANDARDS

An IP address is a set of numbers that are used to identify one particular computer.

The IP address is like a postal address. It allows Internet data and messages to be sent to the correct computer. It is therefore essential that they all conform to the same standard.

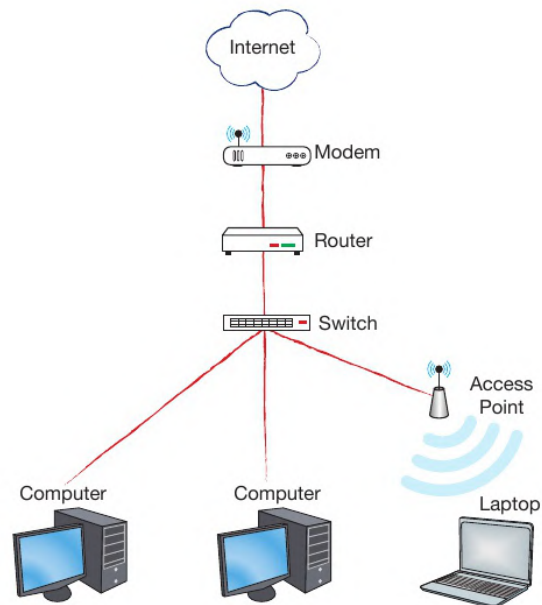
IP addresses originally consisted of four 8-bit numbers (octets) for version 4 addresses (IPv4), for example 216.27.61.137. This contains 32 binary bits and provides only 4 billion unique addresses.

IPv6 was introduced so that more addresses would be available. IPv6 uses 128 binary bits to create a single unique address on the network. An IPv6 address is expressed by eight groups of hexadecimal numbers separated by colons, as in 2001:cdba:0000:0000:0000:0000:3257:9652. Note that 128 bits provide 3.4×10^{38} unique addresses (approximately 4.3 billion).

COMPONENTS NEEDED TO ACCESS THE INTERNET

Devices are needed to connect to a local area network (LAN) and that LAN to a wide area network (WAN) such as the Internet. As mentioned previously, devices can connect to a LAN using either cables or wirelessly, using radio waves.

► **Figure 5.10** Devices allowing network users to connect to the Internet



SUBJECT VOCABULARY

switch a device that receives incoming data packets and redirects them to their destination on a local area network (LAN)

SWITCH

A **switch** is used to link the computers. Cables from each computer feed into it and so messages can be transmitted from one to the other. Switches are 'intelligent'. They can read the destination addresses and send them to only the intended computers rather than to all of them. They can do this because they build up a table of all of the addresses on the network. This cuts down on network traffic.

► **Figure 5.11** A switch



SUBJECT VOCABULARY

Wireless access point (WAP) a device that is used to connect one or more computers wirelessly to an existing physical wired network

modem short for 'modulator–demodulator'. It modulates and demodulates signals (i.e. converts them from digital to analogue and vice versa) sent from and received by a computer over a communications network

WIRELESS ACCESS POINT (WAP)

Wireless access points allow wireless devices to connect to a wired network using Wi-Fi. They convert data they receive through cables into a wireless signal and vice versa.

They are commonly used in public buildings to provide 'Internet hotspots'. They are similar to switches but cannot direct messages to particular devices: they transmit them to all.



► Figure 5.12 A wireless access point (WAP)

ROUTER

Routers are similar to switches because they read the address information, but they transmit the messages between networks.

A switch does this within a single network, but a router does this across several networks. When you view a web page on a server in the United States, the data is transferred from router to router, until it reaches your network. All in less than a second!

Routers are commonly used in the home to allow many computers to share an Internet connection. The router links the home network to the Internet. The router will transmit the incoming web pages, stream audio, etc. to the correct computer on the network. Routers can have both cable and Wi-Fi connections.



► Figure 5.13 A router

MODEM

A **modem** is needed to convert the signals in a LAN, such as a home network, into signals that can be transmitted along the cables provided by the Internet Service Provider (ISP).

The type of modem required will depend on the type of cable to the ISP, i.e. it could be a copper telephone line or cable, carrying electrical signals, or a fibre-optic cable which carries the signals as light.

To make setting up easier, modems are usually combined with routers in a single box and referred to as an Internet router.

SKILLS CRITICAL THINKING**ACTIVITY 16****COMPONENTS OF A NETWORK**

Karim & Abbas are a firm of solicitors who have three desktop computers in their office.

What components will they need to network these computers and access the Internet? The network must also allow members of staff to connect their laptops, tablets and smartphones to the network.

SKILLS CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING, REASONING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING, REASONING**SKILLS** CRITICAL THINKING**CHECKPOINT****Strengthen**

S1 Provide your own definition of what the Internet is.

S2 Provide a definition of what the World Wide Web is and add an explanation of how it is different from the Internet.

S3 What happens when you enter a web address into a web browser?

S4 Why are IP addressing standards needed?

Challenge

C1 In your own words, outline the role of all the protocols used when requesting a web page from a web server.

C2 Why do you think the Internet and World Wide Web are so embedded in modern life?

C3 What are the roles of a switch, a wireless access point, a router and a modem in a network?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, try finding some online videos that explain the concepts visually.

SUMMARY

- The Internet is a global network of networks. It is used to transfer data between different computer systems.
- The Internet has many different services running on top of it. Two common services are the World Wide Web and email.
- The World Wide Web runs on top of the Internet. Although people often mix the two terms up, they are not the same. As a learner studying computer science, you need to use the terminology correctly.
- IP addresses allow messages to be delivered between computers and therefore must be of a standard format.
- Hardware such as switches, routers, wireless access points and modems are needed to allow users to access LANs and WANs.

UNIT QUESTIONS

SKILLS

CRITICAL THINKING,
REASONING

A01

- 1 A bank wants to share data between its head office and its 400 branches. Explain why the bank would use a WAN rather than a LAN for this purpose. (2)
- 2 State the purposes of the following layers: application, transport layer and data link layer. (3)

HINT

Question 2 asks for you to state the purposes, so you do not need to give a detailed explanation. Use the total number of marks as a guide for how much you need to write.

SKILLS

CRITICAL THINKING

A01

- 3 A small business has a network consisting of three computers, one printer and a server. The network uses a star topology. Draw a diagram of the network and label all parts. (4)

HINT

Question 3 tests your knowledge of network topologies and the components within them. A good answer will show the network cables in the correct layout, any required network hardware and the correct items connected to the network. The diagram will need to be labelled to show what each item is.

SKILLS

CRITICAL THINKING,
REASONING

A01

- 4 Describe how an email phishing attack targeting bank customers might work. (2)

HINT

Question 4 tests your understanding of what a phishing attack is and how one might work. A good answer should describe the chain of events that might lead to someone entering their bank login details into a fake bank website.

SKILLS

CRITICAL THINKING

A02

- 5 Which of the following protocols would **not** be used when transferring a web page from a web server using the Internet? (1)
- a TCP
 - b IP
 - c IMAP
 - d HTTP

HINT

Question 5 is for 1 mark, therefore only one answer is correct.

SKILLS

CRITICAL THINKING

A01

- 6 Explain the difference between the Internet and the World Wide Web. (2)

HINT

Question 6 tests your understanding of how the Internet and World Wide Web are related. A good answer should state what the Internet and WWW are and how the WWW uses the Internet.

SKILLS

CRITICAL THINKING,
REASONING

A01

- 7 a Ayana is using her computer to access the Internet and has used an app to find out her IP address. The app informs her that it is 213.36.27.127.
- (i) Explain why a computer requires an IP address to access the Internet. (2)
 - (ii) An IP address consists of four numbers, each between 0 and 255. State how many bits will be needed to store the IP address. (1)
 - (iii) Describe the difference between IPv4 and IPv6 IP addresses. (2)
- b When Ayana accesses websites, she types in a name such as `https://qualifications.pearson.com/en/home.html` instead of the IP address. Discuss how Ayana is able to access the website using the domain name rather than the IP address. (4)

SKILLS

CRITICAL THINKING,
DECISION MAKING

A01

- 8 When Anika takes her laptop to college, she can connect it to the college network without using a cable.
- a State the devices on the network that allow her to do this. (1)
 - b Anika's family has a home network. Explain the role of a router in Anika's home network. (2)

HINT

There are 4 marks for part b of the question so you cannot just say, 'it connects networks together'. You must name the networks that are being connected and explain how the router ensures that all members of the household receive the correct data.

SKILLS

CRITICAL THINKING

A01

- 9 With the aid of a diagram, describe the role of a switch in a local area network. (6)

SKILLS

CRITICAL THINKING

A01

- 10 a State what is meant by the acronyms HTML and HTTP. (2)
- b Describe how HTTP is different to HTTPS. (4)



UNIT 6

THE BIGGER PICTURE

Assessment Objective 1

Demonstrate knowledge and understanding of the key principles of computer science

Assessment Objective 2

Apply knowledge and understanding of key concepts and principles of computer science

Assessment Objective 3

Analyse problems in computational terms:

- to make reasoned judgements
- to design, program, test, evaluate and refine solutions

In this unit you will learn about the effects of computers on the environment, both beneficial and harmful, and how these can be increased or reduced. You will also look at ethical and privacy concerns regarding computer systems and how their benefits should be shared by all people. Governments are working to ensure that the rule of law is imposed on all aspects of computer use and you will look at intellectual property and copyrights, patents and licensing. Finally, you will learn about emerging trends in computer technology such as quantum and DNA computing, artificial intelligence and nanotechnology.

24 COMPUTING AND THE ENVIRONMENTAL IMPACT OF TECHNOLOGY

HINT

Try to stay up to date with computer science news, so that you know about emerging technologies and are aware of current issues. The BBC's Click website is a good starting point. It's also worth checking out the technology section of the BBC website and subscribing to the *cs4fn* magazine.

SUBJECT VOCABULARY

computing technology an all-encompassing term referring to the hardware, software and infrastructure that underpin current and emerging computer systems

e-waste any form of discarded electronic equipment, including computing technology

GENERAL VOCABULARY

non-renewable materials resources that cannot be replaced naturally; only a certain amount exists

Everything we do has an impact on the environment and the manufacture, use and disposal of computing devices is no exception. We can, however, use computers to help us to combat our destructive effects.

LEARNING OBJECTIVES

- Explain how the manufacture, use and disposal of computing technology damages the environment
- Describe efforts being made to reduce the environmental damage caused by computing technology
- Give some examples of how computing technology is helping to protect the environment

It's hard to imagine what life would be like without the Internet, social media, search engines and e-commerce. Global demand for smartphones, tablets and other forms of **computing technology**, including embedded processors, web servers, sensors and hard drives, is growing rapidly year on year. At the same time, the pace of new product releases and consumers' wishes to own the latest model are shortening the lifespan of these devices. The average life expectancy of a smartphone, for example, is estimated to be less than two years. The more computing technology we buy, the more we throw away.

The manufacture, use and disposal of computing technology has a significant impact on the environment. We are using up resources of **non-renewable materials**, creating huge piles of potentially harmful **e-waste**, consuming vast quantities of energy and damaging people's health.

MANUFACTURE

Manufacturing a smartphone, PC or any other piece of computing technology is a complex process. It starts with gathering raw materials and ends with the finished item being sent to the customer, with lots of stages in between. This makes it difficult to determine accurately the overall environmental impact (although there's no doubt that it's considerable).

RAW MATERIAL EXTRACTION

A number of non-renewable natural resources are used in the manufacture of computer products. They include sand (to make glass for screens), oil (used to make plastics) and various metals used in wiring and circuit boards. Some of the metals used, such as silver, gold, copper and palladium, are precious and in short supply. Others, such as arsenic, cadmium and chromium, are hazardous and pose a serious health risk. **Radioactive** metals used in computer products, such as uranium and thorium, can **contaminate** air, soil and groundwater, and are toxic to human health.

In some regions of the world, mining of raw materials is not properly controlled. Extraction causes extensive damage to the local environment, damaging

GENERAL VOCABULARY

radioactive having or producing energy that comes from atoms breaking up

contaminate making something dirty or polluted

DID YOU KNOW?

China is the world's largest producer of rare-earth metals – a group of 17 chemical elements that, due to their unique magnetic, luminescent and electrochemical properties, help improve the performance of computing technology and make it more energy efficient.



the landscape with holes and waste heaps, contaminating water supplies and putting wildlife habitats in danger. Poorly equipped miners working in dangerous conditions run the risk of being seriously injured. They are also at risk of suffering long-term breathing illnesses, such as silicosis, bronchitis or lung cancer.

PRODUCTION

Once extracted, the raw materials are sent to factories – often thousands of miles away – to be manufactured into components, such as circuit boards, chips, screens, disk drives and cases. In turn, the components are sent onwards for assembly into finished products.

The manufacture of computing technology uses a lot of energy. Large amounts of non-renewable fossil fuels, such as coal and oil, are used during the process. Burning such fuels produces carbon dioxide (CO₂) and contributes to global warming.

Semiconductors are present in every piece of computing technology. Manufacturing semiconductors uses a very large amount of water. For example, a factory producing 40 000 semiconductors a month uses around 20 million litres of water a day – it compares with the consumption of a city with a population of 60 000. This can result in water shortages in areas where semiconductor factories are located. Untreated wastewater discharge can cause environmental pollution.

The United Nations University has estimated that the manufacture of a computer and monitor weighing 24 kg requires ten times the amount of fossil fuels (240 kg). This is approximately the same weight of chemicals and around 1500 litres of water.

GENERAL VOCABULARY

semiconductor a substance that can conduct electricity under some conditions but not others, making it a good medium for the control of electrical current

exposure being affected by something harmful because you are in the same place as it

This table lists six of the most hazardous materials used in the manufacture of computing technology.

Research indicates that **exposure** to these materials is harmful to human health, causing both physical and neurological damage. Furthermore, chemical emissions and wastewater from manufacturing plants put people living nearby at risk. There is growing recognition of the need to address this problem. Most countries have regulations restricting the use of hazardous substances,

The EU Restriction of Hazardous Substances (RoHS) Directive restricts the use of all six of the materials listed in the table above. This forces computing technology manufacturers to replace them with safer materials. The China RoHS2, published by the Chinese authorities in 2019, introduces a voluntary certification system to demonstrate compliance with hazardous

MATERIAL	EXAMPLES OF USE
Cadmium	A metal used in the manufacture of rechargeable batteries, printer inks and toners.
Lead	A metal used in the manufacture of circuit boards and cable sheathing.
Mercury	A metal used in the manufacture of LCD screens.
Hexavalent chromium	A chemical compound used to make casings.
Polychlorinated biphenyls (PCBs)	Toxic compounds added to plastics, circuit boards and connectors to make them more fire retardant.
Polybrominated diphenylethers	

GENERAL VOCABULARY

impose to force a rule to be followed

substances. It lists the same six hazardous substances as the EU directive. The United States has similar regulations in the Toxic Substances Control Act (TSCA).

At the same time, governments are **imposing** tough recycling targets. These are designed to ensure that more reusable material is from unused computing technology. The aim is to reuse the material so that fewer raw materials are needed and reserves of limited resources are protected.

Growing public awareness is putting pressure on manufacturers to improve working conditions in their plants and impose stricter requirements on their component suppliers.

USAGE

DID YOU KNOW?

A lot of energy is wasted while a computer or printer sits idle. Using the 'sleep mode' when a device is not in use can reduce consumption by more than 50 per cent.

The amount of energy consumed in the manufacturing process seems insignificant when compared with the energy required to keep mobile phones, computers, networks, telecommunication links, etc. up and running day after day. Even though each individual device doesn't require a huge amount of electricity, close to 2 billion connected PCs and laptops and more than 6 billion mobile devices collectively do.

In recent years, considerable efforts have been made to improve the energy efficiency of computing devices. However, the amount of energy they actually consume depends on how they are used and what they are used for. The task that a computer is performing and the software being used are key determinants of the actual energy usage. High-end applications, complex calculations, 3-D modelling and video games use up a lot of power.

Cloud computing (see Unit 4) and data centres in particular are major energy users. Vast amounts of electricity are needed to power and cool all the computer equipment that is needed, putting them ahead of the aviation industry in terms of the energy they consume. The worst offenders are the small, inefficient data centres hosted by private organisations and government departments. They tend to be far less efficient than the large facilities operated by cloud providers such as Google and Apple.

SUBJECT VOCABULARY

carbon footprint the amount of carbon dioxide an individual or organisation produces as a result of the energy they consume

Energy efficiency measures and the use of renewable energy can significantly reduce the **carbon footprint** of data centres. Facebook, for example, has built a huge data centre in northern Sweden, just 100 km south of the Arctic Circle. Its location was selected because of its access to renewable hydroelectricity and the cold climate that helps to keep the servers cool.

SKILLS RESEARCH

ACTIVITY 1

LAPTOP VERSUS DESKTOP

Research why using a laptop rather than a desktop is more energy efficient.

SKILLS RESEARCH, CRITICAL THINKING

ACTIVITY 2

ENERGY USE BY DATA CENTRES

Data centres consume large amounts of energy.

- 1 Research what this energy is used for.
- 2 List four measures that can be taken to make data centres more environmentally friendly.

DISPOSAL



The disposal of unused computing technology represents another serious threat to the environment. The quantity of e-waste is growing at a tremendous rate. According to the UN's StEP Initiative, e-waste will soon weigh as much as eleven of the great Egyptian pyramids.

Although great efforts are now being made to recycle more e-waste, large amounts are still sent overseas to developing countries where they are dumped in landfill sites. This can have serious consequences for the environment and public health. The problem is made worse by the fact that the developing nations themselves are generating more and more waste of their own.

E-waste that is not recycled properly can be a serious health and environmental issue. As you know, computer products contain a whole host of dangerous materials. Once a computer is dumped in a landfill site, the likelihood is that some, if not all, of these toxic substances will leak out into the ground, contaminating water supplies, infiltrating the food chain and polluting the air.

