Cambridge Assessment International Education

Interactive

Learner Guide

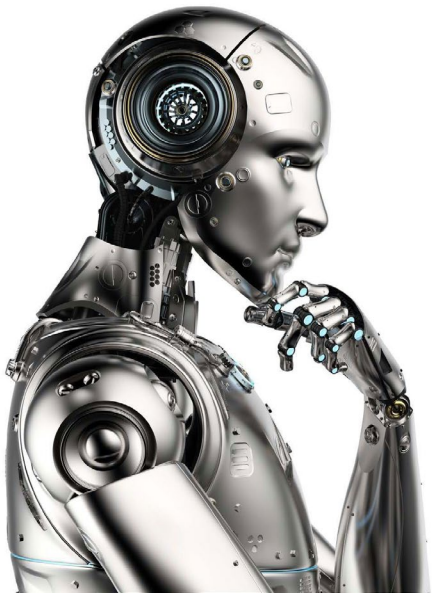
Cambridge IGCSE™ / Cambridge IGCSE (9–1)

Computer Science 0478 / 0984

Cambridge O Level

Computer Science 2210

For examination from 2023



Version 1

Cambridge Pathway

In order to help us develop the highest quality resources, we are undertaking a continuous programme of review; not only to measure the success of our resources but also to highlight areas for improvement and to identify new development needs.

We invite you to complete our survey by visiting the website below. Your comments on the quality and relevance of our resources are very important to us.

www.surveymonkey.co.uk/r/GL6ZNJB

Contents

About this guide

This guide explains what you need to know about your Cambridge IGCSE and O Level Computer Science course and examinations. You should use this guide alongside the support of your teacher.

It will help you to:

- ✓ understand what skills you should develop by taking this Cambridge IGCSE/O Level course
- ✓ understand how you will be assessed
- ✓ understand what we are looking for in the answers you write
- ✓ plan your revision programme
- ✓ revise, by providing revision tips and an interactive revision checklist (Section 5).

The aims of this syllabus are to enable you to develop:

- computational thinking skills
- an understanding of the main principles of solving problems using computers
- the skills necessary to solve computer-based problems using a high-level programming language
- an understanding of the component parts of computer systems and how they interrelate
- an understanding of the internet as a means of communication and its associated risks
- an understanding of the development and use of automated and emerging technologies.

Section 1: Syllabus content - what you need to know about

This section gives you an outline of the syllabus content for this course. Ask your teacher for more detail about each topic. You can also find more detail in the Revision checklists of this guide.

Computer systems

1. Data representation
2. Data transmission
3. Hardware
4. Software
5. The internet and its uses
6. Automated and emerging technologies

Algorithms, programming and logic

7. Algorithm design and problem-solving
8. Programming
9. Databases
10. Boolean logic

Make sure you always check the latest syllabus, which is available at www.cambridgeinternational.org

Section 2: How you will be assessed

You will be assessed at the end of the course using **two** components:

- Paper 1 Computer Systems
- Paper 2 Algorithms, Programming and Logic

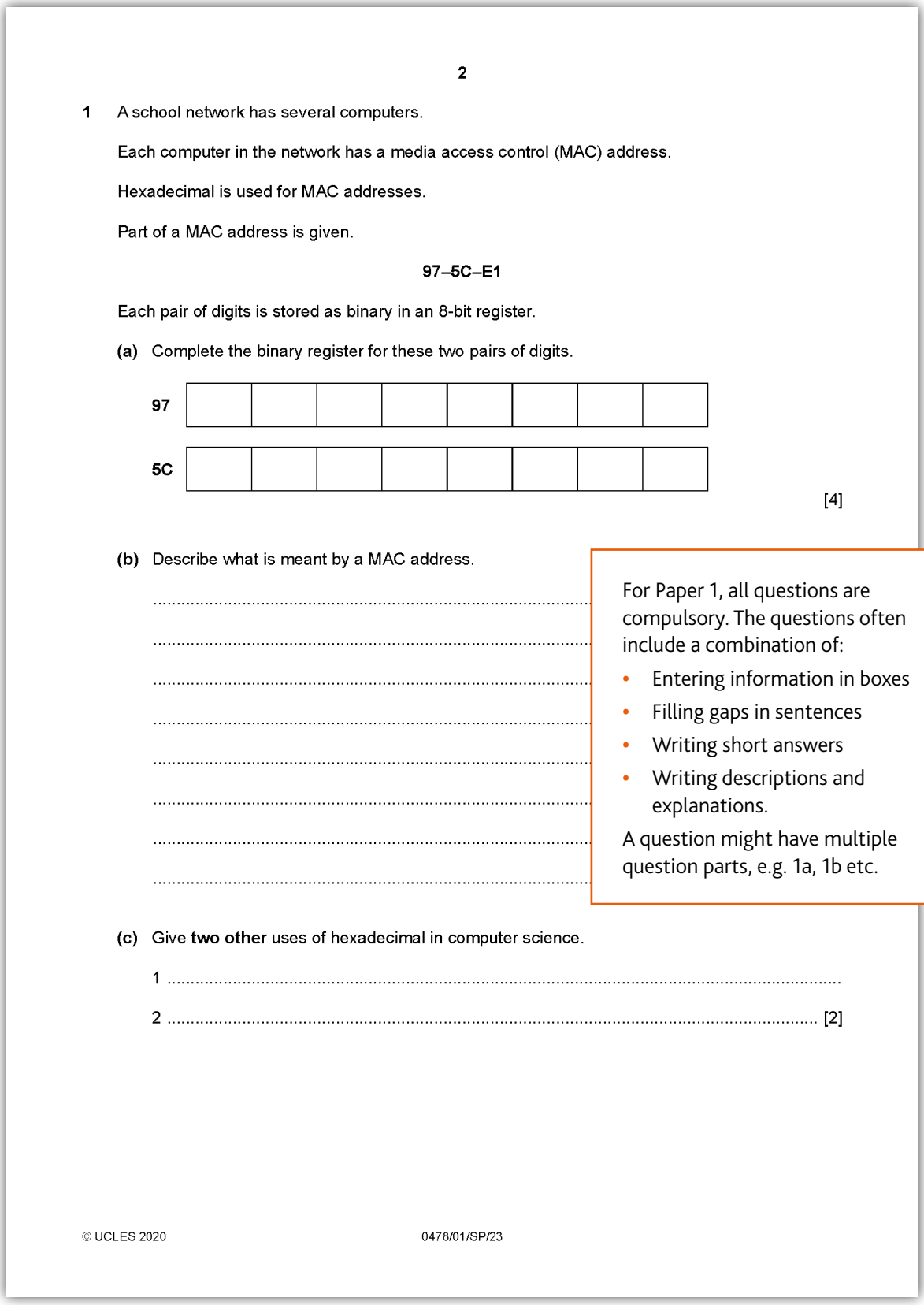
Components at a glance

This table summarises the key information about each examination paper. You can find details and advice on how to approach each component in the 'About each paper' sub-section.

Component	Time and marks	Content/Skills assessed	Details	Percentage of qualification
Paper 1 Computer Systems	1 hour 45 minutes 75 marks	Questions will be based on Topics 1–6 of the subject content	Short-answer and structured questions All questions are compulsory No calculators are permitted Externally assessed	50%
Paper 2 Algorithms, Programming and Logic	1 hour 45 minutes 75 marks	Questions will be based on Topics 7–10 of the subject content	Short-answer and structured questions and a scenario-based question All questions are compulsory No calculators are permitted Externally assessed	50%

About each paper

Paper 1: Computer Systems



2

1 A school network has several computers.

Each computer in the network has a media access control (MAC) address.

Hexadecimal is used for MAC addresses.

Part of a MAC address is given.

97–5C–E1

Each pair of digits is stored as binary in an 8-bit register.

(a) Complete the binary register for these two pairs of digits.

97								
5C								

[4]

(b) Describe what is meant by a MAC address.

.....

.....

.....

.....

.....

.....

.....

.....

(c) Give **two other** uses of hexadecimal in computer science.

1 .....

2 ..... [2]

© UCLES 2020 0478/01/SP/23

For Paper 1, all questions are compulsory. The questions often include a combination of:

- Entering information in boxes
- Filling gaps in sentences
- Writing short answers
- Writing descriptions and explanations.

A question might have multiple question parts, e.g. 1a, 1b etc.

Paper 2: Algorithms, Programming and Logic

5 An algorithm has been written in pseudocode to input some numbers. It only outputs any numbers that are greater than or equal to 100. The number 999 is not output and stops the algorithm.

```
INPUT Number
WHILE Numbers <> 999 DO
    IF Number > 100
        THEN
            OUTPUT Number
    ENDIF
ENDWHILE
OUTPUT Number
```

(a) Identify the **four** errors in the pseudocode and suggest corrections.

Error 1 .....

Correction .....

.....

Error 2 .....

Correction .....

.....

Error 3 .....

Correction .....

.....

Error 4 .....

Correction .....

.....

[4]

(b) Write a pseudocode statement to change the corrected algorithm to output all numbers between 100 and 200 inclusive.

You do **not** need to rewrite the whole algorithm

.....

.....

.....

..... [2]

For Paper 2, all questions are compulsory. The questions require candidates to have practical programming experience. Knowledge of programming language syntax is not examined; in all cases the logic is more important than the syntax.

Paper 2: Algorithms, Programming and Logic continued

13 The 1D array `StudentName[]` contains the names of students in a class. The 2D array `StudentMark[]` contains the mark for each subject, for each student. The position of each student's data in the two arrays is the same, for example, the student in position 10 in `StudentName[]` and `StudentMark[]` is the same.

The variable `ClassSize` contains the number of students in the class. The variable `SubjectNo` contains the number of subjects studied. All students study the same number of subjects.

The arrays and variables have already been set up and the data stored.