For every 1 million mobile phones, 24 kg of gold, 250 kg of silver and 9 tonnes of copper can be recovered. The presence of these valuable metals in old computing technology is a strong reason for local people living near the landfill sites, many of whom are desperately poor, to try to recover them. However, taking apart old computer equipment without protective clothing and specialist training is extremely dangerous. People who do so risk exposure to hazardous materials such as mercury and lead and are in danger of inhaling toxic fumes.

DID YOU KNOW?

Electronic waste is expected to top 52 million tonnes globally by 2021 with only 20 per cent of it expected to be collected and recycled.

SKILLS

RESEARCH, CRITICAL
THINKING

ACTIVITY 3

EXTENDING THE LIFESPAN OF TECHNOLOGY

'One person's rubbish is another person's treasure.' Research, then briefly describe two initiatives (new schemes) that aim to extend the life of pre-owned computing technology.

Most countries around the world set regulations and targets for the collection, recycling and recovery of computing technology and other electronic items. The aim of responsible recycling is to recover valuable metals and reusable components such as plastic, glass and metal, and to dispose of dangerous substances safely.

Major manufacturers of computing technology now have recycling programmes. In developing countries there are some promising initiatives to create **state-of-the-art** recycling plants that can turn e-waste into an **e-opportunity**.

GENERAL VOCABULARY

e-opportunity a chance to benefit by using electronic devices

state-of-the-art the best and most modern of its type

SKILLS

CRITICAL THINKING,
DECISION MAKING

ACTIVITY 4

REDUCING THE ENVIRONMENTAL IMPACT

- 1 Provide three possible environmental impacts of using computing technology.
- 2 Can you think of one possible action that could be taken to reduce each of them?

DID YOU KNOW?

Green computing refers to the efficient use of resources in computing. It makes companies and individuals responsible for producing and using technology with a minimal carbon footprint. The main emphasis is on energy efficiency, use of safer materials during the manufacturing process and promoting environmentally friendly systems.

PRESERVING THE ENVIRONMENT

Things are not all bad. Computing technology can help to fight climate change, provide disaster warnings, protect endangered species and habitats, and reduce energy consumption.

CLIMATE CHANGE

NASA is analysing satellite data and measuring sea surface temperatures to learn more about how and why sea levels are rising.

Networks of wireless sensor probes are used to gather information about glaciers. The probes are placed under the surface of the ice and measure temperature, pressure, stress, weather and sub-glacial movement. A base station collects the data from them. The system is helping scientists to understand more about the speed at which glaciers are melting.

Researchers at the University of Oxford are using spare home computer time to establish if global warming is to blame for heavy flooding in the UK in recent years.

EARLY WARNING

Tsunami early warning systems use sensor networks to detect approaching tsunamis and a communications infrastructure to issue timely warnings so that coastal areas at risk can be evacuated.

CONSERVATION

Information from GPS and satellites is being used to track Malaysian elephants. The results are analysed by computer to help improve conservation strategies and assess the effectiveness of the Malaysian Government's elephant conservation programme.

Miniature transponders fitted to bees allow scientists to study the effects of disease and pesticides.

Mobile phones are being used to listen out for illegal logging activities in the rainforest and provide rangers with real-time alerts.



▲ A lioness with a tracking device. GPS collars help to monitor and conserve wildlife

HINT

Make sure you give real positive and negative examples of how computing technology affects the environment.

ENERGY

Engineers at Manchester Metropolitan University in the UK are working on a project to make buildings more energy efficient. Sensors in each room monitor light levels, temperature, how many people are present and electricity consumption. Real-time analysis of the room data enables automatic adjustment of electricity usage.

The latest generation of giant solar energy farms uses sensors to track the movement of the sun. Computer-controlled motors adjust the position of the solar panels to optimise power generation. Developments in battery storage allow surplus electricity to be stored.

SKILLS CRITICAL THINKING**SKILLS** PROBLEM SOLVING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**CHECKPOINT****Strengthen**

- S1** Can you think of two hazardous substances that are used in the manufacture of computing technology?
- S2** Why is dumping e-waste in landfill sites harmful to the environment?
- S3** List two ways of reducing the environmental damage caused by data centres.
- S4** List two ways in which computing technology is helping to preserve the environment.

Challenge

- C1** Summarise the health risks associated with the manufacture and disposal of computing technology.
- C2** What are three ways in which computing technology can help to reduce energy consumption?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section and have another go at the activities.

SUMMARY

- Some of the materials used in the manufacture of computer components are non-renewable and in short supply. Others are dangerous and pose a risk to human health.
- The Restriction of Hazardous Substances (RoHS) regulations around the world restrict the use of hazardous materials in computing technology, forcing producers to find more environmentally friendly alternatives.
- Computing technology consumes huge amounts of energy. Data centres are one of the worst offenders.
- Energy efficiency measures and the use of renewable energy can significantly reduce the carbon footprint of computing technology.
- There is a possible health risk, especially for children, from exposure to the electromagnetic fields generated by wireless devices, such as smartwatches and smart clothing.
- Disposal of e-waste in landfill sites without controls in place poses a significant threat to the environment.
- Regulations around the world set targets for responsible recycling of e-waste.
- Computing technology is helping to preserve the environment in a number of ways, including monitoring and modelling climate change, conservation and smart energy.

25 PRIVACY

There are concerns about the privacy implications of computer technologies, such as face recognition and automatic number plate recognition, and the impact this might have on the individual. You will look at how these computer technologies collect personal data and how they might threaten privacy.

LEARNING OBJECTIVES

- Explain why computing technology poses a threat to privacy
- Weigh up the benefits and drawbacks of giving away personal information

SUBJECT VOCABULARY

ethics a set of moral principles that govern a person's behaviour

privacy the right to be left alone and free from unwanted scrutiny and intrusion

personal data information that is personal and unique to an individual

GENERAL VOCABULARY

ethical concerned with what is morally right or wrong

invasion of privacy when your right to be alone and not watched by other people is taken away without your permission

Now that you know about the damaging environmental impact that computing technology has, does it make you have second thoughts about swapping your smartphone for the latest model? Is your concern for the environment more important than your 'need' for a new phone? If so, you are making an **ethical** decision. **Ethics** relate to what is right and wrong and govern a person's behaviour.

An action might be legal, but not necessarily ethical. For example, it's perfectly legal to leave your old desktop computer gathering dust in the attic. But if you know that someone in a developing country would benefit from having it, the right thing to do might be to pass it on to them.

Computing technology confers a wide range of social and economic benefits, but it also creates challenging ethical issues. **Privacy** and security are two of them. While most people would agree that computing technology has helped to create a much more open society, some would argue that it comes at too high a cost. The amount of personally identifiable information that is gathered, stored and analysed represents a massive **invasion of privacy**.

PERSONAL DATA

DID YOU KNOW?

Facebook has the biggest database of faces in the world, with over 2.2 billion photos posted and tagged to the website every day. You may receive a message from Facebook letting you know that you appear on someone's photo even though you are not tagged. Facebook knows what you look like!

Every time you post an update on social media, sign up for an online account, use a web-based email service or a search engine you are adding to a store of **personal data** that is held about you. This includes information about where you live, what you look like, who your friends are, your likes and dislikes, your bank account details, products you're interested in buying, the places you visit. This personal data is stored on servers that belong to online services, such as Facebook and Google, not to you.

Every organisation you come into contact with, not just the online companies, is likely to collect information about you. Your school, for example, stores your attendance record, your end-of-year exam results, which books you've borrowed from the library, the after-school activities you take part in and much more besides. Does this worry you?

Some people are very concerned about the amount of personal information that is collected. It is often without their permission and they believe they have

SUBJECT VOCABULARY

identity theft the stealing of another person's personal details, such as their bank account number, sort code or passport number, for the purpose of making purchases and running up debts in their name

little or no control over this. They are worried about who has access to it, what they are using it for, how secure it is and how accurate it is.

Weak security could result in personal information falling into the wrong hands. This makes people vulnerable to phishing attacks, scams, **identity theft** and fraud.

Sometimes the information is inaccurate, but getting it changed or removed is extremely difficult, if not impossible. It is quite common for inaccurate information about a person to follow them throughout their entire life, affecting how they are seen and treated by others.

DID YOU KNOW?

Around the world, data protection legislation controls how governments and organisations can use personal data. Many of them follow similar guidelines, such as:

- Data must be processed fairly and lawfully.
- Data must be obtained and used only for the specific and lawful purposes for which it was collected.
- Data must be adequate, relevant and not excessive.
- Data must be accurate and up to date.
- Data must be kept for no longer than necessary.
- Data must be kept secure.

SKILLS

RESEARCHING,
CRITICAL THINKING

DID YOU KNOW?

Cookies are small data files that keep a record of your web browsing history. They record which websites you visit and how often, which products or services you buy or show an interest in. Cookies enable online stores to learn a lot about you. There is legislation in Europe that gives consumers the right to opt out of having data collected in this way.

Unlike a cookie, spyware is a computer program stored on your hard drive (usually without you realising it's there) that collects information about you and transmits it to a third party. It represents another serious threat to privacy.

ACTIVITY 5

SEARCHING FOR YOURSELF

What information can you find out about yourself by typing your name into a search engine? Is it accurate? What sort of impression of you does it portray?

One reason why so many people give away information about themselves is that it enables an organisation to understand their needs better and provide them with a more personalised service. For example, setting up an account with an online supplier makes it faster and more convenient to purchase from them.

But are these benefits more important than the drawbacks?

You might not mind that an online retailer knows who your favourite band is if it means you get to hear quickly when their next album is released. But is it right to target a financially vulnerable person with adverts for products they will want but can't afford?

BIG DATA

Data analysts are able to learn more and more about us and gain insights into our behaviour by analysing huge volumes of personal data gathered from various sources.

Analysis of so-called 'big data' can benefit society. For example, by helping to identify adverse side effects of drugs that might otherwise go unnoticed, optimising energy use in cities and providing insights into the spread of disease.

But is the price we pay too high? Where do we draw the line? Big data comprises large amounts of information, each piece of which on its own could be seen as being harmless, for example your phone number, what music you download, your hobbies, etc. However, when collated together these individual bits of information produce a very accurate, detailed profile of an individual. It reveals far more about them than they might have disclosed. This might lead to the individual experiencing identity theft or an intruder illegally accessing personal information through social engineering.

SKILLS RESEARCH

ACTIVITY 6
THE BENEFITS OF BIG DATA

Find two examples of how society is benefiting from big data analysis.

SUBJECT VOCABULARY

surveillance technology CCTV, drones, number plate recognition, bugging and tracking devices used to monitor and record people's activities, often without their knowledge

whistle-blower someone who draws attention to the activities of an organisation or person believed to be acting illegally or unethically

SURVEILLANCE

Have you any idea how often you've been watched on CCTV today? Could a drone have been hovering overhead taking aerial photographs of you on your walk to school? If you've driven anywhere by car, travelled by public transport or been in a shop, the chances are you've been recorded by some form of **surveillance technology**.

Most people are willing to allow the security forces to use surveillance technology to track people's movements and tap their phones if it keeps them safe from harm. But what if it were used by companies to monitor your shopping habits or by criminals noting the time you leave the house each morning? It's quite common for employers to use hidden cameras to check up on their staff. Is this acceptable?

Some people believe that use of surveillance technology goes too far. In 2013 Edward Snowden, a so-called '**whistle-blower**', raised awareness of the extent to which governments worldwide are now monitoring and spying on their citizens.

SKILLS DECISION MAKING, CRITICAL THINKING

ACTIVITY 7
INVADING PRIVACY

The right to privacy must be balanced against the needs of society. In a group, think about two situations in which you believe an invasion of privacy is justified. Present your arguments.

DID YOU KNOW?

Police forces can track vehicle movements in real time using automatic number plate recognition and CCTV. Records can be kept for two years to be analysed for intelligence. CCTV combined with facial recognition software also enables the police to identify and track specific individuals.

SUBJECT VOCABULARY

location-based services services that enable people to access and share real-time location information online

SKILLS CRITICAL THINKING

LOCATION-BASED SERVICES

With the help of **location-based services** and Wi-Fi, people can share their current location, arrange to meet up with friends nearby, check in to a venue, find their way to a particular location and much more. A disadvantage is that location-based services also allow other people to track your movements, find out where you live and what you are doing. This can be dangerous and represents a huge invasion of privacy.

ACTIVITY 8

LOCATION-BASED SERVICES

Can you provide three ways in which location-based services can benefit a user? Can you also provide three risks associated with their use?

PRIVACY-ENHANCING TOOLS

Privacy-enhancing tools, while not 100 per cent effective, do give some protection against privacy invasion. This table lists some of the most popular of these tools.

TOOL	PURPOSE
Encryption	Prevents unauthorised people from reading your data.
Cookie cleaners, anti-spyware and ad blockers	Software that detects and removes cookies, spyware and adware installed on your computer.
Identity management service	A trusted third party holds evidence of your identity and issues you with an identifier that enables you to conduct transactions with other parties without revealing any personal information about yourself.
Password manager	Stores all your website login information in an encrypted password database with a master password, which is the only one you have to remember.

SKILLS CRITICAL THINKING

GENERAL VOCABULARY

implications the effects of a decision

ACTIVITY 9

ONLINE DATA BREACHES

Through online data breaches, personal and financial details stored by a company can be stolen. In pairs, research the **implications** of this breach for both the customers and the company.

DID YOU KNOW?

Ransomware is a type of malicious software. It blocks access to a computer system until a sum of money is paid. It is often delivered by emails which contain attachments. The 'biggest ransomware outbreak in history', caused by software called WannaCry, occurred in 2017. It infected computers in 150 countries, including the international shipper FedEx, Spanish telecommunications company Telefonica, Telefonica Argentina and the National Health Service (NHS) in the United Kingdom. The most affected countries were Russia, Taiwan, Ukraine and India.

SKILLS CRITICAL THINKING**SKILLS** CRITICAL THINKING,
PROBLEM SOLVING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING,
PROBLEM SOLVING**CHECKPOINT****Strengthen**

- S1** Can you think of two ways in which an individual's personal data could end up stored in databases owned by a third party?
- S2** What might happen if personal information falls into the wrong hands?
- S3** List two privacy-enhancing tools and describe what they do.

Challenge

- C1** Why do some people decide that the benefits of revealing personal information about themselves are worth the disadvantages?
- C2** Can you think of a situation where the right to privacy is less important than the needs of society?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section and have another go at the activities.

SUMMARY

- Computing technology enables organisations to gather, store and analyse vast quantities of personal information about the people they come into contact with.
- Individuals give away all sorts of personal information about themselves online.
- Collecting and analysing information about the people they come into contact with enables organisations to provide a more personalised service.
- Big data analysis benefits society at the expense of many individuals' privacy.
- Surveillance technology helps keep us secure but invades our privacy. It is difficult to determine what level of surveillance is acceptable.

26 DIGITAL INCLUSION

Digital devices have had many beneficial effects on the lives of people using them. But not everyone has access to digital devices, leading to inequality in society. Governments around the world have designed legislation and policies in the hope that the benefits are spread to all.

LEARNING OBJECTIVES

- Explain how people benefit from being 'technology-empowered' and the disadvantages of being 'technology-excluded'
- Describe measures that are being taken to promote digital inclusion

SUBJECT VOCABULARY

digital inclusion ensuring that everyone has **affordable** access to computing technology and the necessary skills to take advantage of it

digital divide the gap between people who are technology-empowered and those who are technology-excluded

GENERAL VOCABULARY

affordable cheap enough so that most people can afford to buy it

Computing technology is a great enabler, giving many people access to news, information, products and services at any time, wherever they are. But those people who neither have the opportunity nor knowledge to use this technology are excluded from the advantages it provides. Is this fair?

Digital inclusion is about providing everyone with affordable access to computing technology and the skills to use it. The gap between those who are 'technology-empowered' and those who are 'technology-excluded' is known as the **digital divide**. There is a digital divide between industrialised and developing countries and also between people who live in the same country.

IMPACT

There are many reasons why technology exclusion is not a good idea.

TOOL	PURPOSE
Information and services	The Internet is becoming the default option for accessing information, public services and entertainment.
Employment	Having poor digital literacy skills makes it harder to find a job and limits employment opportunities. Individuals might only have access to poorly paid work with little prospect of progression.
Democracy	The Internet gives people a voice and lets them express their views to a worldwide audience. This is particularly important where citizens have limited freedom of expression.
Economic growth	Businesses that are able to exploit computing technology to the full have a competitive advantage over those that can't.
Saving money	Paying bills and shopping online often saves consumers money and gives them better protection.
Social isolation	Having access to the Internet helps people to keep in touch with friends and relatives.

TOWARDS DIGITAL INCLUSION

DID YOU KNOW?

Project Loon, an initiative by Google, hopes to increase global connectivity using balloons flying 20 km up into the stratosphere. Loon has taken the most essential components of a cell tower, then redesigned them so they are light and durable enough to be carried by a balloon. Solar panels power the equipment.

There are many initiatives to promote digital inclusion, such as:

- A subsidy programme for low-income households has been started in Colombia by the government. It includes subsidies for fixed Internet access. Also, in partnership with the private sector, schools in remote areas have been connected via satellite.
- The government of India has created an online gateway of public information and on-demand services, such as national identity and passport services, employment as well as tax and pension information and services. This is to enhance the relevance of Internet content for its citizens.
- Kenya has established a digital learning programme to drive primary education to enhance digital skills and improve the quality of education.
- The government of Thailand have established more than 20 rural Internet centres nationwide. The centres strengthen ICT skills among students, youth and local communities and help to promote social and economic development in some of the most remote areas of the country.

According to the World Bank, 77 per cent of the world's population already live within range of a mobile phone network. In areas with a limited or non-existent landline infrastructure, mobile phone technology can fill the gap. Even though the number of phones per 100 people in poor countries is much lower than in the developed world, they are having a huge impact.

SKILLS**RESEARCH,
COMMUNICATION****ACTIVITY 10****MOBILE PHONE USE**

Using your own online research, find some examples of how mobile phones are being used in different countries to encourage the use of digital resources. Write a brief report to summarise your findings.

SKILLS**CRITICAL THINKING****ACTIVITY 11****REDUCING DIGITAL EXCLUSION**

Can you name five actions a government can take to reduce digital exclusion?



- ▲ Ambitious digital literacy programs are under way in India, Kenya, Colombia and elsewhere to ensure that people have the know-how and skills they need to exploit digital technology to the full

DID YOU KNOW?

One of the advantages of mobile data access is the ability to use GPS to plan routes. However, in many rural and remote areas around the world, there are no addresses, postcodes or zip codes to enter into a navigation app. Longitude and latitude coordinates are very long and difficult to enter correctly (e.g. 51.74743863117572).

To solve this problem, a system called 'what3words' divides the surface of the globe into a grid of 3m x 3m squares. Each one is assigned a unique three-word address.

Therefore, you can follow a route to anywhere in the world – it doesn't have to have a road or address. Have a look at the web site and find out your three-word address: <https://what3words.com/daring.lion.race>

SKILLS CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**CHECKPOINT****Strengthen**

- S1** What is meant by the terms 'technology-empowered' and 'technology-excluded'?
- S2** List three drawbacks of being 'technology-excluded'.
- S3** List four factors that contribute to the digital divide.
- S4** Can you think of two ways of providing access to the Internet in areas with a poor landline infrastructure?

Challenge

- C1** Access to the Internet is a key factor in reducing the digital divide. List two further measures that governments can take to promote digital inclusion.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section and have another go at the activities.

SUMMARY

- Digital inclusion means that everyone has affordable access to computing technology as well as the skills to use it.
- A number of factors contribute to the digital divide – lack of, or poor connectivity is one of the main ones.
- Someone who is technology-excluded misses out on all the opportunities computing technology offers.

27 PROFESSIONALISM

Professionalism refers to the skill and competence expected of a person in a professional setting. You will look at what this means in relation to the world of computer science.

LEARNING OBJECTIVES

- Explain what professionalism means in the context of computer science

GENERAL VOCABULARY

start-up a company that is in the first stage of its attempt to develop a product or service for which they believe there is a demand

demonstrate to show how you are feeling through your actions

professionalism the skill and competence expected of a person in a professional setting

Computer scientists write software to make computers do new things or accomplish tasks more efficiently, create mobile apps, design and build embedded systems, devise security policies, invent new products and much more besides.

Some work for big multinational computing companies, such as Microsoft and Apple. Others for small **start-ups**; some are self-employed, some are employed in the IT departments of organisations such as hospitals, universities and companies.

Wherever they work they are expected to behave ethically and **demonstrate professionalism**.

Most countries have ethical guidance for computer professionals. The British Computer Society (BCS) is the Chartered Institute for IT. The BCS Code of Conduct sets out the professional standards its members are expected to uphold. Among many other matters, it specifies that computer scientists should always respect the wellbeing, privacy and security of the environment and people, never stop learning and gaining skills and professional knowledge. For a full list of the code of conduct of the BCS please visit their website.



DID YOU KNOW?

Most computer scientists pay an annual membership fee to belong to a professional association. This gives them access to specialist technical conferences, training and publications. It also provides them with an opportunity to interact and share knowledge and expertise with each other. Having membership of a professional association on their curriculum vitae (also known as a résumé) sometimes carries weight when a computer scientist is applying for a new job.

SKILLS

CRITICAL THINKING,
COMMUNICATION

ACTIVITY 12

ETHICAL BEHAVIOUR

A computer scientist is developing an embedded system to control car brakes.

In a group, decide how she should respond if:

- 1 she spots a bug, but knows that fixing it will delay completion of the project
- 2 she's aware that there is a remote possibility that someone could hack into the code and stop the brakes from working
- 3 a competitor offers to pay her for details of the code
- 4 her employer makes claims for the system that are not entirely true.

Are there any other ethical dilemmas she might encounter as a professional?

SKILLS

CRITICAL THINKING

CHECKPOINT

Strengthen

S1 What does the BCS Code of Conduct say a computer scientist should do?

Challenge

C1 What might 'professionalism' mean to a computer scientist?

SKILLS

CRITICAL THINKING

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section and have another go at the activities.

SUMMARY

- Computer scientists must abide by the law.
- They should behave ethically by adhering to a professional code of conduct.
- It is important that programmers demonstrate professionalism as the work they do could put the lives of other people at risk.

28 COMPUTING AND THE LEGAL IMPACT OF TECHNOLOGY

Human activities must be observed to ensure that they do not have negative impacts on others. Where these are very serious or cannot be prevented by encouraging good practice, laws have to be followed to protect individuals. In this section you will look at some of these laws.

LEARNING OBJECTIVES

- Describe how copyright and patents can protect intellectual property
- Explain the purpose of a software licence
- Differentiate between open-source and proprietary software and outline the benefits and disadvantages of each
- Be aware of Computer Misuse Acts to enhance cybersecurity

INTELLECTUAL PROPERTY

Intellectual property (IP) – not to be confused with an IP address – is a unique creative product of a human mind. A piece of software, a computer game, a design for a new processor, a digital image, a piece of music and a literary work are all examples of IP. Each of them was created by somebody, is unique and has a commercial value.

SUBJECT VOCABULARY

intellectual property (IP) a creation of the human mind that is unique and has a commercial value

patent an exclusive right granted to an inventor to make, use and sell an invention for a fixed period of time

COPYRIGHT AND PATENTS

There is legislation in some parts of the world that makes it illegal to copy, modify or distribute intellectual property without permission. However, there are a multitude of peer-to-peer networks, torrent sharing websites and forums on the Internet that allow people to download copyrighted software without paying for it. This is illegal and also unethical, since it means that the programmer who wrote the code doesn't get the money that is due to them.

Copyright only protects the expression of an idea, not the idea itself. So if you were to develop an original piece of software, its source code would be protected. But, there's nothing to stop someone else from copying the idea and writing a program that essentially performs the same task. You would have to prove that the similarities between the two programs are more than just coincidence and can only be explained by copying.

The © symbol indicates that a piece of software, a movie or some other type of artefact is protected by copyright.

A **patent** offers more protection than copyright. It protects the idea or design of an invention, rather than just a particular form of it. In order to get a patent you have to be able to demonstrate that what you have invented is distinct from anything else that already exists. A patent holder has the exclusive right for 20 years to make, use and sell their invention.

DID YOU KNOW?

The creators of the illegal software sharing website, The Pirate Bay, received a prison sentence and were fined over US\$4 000 000 for hosting hyperlinks to illegally obtained software and media.

DID YOU KNOW?

In 1994, Apple agreed to license parts of its GUI (see Unit 4) to Microsoft for use in the first version of Microsoft Windows®. When Microsoft released Windows 2.0, it used the overlapping windows feature of the Macintosh OS, which was not included in the original licence agreement. Apple then sued Microsoft for copyright infringement.

EXTEND YOUR KNOWLEDGE

Research the laws affecting computer use in the country in which you live.

DID YOU KNOW?

For years, various smartphone manufacturers including Apple and Samsung® have been fighting in the courts over patent infringements. Apple claimed that Samsung had stolen the 'look and feel' of its iPhone® and used it in the Galaxy® smartphone.

There is a real concern that these so-called 'patent wars' will discourage people from being imaginative with their inventions. This refers to manufacturers spending their money on lawsuits rather than on new inventions. This would be very bad news for consumers.

SKILLS

CRITICAL THINKING,
COMMUNICATION,
CREATIVITY

ACTIVITY 13**INTELLECTUAL COPYRIGHT**

- 1 Get into groups of three. Each person in the group should describe how they would feel if an app they had created was made available for free on the Internet, without their permission.
- 2 Create a podcast for software developers, explaining how software can be protected as intellectual copyright.

LICENSING

Every piece of software, even if it is free, has a licence. Even though the user purchases a piece of software, the licence states that they don't actually own it. The licence allows the buyer to use the software subject to the licence terms, but the manufacturer retains ownership. Before you can install the software, you have to agree to the terms of its licence. These specify:

- how many copies of the software you are allowed to use
- whether you can install the software on more than one computer
- what type of organisation can use the software – some licences are for charities, students or home users only
- how long the software can be used for – perpetual software licences last forever, but some licence agreements expire unless you renew them.

You are usually not allowed to resell the software. Paid-for software is often supplied with a unique licence key, which certifies that the software is genuine and prevents illegal copying.

SUBJECT VOCABULARY

Creative Commons an organisation that allows people to set copyright terms for their intellectual property. One use of a Creative Commons licence is to allow people to copy material as long as it is not used commercially

If a computer scientist wants to permit other people to use their code without charge, they can use an open-source licence to specify what restrictions (if any) there are. For example, in the UK a **Creative Commons** licence provides a way for the creator of a piece of music, a photograph or other form of intellectual property – including software – to allow other people to use it providing they abide by the conditions specified in the licence.

OPEN-SOURCE AND PROPRIETARY SOFTWARE

Open-source software is freely available on the Internet. Anyone is permitted to edit the code and pass it on to others, providing they don't charge a fee.

The advantages and disadvantages of open-source software are shown in this table.

SUBJECT VOCABULARY

open-source software software that is free to edit and redistribute

proprietary software software that belongs to an individual or a company. Its licence specifies that users are not allowed to modify the source code and places restrictions on its use

ADVANTAGES	DISADVANTAGES
It is free to use	It might not be particularly user friendly and might look unprofessional
It can be modified	There might be little or no technical support available
It can be used to demonstrate programming concepts	Criminals may be able to identify and exploit vulnerabilities in the code

DID YOU KNOW?

The Python programming language is open source and is maintained by the Python Software Foundation. Members of the programming community often contribute to the source code to build on the language.

DID YOU KNOW?

Microsoft Windows 10, iTunes®, Adobe Photoshop® and Mac OSX® are all examples of proprietary software.

Proprietary software is the opposite of open source – it is closed source. This means its source code is protected and users are not allowed to modify it.

On the plus side, proprietary software is extensively tested prior to release. This means any bugs that do appear are quickly fixed and there is plenty of user support. The disadvantage is that if the software doesn't exactly do what you want it to, you're not allowed to change it.

SKILLS CRITICAL THINKING

ACTIVITY 14

OPEN-SOURCE VERSUS PROPRIETARY SOFTWARE

Create a table that summarises the differences between open-source and proprietary software.

CYBERSECURITY

As you learnt in Unit 5, hacking represents a serious security threat. Many countries around the world including those in Europe and the GCC, as well as Australia and Singapore, have Computer Misuse Acts. These laws make it illegal for anyone to gain access to computer systems without permission, with higher penalties for deleting data or using information for further illegal purposes such as fraud and data theft.

SKILLS CRITICAL THINKING**SKILLS** REASONING**SKILLS** CRITICAL THINKING**SKILLS** CRITICAL THINKING**CHECKPOINT****Strengthen****S1** What is the purpose of a software licence?**S2** What are the main differences between open-source and proprietary software?**Challenge****C1** How would a patent protect you as an inventor of a product?**C2** List the relevant legislation that computer scientists should be familiar with.

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section to enhance your knowledge.

SUMMARY

- A programmer loses out financially if their software is downloaded illegally.
- The Copyright, Designs and Patents Act makes it illegal to copy, modify or distribute intellectual property without permission.
- Copyright protects the expression of an idea, not the idea itself.
- A patent offers more protection than copyright. It protects the idea of an invention rather than just a particular form of it.
- A software licence specifies how a piece of software can be used.
- Open-source software is freely available and can be edited and shared.
- The source code of proprietary software is protected, and users are not allowed to modify it.
- Computer Misuse Acts make it illegal to access computer systems without permission.

29 CURRENT AND EMERGING TRENDS

Computer science and digital devices are rapidly changing. In this section you will look at some of the new areas where major developments are being made.

LEARNING OBJECTIVES

- Be aware of current and emerging trends in computing technology, including:
 - artificial intelligence (AI)
 - DNA computing
 - nanotechnology
 - quantum computing

DID YOU KNOW?

It has been estimated that if the car industry had progressed as quickly as the computer industry over the last thirty years, a Rolls-Royce would today cost only \$40. It could circle the globe four times on 1 gallon of petrol with a top speed of 2.4 million miles per hour!

Computing technology changes rapidly in order to make computers more powerful and capable of processing even more commands each second. Over the last 20 years, the cost of computer components has fallen dramatically as their capabilities have improved.

ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) covers a range of ideas and technologies. It can mean different things to different people, for example:

- self-driving cars
- robots that impersonate humans
- machine learning.

A definition to cover all aspects could be:

The ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. Intelligent beings are those that can adapt to changing circumstances.

GENERAL VOCABULARY

cognitive connected with thinking or conscious mental processes

In other words, these are machines that solve **cognitive** problems commonly associated with human intelligence, such as learning, problem solving and pattern recognition.

The core challenges of artificial intelligence include programming computers for certain traits, such as:

- knowledge
- reasoning
- problem solving
- perception
- learning
- planning
- ability to manipulate and move objects.

In Units 1 and 2, we looked at algorithms and computer programs as a set of instructions for a computer to follow. We provide the intelligence and the computers follow instructions.

GENERAL VOCABULARY

machine learning the ability of computers to learn without being explicitly programmed

neuron a nerve cell



▲ Sophia, a humanoid robot. She uses voice recognition (speech-to-text) technology and is designed to learn over time. Her AI program analyses conversations and extracts data that allows it to improve responses in the future

Industrial robots do one job efficiently and well, such as paint spraying a car. They do the same thing over and over again and nothing else. They cannot be used to do different jobs as they are not 'intelligent'.

But what if computers could modify and write their own programs? If something didn't achieve the expected outcome, they could try something else. If something worked, they could remember it and use it in the future. They could learn new things without having their programs rewritten by a human. This is known as **machine learning**.

The main problem for scientists working on AI is that nobody fully understands how natural, human intelligence works. We know that the brain contains billions of **neurons** and that we think and learn by establishing electrical connections between them. But we do not know exactly how this results in higher reasoning.

ARTIFICIAL INTELLIGENCE IN USE TODAY

There are many uses in medicine such as:

- Digital consultations – apps to make a patient diagnosis based on the patient's medical history and knowledge. Users report their symptoms into the app, which uses speech recognition to compare against a database of illnesses. It then offers a recommended action, taking into account the user's medical history.
- Analysing test results – the computer is trained using millions of scans with their confirmed diagnosis. The computer can then spot medical markers far more accurately than a human.
- Health monitoring – wearable health trackers monitor heart rate and activity levels. They can send alerts to the user to get more exercise. It can share this information to doctors and AI systems for additional feedback.

There are also many uses in everyday life:

- Whenever Amazon® recommends a book to you, or Netflix® suggests a film or TV show, those recommendations are based on algorithms that examine what you've bought or watched in the past. The algorithms learn from those purchases, using them to suggest other things you might enjoy. Artificial intelligence lies behind those algorithms.
- Virtual assistants such as Siri® and Cortana® find the information and return the answers to your device, or they control other apps to achieve what you have commanded. Cortana can predict your needs by learning from information gathered and the way you use it.
- When flying an Airbus, the pilot only spends about 3 to 4 minutes manually flying the plane. The planes use a combination of motion sensors and GPS to track their position in flight. The software can even land commercial aircraft.
- Financial companies often use artificial intelligence to monitor transaction requests since algorithms process information so quickly. They spot patterns in your transactions and alert users to suspicious activity.

SKILLS

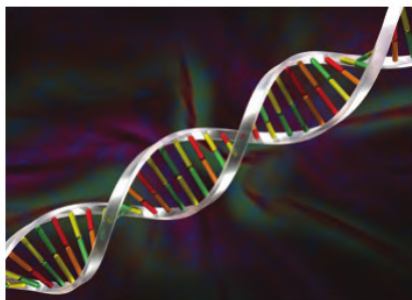
CRITICAL THINKING,
CREATIVITY

ACTIVITY 15

THE USES OF ARTIFICIAL INTELLIGENCE

Create a poster to illustrate the different uses of artificial intelligence.

DNA COMPUTING



SUBJECT VOCABULARY

DNA (deoxyribonucleic acid) is a self-replicating material which is present in nearly all living organisms as the main constituent of chromosomes. It is the carrier of genetic information

DID YOU KNOW?

Half a kilogram of DNA has the capacity to store more information than all the electronic computers ever built.

DNA (deoxyribonucleic acid) is the material that stores the genetic material of all organisms in their chromosomes. In every human cell, there are 46 chromosomes that contain all of the information to 'make' a person and allow them to function within their environment.

Millions of years ago, evolution created a perfect data system, capable of storing huge amounts of data. This has the ability to replicate itself but also has a mechanism to process the stored data to make the chemicals essential for life.

Like modern storage devices, DNA is digital, but it is not binary. Binary encoding uses two bits (0 and 1) but DNA uses four possible bits named adenine (A), thymine (T), guanine (G) and cytosine (C) after their chemical structure.

The equivalent to the byte (8 bits) in DNA is the 'codon' which consists of three of the digits. Because each digit can have four values (A, T, G, C) instead of two (0, 1). A DNA codon has 64 possible values compared to a binary byte which has 256.

Because of its microscopic size and data storage capabilities, DNA has become a research area for computer scientists. DNA computers would have many advantages over silicon-based ones, such as:

- there will always be a supply of DNA (as long as there are cellular organisms)
- the large supply of DNA makes it a cheap resource
- DNA biochips can be made cleanly, unlike the toxic materials used to make traditional microprocessors
- DNA computers are many times smaller than today's computers.

Its use for computation was first demonstrated in 1994 to solve the problem of finding the shortest route between seven cities, going through each city only once. Strands of DNA were made to represent the seven cities. The strands were mixed in a test tube and they combined together to represent all of the possible routes. Although all possible solutions were generated immediately, it took several days for the scientist to analyse the combinations to find the correct one. Another disadvantage to this DNA computer was that it also required human assistance.

Since this experiment, scientists have experienced barriers (both in theory and in practice). It may be the case that there will never be a computer made from DNA that rivals today's silicon-based microelectronics.

Research into using DNA as a data storage medium has been more successful. It is seen as the storage medium of the future. It has many advantages, such as size and the fact it can survive in a cool, dry environment for hundreds of thousands of years.

DNA computing is now seen as a field that focuses more on new combinations of biology and computer science, rather than competing with silicon.

Scientists are still working hard on ways to use the number-crunching abilities of DNA for specialised types of applications, such as code breaking. But beyond that, the intelligence built into DNA molecules could help create and build tiny, complex structures – using computer logic, not to crunch numbers, but to build new things.



▲ A scientist mixes DNA samples for the storage of digital files such as images, text and video

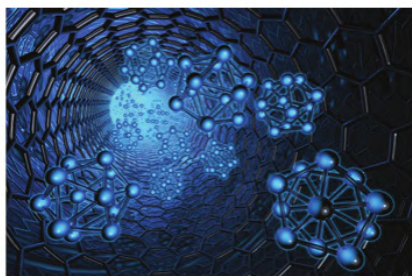
SKILLS CRITICAL THINKING

ACTIVITY 16

DNA CODE

How can a DNA code have 64 possible values?

NANOTECHNOLOGY



SUBJECT VOCABULARY

nanotechnology the manipulation of matter with a size of from 1 to 100 nanometres

nanometre (nm) one-millionth of a metre

GENERAL VOCABULARY

interdisciplinary involving two or more areas of knowledge

Nanotechnology is the manipulation of matter with a size of from 1 to 100 **nanometre (nm)**. One nanometre is 1 millionth (10^{-9}) of a metre. Therefore, nanotechnology is the manipulation of matter on an atomic and molecular scale.

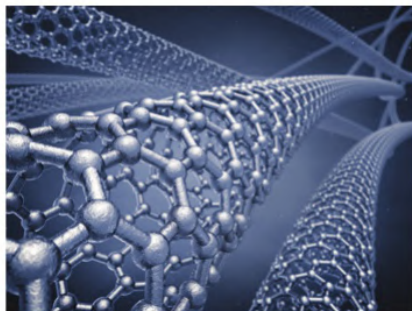
Nanotechnology is quickly becoming an **interdisciplinary** field as biologists, chemists, physicists and engineers are all involved in the study of substances at the nanoscale.

As mentioned in Unit 4, one way that scientists can make more powerful processors is by increasing the number of transistors that they contain. To do this, they have gradually reduced the size of transistors. For example, today Intel® are producing transistors that are 14 nm in width. An Intel dualcore® i7 processor contains 1.75 billion transistors with a transistor density of 17.18 transistors per square millimetre.

Further research is continuing, but below 7 nm it may not be technically possible to produce transistors using silicon. This is because of problems when the transistors are so close together. Therefore, scientists are experimenting with carbon nanotubes, which are cylinders of carbon atoms with walls one molecule thick. Processors using carbon nanotube transistors could be five times faster and use less energy than silicon chips.

DID YOU KNOW?

Nanotechnology is making an impact in the tennis world. The company Babolat® produces a tennis racket out of graphite infused with carbon nanotubes. The racket is very light, yet also many times stronger than steel.



▲ A carbon nanotube made by folding a sheet of carbon atoms

Nanotechnology is being used in a variety of areas:

- Self-cleaning glass – nanoparticles are included in the glass. UV (ultraviolet) radiation in light causes them to break down and loosen organic molecules, such as dirt, on the glass.
- Clothing – coating fabrics with zinc oxide nanoparticles gives them better protection from UV light.
- Scratch-resistant coatings – adding aluminium silicate nanoparticles to polymer coatings makes the coatings more effective. It increases resistance to chipping and scratches.
- Medicine – anti-cancer drugs that could harm normal tissue can be attached to nanoparticles. They can be delivered directly to the tumour using white blood cells.

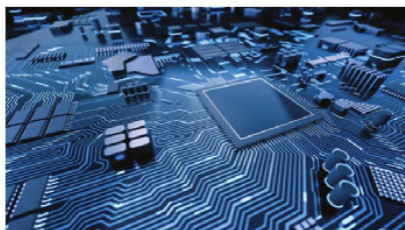
SKILLS CRITICAL THINKING, CREATIVITY

ACTIVITY 17

THE USES OF NANOTECHNOLOGY

Create a poster illustrating the uses of nanotechnology in the modern world.

QUANTUM COMPUTING



▲ Part of a quantum computer

GENERAL VOCABULARY

hypothetical an idea about something which is not necessarily real or true

cyanide an extremely toxic substance

SUBJECT VOCABULARY

quantum mechanics a theory that explains the nature and behaviour of matter and energy on the atomic and subatomic level

subatomic particle a unit of matter below the size of an atom, including the elementary particles such as quarks that all atoms are made up of

superposition the ability of a quantum system to be in multiple states at the same time until it is measured

qubit a quantum bit, the counterpart in quantum computing to the binary digit or bit of classical computing

superconducting circuits electrical circuits that have zero electrical resistance

One of the problems of using small atomic and sub-atomic sized materials is that the normal laws of classical physics don't work. This is when the rather 'strange' laws of quantum mechanics begin to take over.

Quantum computers are not better, faster, more super versions of the classical computers we know today. They are very different and are based on a completely different reality.

Quantum computers are based on **quantum mechanics**. This is the branch of physics which describes the behaviour of the very small, where **subatomic particles**, such as photons or quarks, can exist as both particles and waves.

One of the principles of quantum mechanics is the idea of **superposition**. This is the idea that an object can have all possible states, until its state has been measured.

DID YOU KNOW?

A scientist set up a thought experiment to illustrate superposition. It is known as 'Schrödinger's cat'. He imagined that a **hypothetical** living cat was placed in a thick lead box. At this stage, there is no question that the cat is alive. We then throw in some **cyanide** and close the box. We do not know if the cat is alive or if the cyanide capsule has broken and the cat has died. Since we do not know, the cat is both dead and alive, according to quantum law – it is in a 'superposition of states'. It is only when we break open the box and see what condition the cat is in that the superposition is lost, and the cat must be either alive or dead.

Similarly, when we toss a coin, we call one of two states: heads or tails. But when it is spinning in the air it has both states. It is both heads and tails until it is observed when it settles.



▲ A design showing that Schrödinger's cat is alive and dead at the same time

Today's classical computers use bits. As you have learnt in Unit 3, a bit can be set at either 1 or 0. Quantum computers use **qubits**. They are essentially subatomic particles, such as electrons or photons. To generate qubits, IBM and Google researchers use **superconducting circuits** cooled to temperatures colder than deep space.

Using microwave lasers, the qubits can be put into superposition – they exist in multiple states. So a qubit can be 1 and 0 at the same time! This allows quantum computers to carry out calculations and crunch through a vast number of potential outcomes simultaneously.

As a simple example, we could use four bits to try and crack a password which has 16 possible solutions. The four bits can be arranged in 16 possible combinations and each one can be tried at a time.

0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111

Each quantum bit is both 1 and 0 and so they can represent all the 16 possible solutions at the same time. The four qubits have all the wrong answers and the right answer at the same time. Instead of trying them one at a time, they are all tried at once.

This is a simple example. But to solve a complex encryption code, a classical computer could take millions of years looking at all of the possible combinations, while a quantum computer could consider the combinations in seconds.

SUBJECT VOCABULARY

entanglement co-dependence of the quantum states of pairs or groups of particles

Another principle of quantum mechanics can also be used in quantum computers. This is **entanglement**. This means that each of a pair of qubits exist in a single quantum state and they can influence each other. Changing the state of one of the qubits will immediately change the state of the other. This can happen over huge distances.

Nobody knows how this works. Einstein described it as 'a spooky action at a distance', but in a quantum computer, entangled qubits produces an exponential increase in its computing power.

The quantum state of the qubits is very fragile. The slightest vibration or change in temperature can make them change from a superposition state. This interruption is called 'noise' and causes lots of errors in calculations.

Quantum computing is still in its infancy but is being used today. For example:

- Volkswagen are using it to simulate the chemical composition of electrical vehicle batteries to improve their performance.
- Pharmaceutical companies are using it to analyse and compare chemical compounds that could lead to the creation of new drugs.
- Quantum computers can be used to create models to find **optimal conditions**. Airbus is using them to help calculate the most fuel-efficient ascent and descent paths for aircraft.
- Another possible use is in optimising traffic flow in a future city. A central quantum computer could imagine all possible solutions to traffic flow immediately, controlling traffic lights and sending driverless cars along different routes to stop **congestion**. But that is going to take quite a while.

GENERAL VOCABULARY

congestion blocked or crowded and causing difficulties

optimal conditions best or most favourable

SKILLS

CRITICAL THINKING,
CREATIVITY,
COMMUNICATION

ACTIVITY 18

THE ADVANTAGES OF QUANTUM COMPUTERS

In pairs, create a presentation explaining why quantum computers can solve complex problems more easily than classical ones.

SKILLS → CRITICAL THINKING

SKILLS → CRITICAL THINKING

SKILLS → CRITICAL THINKING

SKILLS → CRITICAL THINKING

SKILLS → REASONING

CHECKPOINT**Strengthen****S1** What is meant by artificial intelligence (AI)?**S2** Why is DNA suitable for storing data?**S3** What is meant by nanotechnology?**Challenge****C1** In quantum mechanics, what is meant by superposition and entanglement?**C2** How can quantum computers solve complex arithmetical problems far more rapidly than classical computers?

How confident do you feel about your answers to these questions? If you're not sure you answered them well, re-read this section to enhance your knowledge.

SUMMARY

- Artificial intelligence (AI) is the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings.
- Artificial intelligence has many applications, such as 'smart' robots, diagnostic systems and monitoring user behaviour and making suggestions.
- DNA is the material that stores the genetic information in all organisms.
- DNA can be used to store digital information.
- Nanotechnology is the manipulation of matter with a size from 1 to 100 nanometres (1 to 100 millionths of a metre).
- Nanotechnology can be used to produce transistors using carbon nanotubes.
- Quantum mechanics is the branch of physics that describes the behaviour of very small subatomic particles, which can exist as both particles and waves.
- Quantum computers use qubits, which can represent both 1 and 0 at the same time.

UNIT QUESTIONS

SKILLS CRITICAL THINKING A01

1 Discuss the impact of computing technology on the environment. (6)

SKILLS CRITICAL THINKING A02

2 Airtest produces exhaust emissions testing software. A programmer discovers that there is a bug in the software that produces inaccurate results under particular circumstances.

Discuss what course of action the programmer should take and explain why. (4)

HINT

Question 2 is scenario based, so make sure you relate your answer to the scenario – don't mention what you personally would do, but what the Airtest programmer should do. Don't forget to refer back to the code of conduct.

SKILLS CRITICAL THINKING A02

3 Assess the extent to which the patent system is a barrier to technological innovation. (4)

HINT

The command word in Question 3 is 'assess', which means you should consider both sides of the argument and come to a conclusion.

SKILLS CRITICAL THINKING A01

4 State two uses of artificial intelligence in the modern world. (2)

SKILLS CRITICAL THINKING A02

5 Discuss the legal and ethical implications of the following situations.

a Users changing their devices (such as smartphones, tablets and computers) many times. (3)

b Peer-to-peer file-sharing websites which allow users to distribute commercial audio, images and video to other users across networks including the Internet. (3)

c A student noticed that the teacher kept a copy of the password for past examination papers on a sticky note attached to his computer. The student used it to log in as the teacher to take a look. (3)

SKILLS CRITICAL THINKING A01

6 Some people have poorer access to computer science technology than others. List two reasons why this is the case. (2)

SKILLS CRITICAL THINKING, REASONING A01

7 Discuss the positive and negative effects of computer science technology on the environment. (6)

HINT

Remember that you will need an introduction and a conclusion for Question 7. Think about what you want to say and make a plan before you start writing your answer. You might find it useful to use headings to remind you of the different topics you need to write about. Your arguments should be given in a sensible, ordered sequence rather than being random.

SKILLS CRITICAL THINKING, DECISION MAKING A01

8 Some copyright holders grant Creative Commons licences to users of their works. Provide two reasons why a content creator might consider using a Creative Commons licence to make their work available to others. (2)

SKILLS CRITICAL THINKING A01 A02

9 TicToc manufactures smartwatches. It uses proprietary software to develop apps for the watch.

a What is meant by proprietary software? (1)

b Explain one advantage to TicToc of using proprietary software rather than open-source software to develop apps. (2)

SKILLS CRITICAL THINKING A02

10 State **two** advantages of DNA computing over silicon computers. (2)

PREPARING FOR YOUR INTERNATIONAL GCSE COMPUTER SCIENCE EXAMINATION (PAPER 1)

PAPER 1 PRINCIPLES OF COMPUTER SCIENCE

Time: 2 hours

QUESTION 1

SKILLS PROBLEM SOLVING

A02

- (a) (i) Add together the following two 8-bit binary numbers.

```

1 0 1 1 1 0 1 1
0 1 1 0 1 0 1 0

```

(2)

SKILLS CRITICAL THINKING

A01

- (ii) State the name of the error that has occurred when adding these two binary numbers.

..... (2)

SKILLS PROBLEM SOLVING

A02

- (b) Convert the binary number 10111011 to decimal, showing your working.

..... (2)

SKILLS PROBLEM SOLVING

A02

- (c) (i) Convert the binary number 11110111 to hexadecimal.

..... (2)

SKILLS REASONING

A01

- (ii) Explain why programmers prefer to use hexadecimal.

..... (2)

(Total for Question 1 = 10 marks)

QUESTION 2

SKILLS

CRITICAL THINKING

A01

Aliyah has just bought a new laptop. It was advertised as having a 2.6 GHz, quad-core processor with a 6-megabyte cache, 16 gigabytes of RAM and a 512 gigabyte solid-state drive.

- (a) The role of the CPU is to process program instructions using the fetch–execute cycle. Complete the table below to give the functions of the following components of the CPU in the fetch–execute cycle.

COMPONENT	FUNCTION
CU (Control Unit)	
Address bus	
Clock	
ALU (Arithmetic Logic Unit)	
Data bus	

(5)

SKILLS

CRITICAL THINKING

A01

- (b) In addition to RAM, the laptop's memory also contains ROM. Tick one box in each row of the table below to show whether each of the statements is true for ROM or RAM.

	RAM	ROM
Program instructions and data are stored here.		
It is used to boot up the laptop when it is switched on.		
It is volatile.		

(3)

SKILLS DECISION MAKING

A01

(c) Identify the type of secondary storage used by the laptop.

- ☐ A magnetic storage
☐ B optical storage
☐ C electrical storage
☐ D dynamic storage

(1)

SKILLS DECISION MAKING

A01

(d) Identify the number of processor cores in a quad-core processor.

- ☐ A 2
☐ B 16
☐ C 8
☐ D 4

(1)

(Total for Question 2 = 10 marks)

QUESTION 3

A team of programmers is developing an online game that can be played on different devices including computers, tablets and smartphones.

(a) The program is being written in a high-level language and then translated into machine code.

SKILLS REASONING

A01

(i) Describe the difference between high-level code and machine code.

.....

 (2)

SKILLS CRITICAL THINKING

A01

(ii) Explain **two** advantages to the team of using a high-level language to develop the game.

1

 2

 (4)

(b) One type of translator to convert high-level to machine code is an interpreter.

SKILLS CRITICAL THINKING

A01

(i) Describe how an interpreter converts high-level code to machine code.

.....

 (2)

SKILLS

CRITICAL THINKING

A01

- (ii) Identify two other types of translator used to convert source code into machine code.

1
2 (2)

(Total for Question 3 = 10 marks)

QUESTION 4

Jack thinks that his computer may have malware.

SKILLS

CRITICAL THINKING

A01

- (a) Explain what is meant by 'malware'.

.....
.....
.....
..... (2)

SKILLS

CRITICAL THINKING

A01

- (b) Describe **two** types of malware.

Type 1
.....
.....
.....
Type 2
.....
.....
..... (4)

SKILLS

REASONING

A02

- (c) State two precautions that Jack should take to prevent infection by malware.

Precaution 1
.....
Precaution 2
..... (2)

(Total for Question 4 = 8 marks)

QUESTION 5

A design company with 100 employees has decided to install a network to improve communications and productivity.

They have decided to use a client-server network with a star topology.

SKILLS

CRITICAL THINKING

A01

- (a) Draw and label a diagram to show how the computers and any required equipment could be connected using a star topology.

(3)

SKILLS REASONING

A01

- (b) Describe
- two**
- roles of a server in a client–server network.

Role 1

.....

.....

Role 2

.....

.....

(4)

SKILLS CRITICAL THINKING

A01

- (c) When computers communicate with each other on a network they must use the same protocol. Define what is meant by a 'protocol'.

.....

.....

.....

(2)

SKILLS REASONING

A01

- (d) The table below lists the names of the layers in which the protocols are arranged.

- (i) Number the empty cells to indicate the order in which data will pass when it is being transmitted by a computer. The first one has been done for you.

LAYER	ORDER
Transport layer	
Data link layer	
Application layer	
Network layer	

(3)

SKILLS REASONING

A01

- (ii) Describe two advantages to computer scientists of organising the protocols into layers.

Advantage 1.....

.....

.....

.....

Advantage 2.....

.....

.....

..... (4)

SKILLS REASONING

A02

- (e) Describe the security measures and network policies that should be used to safeguard the security and privacy of the company's data on the network.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... (6)

(Total for Question 5 = 22 marks)

QUESTION 6

Figure 1 is a bitmap image that Amira took with her camera during a visit to Istanbul and downloaded to her laptop.

► Figure 1



SKILLS CRITICAL THINKING

A01

- (a) The operating system gives the properties of the image as having a width of 5000 and a height of 4000 and a colour depth of 24.

(i) State the units in which the width and height are measured.

(1)

SKILLS CRITICAL THINKING

A01

(ii) Explain what is meant by colour depth.

(2)

SKILLS PROBLEM SOLVING

A02

(iii) Construct an expression to calculate the size of the image file in mebibytes.

(4)

SKILLS CRITICAL THINKING

A01

- (b) Lossy compression algorithms are often used to reduce the sizes of image files.

(i) Explain what is meant by 'lossy' compression.

(2)

SKILLS REASONING

A02

(ii) State why text files should not be compressed using 'lossy' compression algorithms.

(1)

(Total for Question 6 = 10 marks)

QUESTION 7

Wasim has designed an algorithm to convert any measurement made in centimetres into inches and vice versa.

```

1 SEND 'Which units are you entering/ enter "I" for inches
  and "C" for centimetres.' TO DISPLAY
2 RECEIVE choice FROM (STRING) KEYBOARD
3 SEND 'Please enter the measurement as a whole number' TO
  DISPLAY
4 RECEIVE valueToConvert (INTEGER) FROM KEYBOARD
5 IF choice = 'I' THEN
6     SET conversion TO valueToConvert * 0.39
7 ELSE
8     SET conversion TO valueToConvert * 2.54
9 END IF
10 SEND conversion TO DISPLAY

```

SKILLS

DECISION MAKING

A02

- (a) The variable 'valueToConvert' has been typecast as a **STRING**. Identify two other variables and state the data type of each.

1

Data type.....

2

Data type..... (4)

SKILLS

REASONING/
PROBLEM SOLVING

A02

- (b) When Wasim asks his friends to test the program it will carry out a calculation even if an entry other than 'I' or 'C' is made. Edit the algorithm by adding lines before line 2 to authenticate the user entry for the units they wish to convert.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... (6)

(Total for Question 7 = 10 marks)

(Total for Paper 1 = 80 marks)

PAPER 1: MARK SCHEME

- 1 (a) (i) 100100101 (2)
 (ii) overflow error (2)
 (b) $128 + 32 + 16 + 8 + 2 + 1 = 187$ (2)
 (c) (i) F7 (2)
 (ii) There are 4 bits for each hexadecimal digit and so it is easy to convert. There are fewer digits and so it is easier to remember the code for a particular colour. (2)

2 (a)

COMPONENT	FUNCTION
CU (Control Unit)	The CU controls the other components of the CPU and coordinates the fetch–execute cycle.
Address bus	The address bus carries memory addresses between the CPU and the memory to uniquely identify the memory address to be used.
Clock	The clock emits a pulse at regular intervals to synchronise the actions of other parts of the CPU.
ALU (Arithmetic Logic Unit)	The ALU performs arithmetic and logical operations to carry out program instructions.
Data bus	The data bus carries the item of data to be read from or written to memory to or from the CPU.

(5)

(b)

	RAM	ROM
Program instructions and data are stored here.	X	
It is used to boot up the laptop when it is switched on.		X
It is volatile.	X	

(3)

(c) C – electrical storage (1)

(d) D – 4 (1)

- 3 (a) (i) In high-level code, instructions use words, whereas in machine code instructions are in binary code. High-level code can be portable/translated for different machines whereas machine code is specific to a particular machine. High-level code is designed to be read by human programmers whereas machine code is to be read and executed by the computer. (2)
 (ii) The program can be written more quickly as the programmers are using a language more like their own natural language. There will be fewer errors in the code as the programmers are using a familiar language. The programmers will not have to write completely new code for each device the game is to be played on. (4)
 (b) (i) Translates one line of code at a time and executes it. It stops when it finds an error and can be resumed. (2)
 (ii) 1) assembler; 2) compiler (2)

4 (a) Malware is software that has been designed to gain unauthorised access to a computer system in order to disrupt its functioning or to collect information without the user's knowledge. (2)

(b) Any two of the following:

- A **virus** is a computer program hidden inside another program. Viruses can delete or corrupt data held on an infected computer. A virus can replicate itself and insert itself into other programs or files which can be passed on.
- A **worm** does not need another program to carry it. It can replicate and send itself in emails which are then sent to everyone in a user's address book. Worms consume computer resources as they are reproducing and they allow criminals to gain access to the infected computer and take it over.
- **Trojans** are installed by users when they think they are installing legitimate software. Trojans can delete files, change the desktop layout and send screenshots and key presses to the hacker's computer.
- **Spyware** often comes packaged with other software and the user does not know they are installing it. It spies on the user like a Trojan by sending information to a criminal.

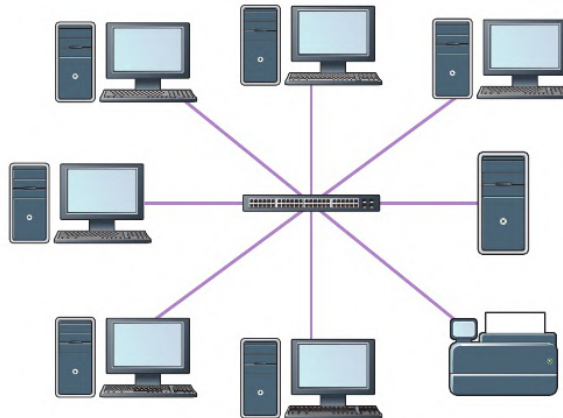
(4)

(c) Any two of the following:

- Install antivirus software and ensure that it is constantly updated.
- Ensure that the antivirus software can scan emails.
- Use adware removal software.
- Install anti-spyware software that removes or blocks spyware.
- Avoid opening emails and attachments from unknown sources.
- Ensure that the operating system is up to date.
- Install the latest security updates.

(2)

5 (a)



(3)

(b) Two of the following:

- The server is responsible for network security by allocating login names and passwords to users.
- The server can allocate the access rights of the users.
- The server stores all of the programs and files for the users.
- The server can centrally back up all data that is stored.

(4)

(c) A protocol is a set of rules for formatting data that is sent over a network so that it can be understood by the sending and receiving computers. (2)

(d) (i)

2
4
1
3

(3)

(ii) Two of the following:

- The overall model is simplified by dividing it into its functional parts.
- Different layers can be combined in different ways as required.
- One layer can be developed or changed without affecting the other layers.
- Allocating specific tasks makes it easier to identify and correct networking errors and problems.
- It provides a universal standard for hardware and software manufacturers to follow so that their devices will be able to communicate with each other.

(4)

(e) Answer should include the following points:

Physical security – electronic locks, CCTV, biometric authentication, swipe cards, etc.

Authentication and validation – login names and passwords. Rules regarding passwords, e.g. number and types of characters, when should be changed, no personal details, etc.

Access control – user's rights to folders and files, e.g. not able to view, read, edit, delete, etc.

Firewalls – software or hardware based. Controls incoming and outgoing data, stops certain protocols, blocks IP addresses.

Encryption – data on network should be encrypted so that even if intercepted cannot be understood without relevant key.

Network policies – listing what users can and cannot do e.g. attach mobile devices, install software, copy data, etc.

(6)

6 (a) (i) pixels

(1)

(ii) The colour depth states the number of bits used to represent the colour value of each pixel. The more bits that are used, the greater number of colours that can be represented.

(2)

(iii) File size = $(5000 \times 4000 \times 24) / 8 / 1024 / 1024$ (Give 1 mark for each element.)

(4)

(b) (i) The file size is reduced by permanently removing some of the information, so when the file is decompressed it is not the same as the original.

(2)

(ii) With text files, letters and words would be permanently deleted and the text would not make sense.

(1)

7 (a) 1) choice – string

2) conversion – real

(4)

(b) This is one possible solution.

```
SET acceptable TO False
WHILE acceptable = False DO
    SEND 'Which units are you entering/ enter "I"
for inches and "C" for centimetres.' TO DISPLAY
    RECEIVE choice FROM (STRING) KEYBOARD
    IF choice = 'True' OR choice = 'C' THEN
        SET acceptable TO true
    ELSE
        SEND 'Sorry that is not recognised.' TO DISPLAY
    END IF
END WHILE
```

(6)

PREPARING FOR YOUR INTERNATIONAL GCSE COMPUTER SCIENCE EXAMINATION (PAPER 2)

PAPER 2 APPLICATION OF COMPUTATIONAL THINKING

Time: 3 hours

QUESTION 1

SKILLS**DECISION MAKING****A03**

- (a) Open the file **Q01a**.
Answer these questions about the code.
- (i) Identify the name of a variable. (1)
.....
- (ii) Identify the line number(s) showing repetition. (1)
.....
- (iii) Identify the name given to a data structure in this file. (1)
.....
- (iv) Identify the line number(s) containing three arithmetical operators. (1)
.....

SKILLS**REASONING****A02**

- (b) One way to make code more readable and understandable is to add comments.
Add suitable comments to lines 4, 10, 11 and 13. (4)
Save your amended code as **Q01a + your name** (e.g. **Q01aYousefBoutros**) with the correct file extension.

SKILLS**PROBLEM SOLVING****A03**

- (c) Open the file **Q01b**.
In this code there are **two** program errors.
Amend the code to correct the errors. (2)
Save your amended code as **Q01b + your name**.

(Total for Question 1 = 10 marks)

QUESTION 2

SKILLS

CRITICAL THINKING

A01

In a program, when data is entered by a user it should be validated.

- (a) Explain what is meant by data validation.

(2)

.....

.....

.....

.....

SKILLS

COMPUTATIONAL
THINKING

A03

- (b) Write a program to calculate the total of ten numbers entered by a user.

Each number should be between 1 and 10.

If the number is not in this range, the user should be asked to re-enter the number.

The program should then print the total of the ten numbers entered.

You must use the structure given in file **Q02a** to write the program.

Save your amended code as **Q02a + your name** with the correct file extension.

(10)

(Total for Question 2 = 12 marks)

QUESTION 3

A publishing company has created a program to analyse sentences to count the number of vowels used.

Open the file **Q03a**.

SKILLS

REASONING

A02

- (a) Identify the name of a **user-defined** subprogram.

(1)

.....

SKILLS

REASONING

A02

- (b) Identify the name of a **built-in** subprogram in the code.

(1)

.....

SKILLS

REASONING

A02

- (c) Identify the name of an **input parameter** in the code.

(1)

.....

SKILLS

REASONING

A02

- (d) Identify the name of a **global variable**.

(1)

.....

SKILLS

REASONING

A02

- (e) a, e, i, o and u are local variables. Identify the name of one other **local variable**.

(1)

.....

SKILLS

REASONING

A02

- (f) Explain why all the characters in the sentence are changed to lower case before they are compared.

(2)

(Total for Question 3 = 7 marks)

QUESTION 4

SKILLS

PROBLEM SOLVING/
COMPUTATIONAL
THINKING

A03

A building company requires a program to calculate the cost of producing concrete flooring for clients.

The program should ask a user to input the length and width of the area in metres, e.g. 3.9.

The cost per square metre is \$50.

The program should use a subprogram to calculate the cost and then print this out when it is returned.

Save your amended code as **Q04a + your name**.

(Total for Question 4 = 7 marks)

QUESTION 5

SKILLS

PROBLEM SOLVING/
COMPUTATIONAL
THINKING

A03

Anil wants to save his computer science scores in a text file named ComputerScience.txt.

- (a) Write a program that will allow him to save 10 computer science scores in the file.

Save your program as **Q05a + your name**.

- (b) Anil has decided to improve the program so that he is not restricted to just 10 entries. Amend the code so that he can add scores until he makes a blank entry.

Save your program as **Q05b + your name**.

- (c) Write a program that will allow Anil to retrieve his scores from the text file and store them in a suitable data structure.

Save your program as **Q05c + your name**.

(Total for Question 5 = 12 marks)

QUESTION 6

SKILLS

REASONING

A02

Open file **Q06a**.

Complete the trace table to show the execution of the code.

You may not need to fill in all the rows of the table.

X	Y	Z	OUTPUT

(Total for Question 6 = 5 marks)

QUESTION 7

SKILLS

PROBLEM SOLVING/
COMPUTATIONAL
THINKING

A03

A teacher is storing the examination results in a two-dimensional array.

He would like to calculate and print out the percentage of students who achieve a certain result or higher.

Open file **Q07a**.

This shows the array and the final line of the program.

Complete the code to calculate the mean result.

Save your program as **Q07a + your name**.

(Total for Question 7 = 7 marks)

QUESTION 8

SKILLS

PROBLEM SOLVING/
COMPUTATIONAL
THINKING

A03

Open file **Q08a**.

In the file, a series of random numbers are stored in a data structure.

These numbers need to be sorted into ascending order.

Write a program to sort them using a bubble sort.

Save your completed program as **Q08a + your name**.

(Total for Question 8 = 20 marks)

(Total for Paper 2 = 80 marks)

PAPER 2: MARK SCHEME

- 1 (a) (i) Any of `input1`, `input2`, `solution` (1)
 (ii) Line 9 (1)
 (iii) `results` (1)
 (iv) 6 and 10 (1)

(b) **Line 4:** `input1` is multiplied by `input2` and the product is divided by the sum of `input1 + input 2`.

Line 10: `input1` is multiplied by `input2` and the product is divided by the sum of `input1 + input2`.

Line 11: the data stored in the variable '`solution`' is added to the array, `results`.

Line 13: `input1` is now incremented by 1. (4)

(c) `Validated = False` added as Line 1 before the while loop.

Line 6 (`print`) has to be indented. (2)

(Total for question 1 = 10 marks)

2 (a) Automatically checking by the program that the data is reasonable and is not outside set boundaries. (2)

(b) Loop set up for entering ten numbers only, e.g.:

`for i in range(1, 11):` (1)

Loop set up to validate input, e.g.:

`validated = False`

`while validated == False:` (2)

User asked to input a number in a specified range, e.g.:

`number = int(input("Please enter a number between 1 and 10."))` (1)

If within range the number is accepted, e.g.:

`if number > 0 and number <= 10:` (2)

`validated = True`

If not in range, the user is asked to re-enter, e.g.:

`print("Please enter a number between 1 and 10.")` (1)

The total is calculated, e.g.:

`total = total + number` (1)

The total is printed, e.g.:

`print("The total is" + str(total))` (1)

Program is completely functional. (1)

(Total for question 2 = 12 marks)

3 (a) `countVowels` (1)

(b) `userSentence[index].lower()` or `print()` or `len()` (1)

(c) `userSentence` (1)

(d) `sentence`, `numberA` to `numberE` (1)

(e) `userSentence` (1)

(f) Because `a` and `A` are different characters and, in the program, the `As` would not be counted, as they are being compared to '`a`'. (2)

(Total for question 3 = 7 marks)

4 1 mark for each of the following:

- function created
- with two parameters
- cost is calculated by correct formula in the function
- the cost is returned to the main program
- in main program, length and width are input
- the function is called with two arguments
- the cost is printed using currency sign and two decimal places.

(Total for question 4 = 7 marks)

5 (a) Text file opened with correct name and 'write' switch.

Loop for 10 entries.

Entries written to the file.

The text file is closed.

(4)

(b) WHILE/DO loop created.

Loop is run until a blank entry is made.

Entries are written to file.

Entry number is incremented by one.

(4)

(c) Data structure created to receive scores.

File opened with 'read' switch.

Scores are read and added to data structure.

File is closed.

(4)

(Total for question 5 = 12 marks)

6 1 mark for each row:

X	Y	Z	OUTPUT
2	3	5	
3	6	5	
4	10	5	
5	15	5	
5	15	5	15

(Total for question 6 = 5 marks)

7 User input to enter target score.

(1)

Loop to go through the list complete list.

(1)

Comparison of each result with the target.

(2)

Number incremented if is higher than the target.

(1)

Calculation of the percentage.

(1)

Printout of percentage with explanatory text.

(1)

(Total for question 7 = 7 marks)

8 Award 1 mark for each of the following points up to a maximum of 11 marks:

- Use function to find the length of the list (1)
- Use Boolean variable to indicate a swap has been made (1)
- Use a variable to count the number of passes (1)
- Create a condition-controlled loop to terminate when there have been no swaps (1)
- Set up a nested count-controlled loop to move through the numbers in the list (1)
- Use **IF** statements to compare each number with the next one (1)
- Check if the numbers need to be swapped (1)
- If a change is needed, then a routine to switch them (1)
- Boolean variable is updated to indicate that a change has been made (1)
- The count-controlled loop successfully moves through the list of numbers (1)
- The condition-controlled loop successfully terminates when the condition is met (1)
- The number of each pass is given (1)
- The list is successfully sorted into ascending order (1)

Award up to a maximum of 9 marks using the levels-based mark scheme below.

BAND 0	BAND 1 (1–3 MARKS)	BAND 2 (4–6 MARKS)	BAND 3 (7–9 MARKS)	MARK
No rewardable content	Little attempt to decompose the problem into component parts	Some attempt to decompose the problem into component parts	The problem has been decomposed into component parts	(9)
	Some parts of the logic are clear and appropriate to the problem	Most parts of the logic are clear and mostly appropriate to the problem	The logic is clear and appropriate to the problem	
	Some appropriate use and manipulation of data types, variables, data structures and program constructs	The use and manipulation of data types, variables and data structures and program constructs is mostly appropriate	The use and manipulation of data types, variables and data structures and program constructs is appropriate	
	Parts of the code are clear and readable	Code is mostly clear and readable	Code is clear and readable	
	Finished program will not be flexible enough with other data sets or input	Finished program will function with some but not all other data sets or input	Finished program could be used with other data sets or input	
	The program meets some of the given requirements	The program meets most of the given requirements	The program fully meets the given requirements	

APPENDIX 1: COMMAND WORDS

The table below lists the command words that may be used in question papers for the Pearson Edexcel International GCSE in Computer Science. It also gives an outline of what each type of question requires students to do.

COMMAND WORD	DEFINITION
Amend	Requires changes/additions/deletions/rearrangement of a symbolic representation.
Analyse	Examine elements in detail.
Assess	Give careful consideration to all the factors or events that apply and identify which are the most important or relevant. Make a judgement on the importance of something and come to a conclusion where needed.
Calculate	Obtain a numerical answer, showing relevant working. If the answer has a unit, this must be included.
Compare and/or contrast	Look for the similarities and/or differences of two (or more) things. This should not require the drawing of a conclusion. The answer must relate to both (or all) things mentioned in the question.
Complete	Requires the completion of a table/diagram/algorithm (in any notation).
Construct	Usually requires creation of an artefact using subject-specific symbolic representation, rules and syntax.
Convert	Requires changing information in one symbolic representation to another representation. May require amending to provide new functionality/facility.
Create	Usually requires creation of an artefact using subject-specific symbolic representations.
Deduce	Draw/reach conclusion(s) from the information provided.
Describe	Give an account of something. Statements in the response need to be developed, as they are often linked, but do not need to include a justification or reason.
Devise	Plan or invent a procedure from existing principles/ideas.
Discuss	Identify the issue/situation/problem/argument that is being assessed in the question. Explore all aspects of an issue/situation/problem/argument. Investigate the issue/situation, etc. by reasoning or argument.

COMMAND WORD	DEFINITION
Draw	Produce a diagram/image, either using a ruler or freehand. May require labelling/annotation to express meaning. Used when symbolic representations need to be manipulated.
Evaluate	Review information, then bring it together to form a conclusion, drawing on evidence, including strengths, weaknesses, alternative actions, relevant data or information. Come to a supported judgement of a subject's qualities and relation to its context.
Explain	An explanation requires a justification/exemplification of a point. The answer must contain some element of reasoning/justification. This can include mathematical/logical explanations. The mark scheme must have marking points that are linked. The answer MUST contain some element of reasoning/justification to satisfy the definition of 'explain'. The mark scheme should be clearly laid out so that to gain full marks, there must be a minimum of 1 mark for some element of reasoning/justification to satisfy the definition of 'explain'.
Give	Recall one or more pieces of information.
Give a reason/reasons	When a statement has been made and the requirement is only to give the reasons why.
Identify	Usually requires some key information to be selected from a given stimulus/resource.
Name	Recall one piece of information from a given diagram/algorithm/process.
Perform	Carry out an identified task/algorithm/process.
Predict	Give an expected result.
State	Recall one or more pieces of technical information.
State and justify/Identify and justify	When a selection is made and then a justification has to be given for the selection.
State what is meant by	When the meaning of a term is expected but there are different ways to describe them. Usually 1 or 2 marks maximum.
Write	Usually requires manipulation/creation/amending of an artefact using subject specific symbolic representations.

Examples of how these command words are used in Paper 1 and Paper 2 are given in the Pearson Edexcel International GCSE in Computer Science Sample Assessment Materials for this qualification.

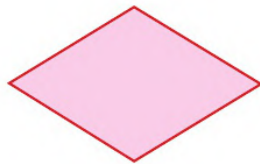
APPENDIX 2: FLOWCHART SYMBOLS



Denotes the start and end of an algorithm



Denotes a process to be carried out



Denotes a decision to be made



Shows the logical flow of the program



Denotes a subprocess that may be defined in a separate flowchart



Denotes input and/or output

APPENDIX 3: PSEUDOCODE COMMAND SET

Questions in the written examination that involve code will use this pseudocode for clarity and consistency. However, students may answer questions using any valid method.

DATA TYPES

INTEGER

REAL

BOOLEAN

CHARACTER

TYPE COERCION

	INTEGER	REAL
INTEGER	INTEGER	REAL
REAL	REAL	REAL

Type coercion is automatic if indicated by context. For example, $3 + 8.25 = 11.25$

(integer + real = real)

Mixed mode arithmetic is coerced like this:

Coercion can be made explicit. For example, `RECEIVE age FROM (INTEGER) KEYBOARD` assumes that the input from the keyboard is interpreted as an INTEGER, not a STRING.

CONSTANTS

The value of constants can only ever be set once. They are identified by the keyword `CONST`. Two examples of using a constant are shown.

```
CONST REAL PI
SET PI TO 3.14159
SET circumference TO radius * PI * 2
```

DATA STRUCTURES

ARRAY

STRING

Indices start at zero (0) for all data structures.

All data structures have an append operator, indicated by `&`.

Using `&` with a STRING and a non-STRING will coerce to STRING. For example, `SEND 'Fred' & age TO DISPLAY`, will display a single STRING of 'Fred18'.

IDENTIFIERS

Identifiers are sequences of letters, digits and underscores (`_`), starting with a letter, for example:

```
MyValue, myValue, My_Value, Counter2
```

FUNCTIONS

LENGTH() For data structures consisting of an array or string.

RANDOM(n) This generates a random number from 0 to n.

COMMENTS

Comments are indicated by the # symbol, followed by any text.

A comment can be on a line by itself or at the end of a line.

DEVICES

Use of **KEYBOARD** and **DISPLAY** are suitable for input and output.

Additional devices may be required, but their function will be obvious from the context. For example, **CARD_READER** and **MOTOR** are two such devices.

NOTES

In the following pseudocode, the < > indicates where expressions or values need to be supplied. The < > symbols are not part of the pseudocode.

VARIABLES AND ARRAYS		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
SET Variable TO <value>	Assigns a value to a variable.	SET Counter TO 0 SET MyString TO 'Hello world'
SET Variable TO <expression>	Computes the value of an expression and assigns to a variable.	SET Sum TO Score + 10 SET Size to LENGTH(Word)
SET Array[index] TO <value>	Assigns a value to an element of a one-dimensional array.	SET ArrayClass[1] TO 'Ann' SET ArrayMarks[3] TO 56
SET Array TO [<value>, ...]	Initialises a one-dimensional array with a set of values.	SET ArrayValues TO [1, 2, 3, 4, 5]
SET Array [RowIndex, ColumnIndex] TO <value>	Assigns a value to an element of a two-dimensional array.	SET ArrayClassMarks[2,4] TO 92

SELECTION		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
IF <expression> THEN <command> END IF	If <expression> is true then command is executed.	IF Answer = 10 THEN SET Score TO Score + 1 END IF
IF <expression> THEN <command> ELSE <command> END IF	If <expression> is true then first <command> is executed, otherwise second <command> is executed.	IF Answer = 'correct' THEN SEND 'Well done' TO DISPLAY ELSE SEND 'Try again' TO DISPLAY END IF

REPETITION		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
<pre>WHILE <condition> DO <command> END WHILE</pre>	Pre-conditioned loop. Executes <command> while <condition> is true.	<pre>WHILE Flag = 0 DO SEND 'All well' TO DISPLAY END WHILE</pre>
<pre>REPEAT <command> UNTIL <expression></pre>	Post-conditioned loop. Executes <command> until <condition> is true. The loop must execute at least once.	<pre>REPEAT SET Go TO Go + 1 UNTIL Go = 10</pre>
<pre>REPEAT <expression> TIMES <command> END REPEAT</pre>	Count controlled loop. The number of times <command> is executed is determined by the expression.	<pre>REPEAT 100-Number TIMES SEND '*' TO DISPLAY END REPEAT</pre>
<pre>FOR <id> FROM <expression> TO <expression> DO <command> END FOR</pre>	Count controlled loop. Executes <command> a fixed number of times.	<pre>FOR Index FROM 1 TO 10 DO SEND ArrayNumbers[Index] TO DISPLAY END FOR</pre>
<pre>FOR <id> FROM <expression> TO <expression> STEP <expression> DO <command> END FOR</pre>	Count controlled loop using a step.	<pre>FOR Index FROM 1 TO 500 STEP 25 DO SEND Index TO DISPLAY END FOR</pre>
<pre>FOR EACH <id> FROM <expression> DO <command> END FOR EACH</pre>	Count controlled loop. Executes for each element of an array.	<pre>SET WordsArray TO ['The', 'Sky', 'is', 'grey'] SET Sentence to '' FOR EACH Word FROM WordsArray DO SET Sentence TO Sentence & Word & ' ' END FOR EACH</pre>

INPUT/OUTPUT		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
SEND <expression> TO DISPLAY	Sends output to the screen.	SEND 'Have a good day.' TO DISPLAY
RECEIVE <identifier> FROM (type) <device>	Reads input of specified type.	RECEIVE Name FROM (STRING) KEYBOARD RECEIVE LengthOfJourney FROM (INTEGER) CARD_READER RECEIVE YesNo FROM (CHARACTER) CARD_READER

FILE/HANDLING		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
READ <File> <record>	Reads in a record from a <file> and assigns to a <variable>. Each READ statement reads a record from the file.	READ MyFile.doc Record
WRITE <File> <record>	Writes a record to a file. Each WRITE statement writes a record to the file.	WRITE MyFile.doc Answer1, Answer2, 'xyz 01'

SUBPROGRAMS		
SYNTAX	EXPLANATION OF SYNTAX	EXAMPLE
PROCEDURE <id> (<parameter>, ...) BEGIN PROCEDURE <command> END PROCEDURE	Defines a procedure.	PROCEDURE CalculateAverage (Mark1, Mark2, Mark3) BEGIN PROCEDURE SET Avg to (Mark1 + Mark2 + Mark3)/3 END PROCEDURE
FUNCTION <id> (<parameter>, ...) BEGIN FUNCTION <command> RETURN <expression> END FUNCTION	Defines a function.	FUNCTION AddMarks (Mark1, Mark2, Mark3) BEGIN FUNCTION SET Total to (Mark1 + Mark2 + Mark3)/3 RETURN Total END FUNCTION
<id> (<parameter>, ...)	Calls a procedure or a function.	Add (FirstMark, SecondMark)

ARITHMETIC OPERATORS

SYMBOL	DESCRIPTION
+	Add
-	Subtract
/	Divide
*	Multiply
^	Exponent
MOD	Modulo
DIV	Integer division

RELATIONAL OPERATORS

SYMBOL	DESCRIPTION
=	equal to
<>	not equal to
>	greater than
> =	greater than or equal to
<	less than
< =	less than or equal to

LOGICAL OPERATORS

SYMBOL	DESCRIPTION
AND	Returns true if both conditions are true.
OR	Returns true if any of the conditions are true.
NOT	Reverses the outcome of the expression; true becomes false, false becomes true.

GLOSSARY

abstraction the process of removing or hiding unnecessary detail so that only the important points remain

access control this decides which users have access to which data, and what they are allowed to do with it

agent a computer system that can interpret its environment (through its sensors). It can act autonomously upon that environment (through its effectors)

algorithm a precise method for solving a problem. All instructions given to a computer must be unambiguous or it won't know what to do

analogue using signals or information represented by a quantity (e.g. an electric voltage or current) that is continuously variable. Changes in the information being represented are indicated by changes in voltage. This method requires very accurate sending and receiving systems

application programming interface (API) code that allows two programs to communicate with each other

application software software that performs a task that would otherwise be done by hand, perhaps with pen and paper

arithmetic/logic unit (ALU) the part of the CPU that performs calculations and logic operations

arithmetic operator an operator that performs a calculation on two numbers

arithmetic shift used for signed binary numbers. When performing a right shift, the bits at the left are replaced by copies of the most significant bit

array an organised collection of related values that share a single identifier

ascending order this is arranging items from smallest to largest (e.g. 1, 2, 3, 4, 5, 6 or a, b, c, d, e, f)

assembler translates the mnemonics of assembly language programs into machine language instructions for the microprocessor to carry out

assembly language a low-level language written using mnemonics

assignment statement the SET...TO command is used to initialise variables in pseudocode

audit trail a record of activities that have taken place on a computer system. This record is generated automatically and will record what has happened and who or what made the change

authentication the process of proving to a computer system who you are (e.g. using a username and password)

autonomous self-directed and acts without requiring any help or guidance. Can choose its own goal and uses its experience to achieve it

back-up a copy of files in another location so that they are still available if the original copy is damaged or lost. Backing up is the process of making a back-up copy

bandwidth the amount of data that can be carried on a network in a given period of time

binary information represented by only two values (e.g. voltage or no voltage; on or off). There are no communication errors or misunderstandings because there are no small differences

binary digit (bits) the smallest unit of data that is represented in a computer. It has a single binary value, either 1 or 0

binary prefix multiplies a unit by powers of 2

binary shift an operation done on all the bits of a binary value in which they are moved by a specific number of places to either the left or right

bit depth the number of bits used to encode the information from each sample. Increasing the number of bits used increases the amount of detail contained in each sample

block of code a grouping of two or more code statements

Bluetooth™ a protocol for the short-range wireless interconnection of mobile phones, computers and other electronic devices

Boolean something that can take only the values True or False; named after English mathematician George Boole

brute force an algorithm design that does not include any techniques to improve performance, but instead relies on computing power to try all possibilities until the solution to a problem is found

bug an error or flaw in a computer program

built-in functions functions that are provided in most high-level programming languages to perform common tasks

bus a group of connections between devices in a computer

bus width the number of wires that make up a bus. This determines the range of binary numbers that can be communicated

byte the basic combination of bits used to represent an item of information. A byte typically consists of 8 bits

cache a temporary data store so that the data can be accessed very quickly when needed

cache memory memory used to make up for the difference in speed between two internal components

cache miss when the data requested for processing by a component or application is not found in the cache memory

carbon footprint the amount of carbon dioxide an individual or organisation produces as a result of the energy they consume

central processing unit (CPU) hardware device that carries out the processing in a computer

character set the defined list of characters recognised by a computer's hardware and software

checksum a technique for finding errors. A mathematical formula is applied to the data and the resulting number value is transmitted with the data. The recipient computer applies the same formula to the received data. It then compares the checksum sent with the data to the calculated checksum. If the checksums don't match, the data is likely to have been corrupted. So, the recipient computer requests the data again

ciphertext the encrypted plaintext

client-server network a network that has at least one server to provide services to the client computers

clock an electronic device inside a CPU that 'ticks' at regular intervals and is used to synchronise the actions of the other parts of the CPU

cloud storage secondary storage, often belonging to a third party, that is accessed via a network, usually the internet, and so is not in the same physical place as the machine's RAM/ROM. Files stored 'in the cloud' can be accessed from anywhere via an Internet connection

code vulnerability a computer program (the code) that has been written in such a way that it creates a security issue that may be taken advantage of to gain access to the computer system or data within it

colour depth the number of bits used to encode the colour of each pixel

communication media the means by which data is transmitted between devices on a network. Coaxial cable, fibre-optic cable and microwaves are all forms of communication media

compiler a translator that converts high-level language source code into object code, often machine code. The source code is translated all at once and saved to be executed later

compression changing the format of a data file so that the size of the file becomes smaller

computational thinking the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by a computer

computing technology an all-encompassing term referring to the hardware, software and infrastructure that underpin current and emerging computer systems

concatenation the linking together of two or more items of information

concurrent processes that run at the same time are described as being concurrent

constant a 'container' that holds a value that never changes. Like variables, constants have unique identifiers

construct a smaller part from which something is built. Letters and numbers (i.e. A to Z and 0 to 9) are the constructs we use to build our language and convey meaning. Bricks and cement are the basic constructs of a building

control unit the part of the CPU that organises the actions of the other parts of the CPU

Creative Commons an organisation that allows people to set copyright terms for their intellectual property. One use of a Creative Commons licence is to allow people to copy material as long as it is not used commercially

cyber attack any kind of malicious attack on a network-connected device

data structure an organised collection of related elements. Arrays and records are two common data structures used in programming

data type specifies what kind of data it can hold. Common data types are integer, real, Boolean and character. The data type of a value determines the operations that can be performed upon it

debugger a computer program that assists in the detection and correction of errors in other computer programs

decimal prefix multiplies a unit by powers of 10

decomposition breaking a problem down into smaller, more manageable parts, which are then easier to solve

definite iteration this is used when the number of iterations, or 'turns of the loop', is known in advance. It can be set to as many turns as you want. This sort of loop is said to be 'count controlled'

defragmenter a utility that moves file clusters on a disk so they are closer to each other in order to speed up disk access

Denial of Service (DoS) an attack on a network that attempts to prevent legitimate users from accessing its services

descending order this is arranging items from largest to smallest (e.g. 6, 5, 4, 3, 2, 1 or f, e, d, c, b, a)

developer a person whose job it is to create new software

digital information represented by certain fixed values (e.g. high, medium or low). Any signal between these values would be meaningless and not used. Sending and receiving systems do not have to be as accurate as for analogue communication

digital divide the gap between people who are technology-empowered and those who are technology-excluded

digital inclusion ensuring that everyone has affordable access to computing technology and the necessary skills to take advantage of it

disk thrashing a very high rate of hard disk access

divide and conquer an algorithm design that works by dividing a problem into smaller and smaller sub-problems, until they are easy to solve. The solutions to these are then combined to give a solution to the complete problem

DNA deoxyribonucleic acid. It is a self-replicating material which is present in nearly all living organisms as the main constituent of chromosomes. It is the carrier of genetic information

domain name this is part of the URL for a resource on the Internet. When the domain name is used, it will be converted to the correct IP address by the domain name service (DNS) and the contact will take place

dual-core processor a processor with two processor cores

eavesdrop having unauthorised sight of data being sent from one computer to another over a network

effectors something that produces an effect by carrying out an action

emulator hardware or software that allows one type of computer system to behave like another

encryption the process of turning information (e.g. a message) into a form that only the intended recipient can decode, or decrypt and read. The message is encoded using an agreed method or algorithm. This is called the key. The encrypted message is called a cipher

entanglement co-dependence of the quantum states of pairs or groups of particles

ethics a set of moral principles that govern a person's behaviour

e-waste any form of discarded electronic equipment, including computing technology

execution the process by which a computer carries out the instructions of a computer program

fetch–decode–execute cycle sequence of steps carried out repeatedly by a CPU

fidelity the relationship between the original sound and the recording. The greater the fidelity, the greater the similarity between them. Very high accuracy is called hi-fidelity or just hi-fi

field an individual element in a record

firewall a utility that controls program access to the network, both incoming and outgoing

flowchart shows an algorithm as a diagram. Each step in the algorithm is represented by a symbol. Symbols are linked together with arrows showing the order in which steps are completed

function a function is a subprogram that performs a specific task and can be used at any point in the program. High-level programming languages have a number of useful built-in functions. You can also create your own or use functions available in online libraries

function machine a metaphor or diagram that represents a machine that takes an input. It applies a rule such as a set of operations and delivers the answer as an output

gigahertz (GHz) a measure of frequency equivalent to 1000 million cycles per second

global variable a variable that can be accessed from anywhere in the program, including inside subprograms

hacking the act of gaining unauthorised access to a computer system and the data it contains

heuristic a type of algorithm capable of finding a solution to a problem quickly and easily. This is done through trial and error and educated guesswork to cut corners and remove less likely alternatives. Heuristic algorithms don't always find the best solution, but they will usually find one that works

hexadecimal a base-16 number system. There are 16 digits and the place values increase in powers of 16

high-level programming language a programming language that is similar to natural human language

HyperText Markup Language (HTML) essentially a text document that contains any text to be displayed along with:

- details of how the text should be formatted (e.g. font size, colour, etc.)
- details of any hyperlinks and where they link to using a URL

- details of any objects such as pictures or videos that should be shown within the web page when it is displayed

identifier a unique name given to a variable or a constant. Using descriptive names for variables makes code much easier to read

identity theft the stealing of someone's personal details, such as their bank account number, sort code or passport number, for the purpose of making purchases and running up debts in their name

IF...THEN...ELSE statement the IF...THEN...ELSE statement allows a choice to be made between two alternatives based on whether or not a condition is met (e.g. IF it is cold THEN wear a jumper ELSE wear a T-shirt)

indefinite iteration this is used when the number of iterations is not known before the loop is started. The iterations stop when a specified condition is met. This sort of loop is said to be 'condition controlled'

infinite loop a loop that is never-ending since the condition required to terminate the loop is never reached

initialisation the process of assigning an initial value to a variable

initialise to set variables to their starting values at the beginning of a program or subprogram

input to enter data into a computer

instruction set the list of all possible commands a particular CPU knows how to carry out

integer a whole number (e.g. 3, 6, 9)

Integrated Development Environment (IDE) a package that helps programmers to develop program code. It has a number of useful tools, including a source code editor and a debugger. One of the most useful features of an IDE is the debugger. One of its tasks is to flag up syntax errors in the code and issue helpful error messages

intellectual property (IP) a creation of the human mind that is unique and has a commercial value

Internet a worldwide system of interconnected networks that enables information to be exchanged and shared

Internet of things (IoT) the interconnection of digital devices embedded in everyday objects

Internet Service Provider (ISP) an organisation that provides its customers with a connection to the Internet

interpreter a translator that converts high-level language source code into object code, often machine code. The source code is translated and executed one line at a time

iteration a construct that means a process is repeated. An action is repeated until a condition is met or a particular outcome is reached. It is often referred to as a 'loop'

keystream the characters that are combined with plaintext to produce an encrypted message

keyword the text that is chosen to generate the keystream

latency refers to any kind of delay that data travelling through a network might encounter

local area network (LAN) a network that covers a relatively small geographical area, often a single site

local variable a variable that is accessed only from within the subprogram in which it is created

location-based services services that enable people to access and share real-time location information online

logic the system that is used to get a computer to perform a task; the principles and reasoning underlying the constructs and elements to be applied in solving problems

logic circuit an electronic circuit that has inputs and outputs that follow one of the Boolean operators

logic error an error in an algorithm that results in incorrect or unexpected behaviour

logical operator a Boolean operator using AND, OR and NOT

lossless compression compressing a file in such a way that it can be decompressed without any loss of data

lossy compression compression where some of the data is removed; the original file cannot be restored when the lossy file is decompressed

low-level programming language a programming language that is closely related to the CPU's machine code

machine code the binary codes representing each of the instructions in the instruction set

magnetic storage secondary storage that works by making parts of a substance behave like a magnet, with north and south poles to represent binary 1s and 0s

main memory/random-access memory (RAM) a temporary store for data and instructions (programs)

malware short for 'malicious software'. It is used as a generic term for any kind of software that is designed to disrupt the use of a computer system

median the middle number when the numbers are put in ascending or descending order (e.g. if there are 13 numbers, then the 7th number is the median). If there are an even number of items in an ascending list, choose the item to the right of the middle (e.g. if there are 10 numbers, then choose the 6th as the median). In a binary search, the higher of the two numbers would be chosen

memory address a number that uniquely identifies a (memory) storage location

microprocessor the central unit that executes and manages the instructions passed to it

mnemonic a short, simple acronym that represents each of the instructions in a CPU's instruction set, e.g. LDR (load register), STR (store) and CMP (compare)

modem short for 'modulator–demodulator'. It modulates and demodulates signals (i.e. converts them from digital to analogue and vice versa) sent from and received by a computer over a communications network

modular testing testing each block of code as it is completed to ensure the code works as expected

Monte Carlo method carrying out a statistical analysis of a number of random samples to get approximate solutions to a problem. The larger the number of samples used, the more accurate the result is likely to be

most significant bit (MSB) the bit with the highest value in a multiple-bit binary number

multicore processor a processor with many processor cores

nanometre one-millionth of a metre

nanotechnology the manipulation of matter with a size of 1 to 100 nanometres

nested IF statement a nested IF statement consists of one or more IF statements placed inside each other. A nested IF is used where there are more than two possible courses of action

nested loop a loop that runs inside another loop. The inner one executes all of its instructions for each turn of the outer loop

network an arrangement of computers and other devices connected together to share resources and data

network security activities designed to protect a network and its data from threats such as viruses, hacker attacks, denial of service attacks, data interception and theft, and equipment failure

network service an app running on a server which provides facilities or operations such as data storage, printing or communications

network topology describes how the devices on a network are connected together

neural networks processing information in a similar way to human brains and learning and adapting over time. This makes them useful tools for recognising faces, identifying illnesses and quality control, for example. Computers are normally not very good at these things

nibble 4 bits or half a byte

non-volatile memory that is not lost when the power is turned off

object code the translated source code. Often this will be machine code, but might also be an intermediate code, which has to be further translated before it can be executed

OOP language an object-oriented programming language. Instead of data structures and separate program structures, both data and program elements are combined into one structure called an object

open-source software software that is free to edit and redistribute

operating system (OS) software designed for particular hardware and which manages other programs' access to the hardware

operator precedence the order in which you apply the operators (including logical operators) in a mathematical equation

optical storage secondary storage that works using differences in light reflection from a material

output to display or output data that has been processed (or has been stored)

overflow error this condition occurs when a calculation produces a result that is greater than the computer can deal with or store. When this happens, the microprocessor is informed that an error has occurred

packet a small quantity of data being sent through a network. The packet is labelled with the sender's address (source), the recipient's address (destination), how many packets are being transmitted and the position of this packet in the complete message

paging the algorithm the OS uses to move programs from RAM to disk and back again when needed, once main memory is full

parameters values that are passed into a subprogram when it is called

patent an exclusive right granted to an inventor to make, use and sell an invention for a fixed period of time

peer-to-peer network a network that doesn't have any dedicated servers. Each computer in the network can act as client and server

personal area network (PAN) network used for data transmission over short distances by computer devices, such as laptops, mobile phones, tablets, media players, speakers and printers

personal data information that is personal and unique to an individual

petabyte 1 petabyte is 1000 terabytes or 1 million gigabytes

pharming directing Internet users to a fake website that looks like a real one, in order to obtain personal information such as passwords, account numbers, etc.

phishing attempting to get confidential information by sending a user a message that appears to come from a trustworthy person or organisation

physical security controlling access to critical parts of a network using physical methods (such as locked doors) rather than software

pixel short for 'picture element', the smallest single point of colour in a graphic image

plaintext the text that is to be encrypted

platters rigid, rapidly rotating, magnetically coated disks within a hard disk drive

print spooling documents are loaded into a buffer (usually an area on a disk), and then the printer pulls them off the buffer at its own rate

privacy the right to be left alone and free from unwanted scrutiny and intrusion

procedure a subprogram containing a set of statements that are executed when the procedure is called into use by a command in the main program. Unlike a function, a procedure does not return a value to the main program

process to change the meaning or format of some data

proprietary software software that belongs to an individual or a company. Its licence specifies that users are not allowed to modify the source code and places restrictions on its use

protocol a set of rules that govern how communications on a network should be formatted and what data they should include

pseudocode a structured, code-like language that can be used to describe an algorithm

quantum mechanics a theory that explains the nature and behaviour of matter and energy on the atomic and subatomic level

qubit a quantum bit, the counterpart in quantum computing to the binary digit or bit of classical computing

random number a number within a given range of numbers that is generated in such a way that each number in the range has an equal chance of occurring

reading when the CPU retrieves the data stored at a given address

read-only memory (ROM) memory that cannot be altered and is not lost when the power is turned off

reboot to start a device again, e.g. turn off the computer and turn it on again

record a data structure that stores a set of related values of different data types

recursion a process that is repeated. For example, a document can be checked and edited, checked and edited and so on until it is perfect

register a storage location inside the CPU used to hold an instruction, an address or other single item of data

relational operator an operator that compares two values

resolution the number of pixels per inch when the image is displayed (for example, on a monitor or on paper)

router a piece of networking hardware used to forward packets of data from one network to another

runtime error an error that occurs while the program is running – the operation the computer is asked to do is impossible to execute

sampling taking measurements of the sound wave at regular but different intervals of time (e.g. 44 100 samples per second)

scheduling the algorithm that the OS uses to allow each running process to use the CPU

scope the region of code within which a variable is visible

secondary storage any kind of permanent storage to which the contents of ROM/RAM are copied (usually a hard disk, optical or solid-state device)

selection a construct that allows a choice to be made between different alternatives

sequence an ordered set of instructions

shoulder surfing gaining access to information by watching someone enter it into a computer system

simulation a representation of a real-world process or system

single-core processor a processor with only one processor core

social engineering any kind of attack on a computer system or network that takes advantage of how people behave and respond to certain situations

software the set of programs run by a computer system

solid-state storage secondary storage that works by storing charge (electrons)

source code the text of the program that a programmer writes

spyware software, possibly a virus, that is designed to be installed secretly on a computer. It records private information as the user enters it

static RAM (SRAM) memory that retains data bits in its memory as long as power is being supplied and does not have to be refreshed

string a sequence of characters. They can be letters, numbers, symbols, punctuation marks or spaces

string traversal using a loop to cycle through each character in a string

subatomic particle unit of matter below the size of an atom, including the elementary particles such as quarks that all atoms are made up of.

subprogram a self-contained module of code that performs a specific task. It can be 'called' by the main program when it is needed

superconducting circuits electrical circuits that have zero electrical resistance

superposition the ability of a quantum system to be in multiple states at the same time until it is measured

surveillance technology CCTV, drones, number plate recognition, bugging and tracking devices used to monitor and record people's activities, often without their knowledge

switch a device that receives incoming data packets and redirects them to their destination on a local area network (LAN)

synchronised occurring at the same time or operating at the same rate

syntax error an error that occurs when a rule of the programming language is broken

system software operating system and utility software

text file a sequence of lines, each of which consists of a sequence of characters

throughput the amount of data to pass through a system

trace table a technique used to identify any logic errors in algorithms. Each column represents a variable or output and each row a value of that variable

translator a program that converts source code into machine code

traversal travel across or through something. An array can be traversed by moving from the first to the last element in order to examine the data stored at each index position

truth table a table showing all possible combinations of the inputs and outputs of an operator

two-dimensional array a matrix of rows and columns resembling a table. Two indices are used, one to reference the rows and the other the columns. All the elements of a two-dimensional array share the same data type

two-factor authentication a security check where users have to type in the code from a portable hardware device called a 'secure token' or from an SMS message sent to their mobile phone

type coercion the process of converting the value stored in a variable from one data type to another

unambiguous this means that the instructions cannot be misunderstood. Simply saying 'turn' would be ambiguous (i.e. unclear) because you could turn left or right

unpatched software software that hasn't had the latest security updates applied to it, making it vulnerable to attack

USB (Universal Serial Bus) sockets found on most modern computer systems

user interface the way the user interacts with the operating system

utility software software that does a useful job for the user that is not essential to the operating system and not the reason for using a computer in the first place

validation to check that the data entered by a user or from a file meets specified requirements

variable a 'container' used to store data. The data stored in a variable is referred to as a 'value'. The value stored in a variable is not fixed. The same variable can store different values during the course of a program and each time a program is run

virtualisation any process that hides the true, physical nature of a computing resource, making it look different, usually to simplify the way it is accessed

virus software that is designed to make and distribute copies of itself. This is usually for a malicious purpose

volatile memory that is erased when the power is turned off

von Neumann architecture computer system design in which the program is stored in memory with the data

web server powerful computer system that stores web pages and any multimedia that the pages might contain

'What if?' question running a computer model with a given set of inputs to see what the model produces as an output or prediction

whistle-blower someone who draws attention to the activities of an organisation or person believed to be acting illegally or unethically

wide area network (WAN) a network that covers a large geographical area. It connects together two or more LANs and is usually under shared ownership. The largest wide area network is the Internet

wireless access point (WAP) a device that is used to connect one or more computers wirelessly to an existing physical wired network

wireless local area network (WLAN) a local area network in which connected devices use high-frequency radio waves to communicate

writing when the CPU sends data to memory to be stored at a given address

INDEX

- abstraction 23, 24–6, 27–8, 184
- access control 223
- agents 163
- algorithms 3–22, 29–31, 104
 - coding 26–7, 34, 140
 - computational models 162–3
 - constructs 4, 12–14
 - displaying 5–8, 10
 - efficiency 5, 19, 21–2
 - for calculations 8, 92
 - for encryption 146–9
 - for games 13, 26–7, 104
 - for passwords 10, 163, 164
 - for run-length encoding 140–1
 - for searching 15, 19–22, 65
 - for sorting 15–19
 - heuristic 192
 - input-process-output 161–2
 - logic errors 92–9
 - problem solving process 23–8
 - success of 5
 - test plan 102
 - see also pseudocode
- analogue recording 132
- analogue systems 110
- AND operator 42, 181–3, 184, 185, 298
- application programming interface (API) 188
- application software 187–8, 193, 229–30
- arguments 84–7
- arithmetic operators 8–9, 298
- arithmetic shifts 121–2
- arithmetic/logic unit (ALU) 171
- arrays 62–8, 71–2, 295
 - length 62
 - and text files 74, 75–80
 - treating strings as 56
- artificial intelligence (AI) 264–5
- ascending order 15
- ASCII code 125–9, 146, 148
- assemblers 194
- assembly language 194–5
- assignment statement 36, 62
- asymmetric encryption 144
- audio files 132–4, 138, 142–3
- audit trail 230
- authentication 189, 204–5, 222–3

- Babbage, Charles 166
- back-ups 190
- basic tools 190
- BIDMAS 92
- big data 251–2
- binary 110–24
 - addition 114–17, 118–19
 - denary conversion 112–14, 117–18
 - hexadecimal conversion 123–4
 - multiplication/division 120–2
 - representing text 125–9
 - signed integers 116–19, 121–2
 - two's complement 117–19, 121–2
- binary prefix 136–7
- binary search 20–2
- binary shifts 120–2
- BIOS (Basic Input/Output System) 168
- bit depth 133–4
- bit pattern 130
- bitmap images 129–31, 139–40, 142
- bits (binary digits) 111–12
- blocks of code 51, 52, 75–6
- Blu-Ray™ disks 175–6
- Bluetooth™ 203, 211
- Boolean operators 41, 42, 181–5, 298
- bottlenecks 169, 173–4
- boundary data 72, 103
- browsers see web browsers
- brute force algorithms 19
- bubble sort 15–17, 19
- buffers 169
- bugs 183
- bus networks 205–7
- buses 166, 168, 171
- bytes 112

- C# programming language
 - arrays 62, 64, 66, 67, 76–7, 80
 - casting (type coercion) 37
 - comments 52
 - concatenation 60
 - declaring variables 35, 36
 - file handling 76–7, 79–80
 - loop constructs 45, 47, 48, 56
 - menus 89–90
 - operators 41–2
 - random numbers 49, 83
 - selection 40–1
 - strings 55, 56–60
 - subprograms 83, 84, 85–6, 87
 - testing for errors 96–9, 100
- cache 173–4
- cache memory 168–9, 174
- cache miss 169
- Caesar cipher 145–9
- carbon footprint 246, 248
- 'carry over' 114–16
- casting 37
- CDs 175–6
- central processing unit (CPU) 166–7, 188–9
 - caches 168–9, 173–4
 - components 170–2
 - instruction set 194–5
 - performance 169, 172–5
 - reading/writing data 167–9
- character set 125
- characters 35, 54–61, 125–9
- checksum 214
- ciphers 145–54
- ciphertext 151
- client-server networks 204–5
- clock/clock speed 171, 172
- cloud storage 177–8
 - energy consumption 246
 - security 225–6, 227
- code readability 51–3, 90
- code vulnerability 229–30, 231
- colour depth 130–1
- comments 51–3, 295
- commercial analysis tools 231
- communication 201, 240–1
 - see also Internet; networks
- communication media 210–11
- compilers 196
- compression 138–43, 191
- computational models 162–3
- computational thinking 23, 27–8
 - abstraction 23, 24–6, 27–8
 - algorithm design 12–22
 - decomposition 23–4, 27–8
 - exam practice 283–90
- Computer Misuse Acts 262
- computer scientists 258–9
- computer systems 159–99
 - components 166–70
 - computational models 162–3
 - embedded systems 160–1, 178–9
 - fetch-decode-execute cycle 166, 167, 170–2
 - input-process-output model 160–2
 - logic 181–6
 - machines 160–1
 - performance 169, 172–5
 - programming languages 194–7
 - von Neumann architecture 165–7
 - see also hardware; networks; software
- computing technology
 - benefits 248, 250, 251, 252, 255, 256
 - current and emerging trends 264–70
 - digital inclusion 255–7
 - disposal 247
 - energy consumption 246
 - environmental impacts 244–9
 - environmental monitoring 248
 - legal impacts 260–3
 - manufacturing 244–6
 - privacy implications 250–4
- concatenation 59–60
- confidentiality 221–2

- connectivity 211, 256
- constants 9, 294
- control unit 171
- cookies 251
- copyright 260–1
- cores 173, 174
- CPU *see* central processing unit
- Creative Commons 262
- CSMA/CD protocol 206
- CSV (comma separated value) files 75, 76–7
- cyber attacks 227–9
- cybersecurity 229–32, 262
- data 109–57
 - compression 138–43
 - encryption 144–54, 222, 226
 - graphics 110–11, 129–31, 139–40, 142
 - personal 250–4
 - reading/writing text files 73–80
 - representation 110–11, 125–35
 - searching 15, 19–22
 - security 191, 221–7
 - sorting 15–19
 - sound 131–4, 142–3
 - storage 136–8, 175–8, 225–7, 266
 - test data 72–3, 103
 - text 110–11, 125–9, 138–41
 - transmission 212–18, 234–5
 - validation 70–3
 - see also* binary
- data structures 62–9, 294
 - arrays 62–8
 - multidimensional arrays 65–8
 - records 68–9
 - static/dynamic 62–3
 - see also* strings
- data types 35–7, 50
 - in arrays 63, 64, 67
 - in lists 63
 - in records 68–9
- debugger 90, 101
- decimal prefix 136–7
- decimal system *see* denary system
- decomposition 23–4, 27–8
- decryption 152, 153–4
- definite loops 42–5
- defragmenter 191
- denary system 110, 111–15
 - binary conversion 112–14, 117–18
- denial of service (DoS) 222
- descending order 15
- digital inclusion/divide 255–7
- digital recording 132–4
- digital systems 110, 111
- disk thrashing 170
- divide and conquer algorithms 19, 20
- DNA computing 266–7
- domain name 228, 233–4
- domain name service (DNS) 228, 233–4, 236
- drivers 189
- dual-core processors 173
- DVDs 175–6
- eavesdropping 211, 229
- electrons 176
- email protocols 213, 218
- emails 204–5, 227, 253
- embedded systems 160–1, 178–9
- emulators 195
- encryption 127, 128, 144–54, 222, 226
 - asymmetric/symmetric 144
 - Caesar cipher 145–9
 - keys 144, 152
 - pigpen cipher 145
 - rail fence cipher 152–4
 - Vigenère cipher 149–52
- energy use 246, 248
- environmental impacts 244–9
- environmental monitoring 248
- erroneous data 73, 103
- errors 90, 92–101
 - error messages 101, 216
 - logic 92–9, 100, 183
 - overflow 116
 - runtime 75, 99–100
 - syntax 99, 100, 101
 - types of 100
- ethernet 213–14
- ethics 250, 259, 260
- evaluating programs 104
- e-waste 244, 247
- exam practice 272–9, 283–7
 - algorithms 29–31
 - command words 291–2
 - computational thinking 283–90
 - computer systems 198–9
 - computing technology 271
 - data 155–7
 - mark schemes 280–2, 288–90
 - networks and Internet 240–1
 - programming 106
- execution 34
- fetch–decode–execute cycle 166, 167, 170–2
- fidelity 133–4
- fields 68–9
- file class 76
- file compression 138–43, 191
- file handle 73
- file management tools 190–1
- file reading/writing to 73–80
- file sharing 205, 216, 218
- file size 131, 134, 136–8
- firewalls 191, 223–4
- firmware 168
- flash drives 176, 226
- flowcharts 6–7, 8, 10, 14
 - bubble sort 17
 - Caesar cipher 147
- input–process–output 162
- logic errors 93
- nested IF statements 39
- RLE algorithm 140–1
- selection and iteration 12–13
- symbols 6, 82, 162, 293
- FOR loops 43, 55–6
- FTP (file transfer protocol) 216, 218
- function machines 161, 163
- functions 55, 82–3, 84–7, 295
 - built-in 55, 91
 - encryption algorithms 148–9
- games 172, 195
 - algorithms 13, 26–7, 104
 - creating 24, 25–7, 50, 104
 - game over 181–3
 - highest scores 68, 77
- Garbage In, Garbage Out (GIGO) 70
- global variables 83
- glossary 299–305
- GPS 248, 256, 264
- graphics 110–11, 129–31, 195
 - compression 139–40, 142
- hacking 224, 231, 262
- hard disks 170, 175
- hardware 165–80, 188–9
 - cache memory 168–9, 174
 - input/output devices 166, 167
 - main memory (RAM) 166–70, 173–4, 188–9
 - networking 234, 237–9
 - operating system and 170, 188–9
 - protocols 213
 - read-only memory (ROM) 168
 - secondary storage 175–8, 225–7
 - virtual memory 170
 - see also* central processing unit
- heuristic algorithms 192
- hexadecimal numbers 122–4
- high-level programming language 26–7, 34–5, 196
 - see also specific languages*
- HTML (HyperText Markup Language) 236
- HTTP (HyperText Transfer Protocol) 216, 218
- HTTPS (HyperText Transfer Protocol Secure) 144, 216, 218
- hyperlinks 235, 236
- identifiers 8, 9, 51–2, 294
- identity theft 251
- IF statements 38–9, 70–1, 181–3, 295
- IF...THEN...ELSE statement 38–42, 184–5, 295
- images 110–11, 129–31
 - compression 139–40, 142
- indefinite loops 42, 47–8
- indexes 54, 56–7, 62, 65–6, 67
- infinite loop 93–4

- initialisation 35–6
- input 70–81, 161–2, 297
- input devices 166, 167, 295
- input–process–output model 160–2
- instruction set 194–5
- integers 35, 49, 63–4, 67, 294
 - signed/unsigned 116–22
- integrated development environment (IDE) 101
- intellectual property (IP) 260–2
- Internet 210, 212, 233–41
 - components for accessing 237–9
 - connecting to 233–5
 - IP addresses 216–17, 233–4, 236–7
 - peer-to-peer networks 205
- Internet protocol (IP) 216–17, 234
- Internet service provider (ISP) 234, 235
- Internet of things (IoT) 179, 211
- interpreters 196
- IP addresses 216–17, 233–4, 236–7
- iteration 12–14, 42–8
 - see *also* loops
- Java
 - arrays/lists 62, 63–4, 66–7, 75–6, 78–9
 - comments 52
 - concatenation 60
 - declaring variables 35, 36
 - file handling 74–6, 78–9
 - IDE 101
 - loop constructs 44, 46, 48, 56
 - menus 88–9
 - operators 41–2
 - random numbers 49, 83
 - selection 40
 - strings 55, 56–60
 - subprograms 83, 84, 85, 86, 87
 - testing for errors 95–6, 97–8, 100, 101
 - type coercion 37
- JPEG files 142
- keys/keywords 144–9, 150, 152–3
- keystreams 150
- latency 175, 212, 219
- layered protocol stacks 214, 215, 216–19
- legal issues 260–3
- length check 55, 72
- length functions 55, 62
- licensing 261–2
- linear search 19–20, 21–2, 65
- lists 63, 64, 78–9
- local area network (LAN) 203–4, 213–14
- local variables 83
- location-based services 253
- logic circuits 183
- logic errors 92–9, 100, 183
- logic statements 181–5
- logical operators 41, 42, 181–5, 298
- logical shifts 120–1
- look-up check 71–2
- loops 12–14, 42–8, 296
 - array traversal 62
 - infinite 93–4
 - string traversal 55–6
- lossless compression 138–41
- lossy compression 142–3
- Lovelace, Ada 166
- low-level programming language 194–5
- machine code 194–5, 196
- machine learning 192, 265
- magnetic storage 175
- main memory 166–70, 173–4, 188–9
- malware 191, 224, 228, 229, 253
- manufacturing 244–6
- median 20
- memory address 167–8
- menus 87–90
- merge sort 18–19
- mesh network 209–10
- microprocessors 167–9
- mnemonics 195
- mobile phone networks 211, 219, 256
- modelling 25, 28, 191–2
- modems 238–9
- modular testing 230
- Monte Carlo methods 192
- most significant bit (MSB) 116, 117
- MP3 files 142–3
- multi-agent computational models 163
- multidimensional arrays 65–8
- multitasking 189
- naming conventions 9, 10
- nanotechnology 267
- negative numbers 116–19, 121–2
- nested IF statement 38–9
- nested loops 46–7
- network attached storage (NAS) 177–8, 226
- network security 191, 221–32, 262
 - access control 223
 - authentication/validation 222–3
 - cloud storage 178, 225–6, 227
 - cyber attacks 227–32
 - firewalls 223–4
 - identifying vulnerabilities 231–2
 - importance of 222
 - physical security 224
 - protecting against weaknesses 229–30
- networks 202–20, 240–1
 - connectivity 210–11
 - data speed 212
 - hardware 234, 237–9
 - layers 218–19
 - mobile communications 211, 219, 256
 - policies 231–2
 - protocols 211, 213–18
 - services 202–3
 - storage systems 177–8, 225–6, 227
 - topologies 205–10
 - types of 203–5
 - see *also* Internet; network security
- neural networks 192
- nibble 136
- normal data 72, 103
- NOT operator 42, 182–3, 185, 298
- number systems 111–15, 122–4
 - see *also* binary
- object code 196
- one-dimensional arrays 62–4, 69, 71–2
 - and text files 74, 75–7
- open-source software 262
- operating systems 187–90, 193, 230
- operator precedence 185
- operators 8–9, 41–2, 181–5, 298
- optical storage 175–6
- OR operator 42, 182–3, 185, 298
- output 70–81, 161–2, 297
- output devices 166, 167, 295
- overflow errors 116
- overwriting files 73
- packets 215, 234–5
- paging 188
- parallel computational models 162
- parameters 84–7
- passwords 55, 88–90, 226
 - algorithms 10, 163, 164
 - authentication 189, 204–5, 222–3
- patents 260–1
- peer-to-peer networks 205
- pen (penetration) testing 231
- personal area network (PAN) 203
- personal data 250–4
- pharming 227, 228
- phishing 227
- physical security 224
- pigpen cipher 145
- piracy/illegal downloads 205, 260
- pixels 129–31, 142
- plaintext 150
- platters 175
- presence check 71
- print spooling 169
- privacy 250–4
- problem solving 4–31
 - abstraction 23, 24–6
 - creating algorithms 4–14
 - decomposition 23–4
 - process 23–8
 - searching/sorting data 15–22
- procedures 83–4, 87–90
- processes 188–9
- processing 161, 162–3, 166
- processors 110, 162–3, 267
 - multicore 173, 174, 175
 - see *also* central processing unit
- professionalism 258–9
- programming 34–106
 - assembly language 194–5

- code readability 51–3, 90
- code vulnerability 229–30
- data structures 62–9
- data types 35–7
- input/output 70–81
- iteration constructs 42–8
- logic errors 92–9, 100, 183
- logic statements 181–5
- operators 41–2, 181–6
- random numbers 49–50, 82–3
- runtime errors 99–100
- selection constructs 38–42
- strings 54–61
- structured 148–9
- subprograms 82–91, 148–9
- syntax errors 99, 100, 101
- testing/evaluation 92–105, 229–30
- programming languages 26–7, 34–5, 194–7
 - see also specific languages*
- programs 5, 34, 82, 88–90
 - evaluating 104
 - test plan 101–4
 - see also programming; software*
- proprietary software 262
- protocols 211, 213–18, 234
- pseudocode 7–10, 294–8
 - arrays 62, 295
 - characters and ASCII 127–8
 - data validation 70–2
 - encryption algorithms 149
 - file handling 297
 - functions 82, 127–8, 297
 - input/output 297
 - linear search algorithm 65
 - logic errors 92, 93–5
 - loops 43, 55, 70–2, 296
 - operators 8–9, 181–5, 298
 - procedures 84, 297
 - selection 38–9, 181–5, 295
 - strings 55, 59, 127
 - subprograms 297
 - variables 8, 9, 35–6, 295
- Python 262
 - arrays/lists 62–3, 66, 74, 77–8
 - characters and ASCII 127
 - comments 52
 - concatenation 59–60
 - declaring variables 35, 36
 - file handling 73–4, 77–8
 - IDE 101
 - loop constructs 43–4, 46, 47, 55–6
 - menus 88
 - operators 41–2
 - random numbers 49, 83
 - selection 40
 - strings 55–60
 - subprograms 83, 84, 85, 86, 88
 - testing for errors 95–6, 97–8, 99, 101
 - type coercion 37
- quad-core processors 175
- quantum computing 176, 268–9
- qubits 268–9
- rail fence cipher 152–4
- RAM *see* random-access memory
- random numbers 49–50, 82–3
- random-access memory (RAM) 166–70, 173–4, 188–9
- range check 70–1, 93
- ransomware 253
- read-only memory (ROM) 168
- readability of code 51–3
- real data type 35, 37, 49, 294
- RECEIVE statement 35, 38
- records 68–9
- recursion 18, 20, 30
- registers 171
- relational operators 41, 298
- REPEAT loops 43, 296
- resolution 129, 130
- ring network 207–8
- ROM *see* read-only memory (ROM)
- routers 234–5, 238
- run-length encoding (RLE) 138–41
- runtime errors 75, 99–100
- sampling 132–4
- scheduling 188, 190
- scope 83, 86
- search algorithms 15, 19–22, 65
- secondary storage 175–8, 225–7
- security tools 191
 - see also encryption; network security*
- selection 12–14, 38–42, 181–5, 295
- semiconductors 245
- sequence 4, 5–8, 12, 38
- sequential computational models 162
- servers 202, 204–5, 224
 - DNS servers 233–4
- SET...TO command 36, 62
- shoulder surfing 227, 228
- simulations 28, 191–2
- single-core processors 173
- slicing 58, 61
- social engineering 227–8
- software 187–93
 - applications 187–8, 193, 229–30
 - copyright 260, 261
 - licensing 261–2
 - open-source/proprietary 262
 - operating systems 187–90, 193, 230
 - security weaknesses 228, 229–30, 231
 - simulation and modelling 191–2
 - updating 230, 232
 - utilities 187, 190–1, 231
- solid-state storage 175, 176, 226
- sort algorithms 15–19
- sound data 131–4, 142–3
- source code 194, 196, 260, 262
- spyware 191, 251
- star network 208–9
- static arrays 62, 63
- static RAM (SRAM) 174
- storage 136–8
 - secondary 175–8
 - security 225–7
 - using DNA 266
- storage area network (SAN) 177–8
- stored program computers 165–7, 194
- strings 54–61, 138–40
 - analysing 127
 - comparing 60
 - concatenation 59–60
 - index 54, 56–7
 - length 55
 - repeated 138
 - substrings 58–9
 - traversal 55–6, 140
 - upper/lower case 57–8
- structured programming 148–9
- subprograms 27, 82–91, 148–9
 - arguments and parameters 84–7
 - benefits 90
 - functions 55, 82–3, 86, 87
 - and menus 87–90
 - procedures 83–4, 87–90
 - testing 90, 103
 - variables 83, 84–7
- substrings 58–9
- surveillance technology 252
- switches 237
- symmetric encryption 144
- syntax errors 99, 100, 101
- system administrator 223
- system software 187–91
- TCP (transmission control protocol) 214
- TCP/IP protocol stack 214–15, 216–19, 234
- technical weaknesses 228–9
- test data 72–3, 103
- testing and evaluation 92–105
 - code vulnerabilities 229–30, 231
 - evaluating a program 104
 - finding and fixing errors 92–101
 - IDE features 101
 - test plan 101–4
 - trace tables 94–9
 - types of error 100
- text data 110–11, 125–9, 138–41
- text files 73–80, 138–41
- tools 190–1, 231, 253
- trace tables 94–9
- transistors 110, 176, 267
- translators 194, 196
- transmission control protocol *see* TCP; TCP/IP
- traversal 15–16, 55–6, 62, 140
- truth tables 181–3
- try...catch blocks of code 75–6

- two-dimensional arrays 65–8, 69
 - and text files 77–80
- two-factor authentication 223
- two's complement 117–19, 121–2
- type coercion 37, 294
- UEFI (unified extensible firmware interface) 168
- Unicode 129
- unpatched software 228
- URL (uniform resource locator) 144, 235–6
- USB devices 229
- USB flash drives 176, 226
- user input 70–3
- user interface 189, 261
- user policies 231–2
- usernames 59–60, 71, 204–5, 227
- users 189, 202–3, 222–3
- utility software 187, 190–1, 231
- validation 70–3, 222–3
- variables 8, 9, 295
 - data types 35, 37
 - descriptive names 51–2
 - initialisation 35–6
 - and subprograms 83–7
- Vigenère cipher 149–52
- virtual memory 170
- virtualisation 177
- viruses 191, 222
- von Neumann architecture 165–7
- WAV files 142–3
- web browsers 204, 216, 234, 235–6
- web pages 204–5, 235–6
- web servers 204–5, 216, 235–6
- websites 10, 204, 235, 251
 - fake 227, 228
- 'what if?' question 191–2
- 'what3words' 256
- WHILE loop 70–2, 93–4
- Wi-Fi 211, 214
- wide area network (WAN) 203–4
- wired connections 211
- wireless access points (WAP) 238
- wireless connectivity 211, 214, 219
- wireless local area network (WLAN)
 - 203, 214
- World Wide Web 235–6
- writing to a file 73–5, 76–8, 79
- writing to memory 167–8
- written descriptions 5–6, 10