Students are awarded a grade based on their average mark.

Average mark	Grade awarded
greater than or equal to 70	distinction
greater than or equal to 55 and less than 70	merit
greater than or equal to 40 and less than 55	pass
less than 40	fail

Write a program that meets the following requirements:

- calculates the combined total mark for each student for all their subjects
- calculates the average mark for each student for all their subjects, rounded to the nearest whole number
- outputs for each student:
  - name
  - combined total mark
  - average mark
  - grade awarded
- calculates, stores and outputs the number of distinctions, merits, passes and fails for the whole class.

You must use pseudocode or program code **and** add comments to explain how your code works.

You do **not** need to initialise the data in the array.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

The final question in Paper 2 is a 15-mark unseen scenario question. You will be required to write an algorithm using pseudocode or program code for the context provided. It is expected that you should spend 30 minutes answering this question in the exam.

Section 3: What skills will be assessed

The areas of knowledge, understanding and skills that you will be assessed on are called **assessment objectives** (AO).

The examiners take account of the following skills areas (**AO1**, **AO2** and **AO3**) in the examination papers:

- Knowledge and understanding
- Application of knowledge and understanding, and analysis and design
- Evaluation, judgement and conclusions.

It is important that you know the different weightings (%) of the assessment objectives, as this affects how the examiner will assess your work.

- Assessment objective 1 (AO1) is worth 60% of the total marks in Paper 1, and 20 % of the total marks in Paper 2.
- Assessment objective 2 (AO2) is worth 20% of the total marks in Paper 1, and 60% of the total marks in Paper 2.
- Assessment objective 3 (AO3) is worth 20% of the total marks in Paper 1, and 20% of the total marks in Paper 2.

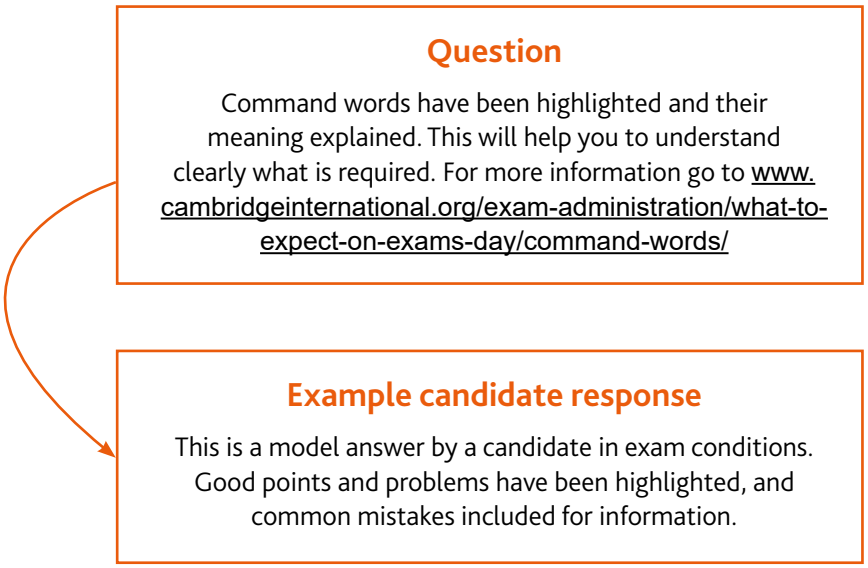
Assessment objectives (AO)	What does the AO mean?	What do you need to be able to do?
AO1 **Demonstrate knowledge and understanding of the principles and concepts of computer science.**	This means you will be need to remember definitions and descriptions. These will be generic, i.e. they will not be around a context. You could be asked to define a term, and describe or explain what something means.	You will need to know the definitions of the terms that are in the syllabus. You will need to be able to give descriptions of what something means, or what something is. You will need to explain how or why something happens, or does not happen. These will all be based around the content that you will have learnt without needing to think about how it could be used in a new way or context.
AO2 **Apply knowledge and understanding of the principles and concepts of computer science to a given context, including the analysis and design of computational or programming problems.**	This means you will need to use your knowledge in a specific context. You have to apply what you have learnt.	You will need to consider what you know, and use this in the context given. For example, if the question is about a student needing to store some documents, you need to relate every point you make back to this student, and why your choices, or points, are relevant to them. You will need to use your computational thinking and programming knowledge to solve problems. This could be that you are given a flowchart, or program, that you have not seen before. You could be asked questions about that algorithm, and you will need to use what you know and consider it in this example.
AO3 **Provide solutions to problems by:** • **evaluating computer systems** • **making reasoned judgements** • **presenting conclusions.**	This means you will need to decide why one decision is more appropriate than another and explain why.	You will need to look at a context and decide what is most appropriate for it. For example, is it more appropriate to use lossy compression for an image instead of lossless. Then, you need to defend your choice, by explaining why it is the best decision. You will need to do this by referring every point to the context, so you are not giving generic definitions or descriptions.

Section 4: Example candidate response

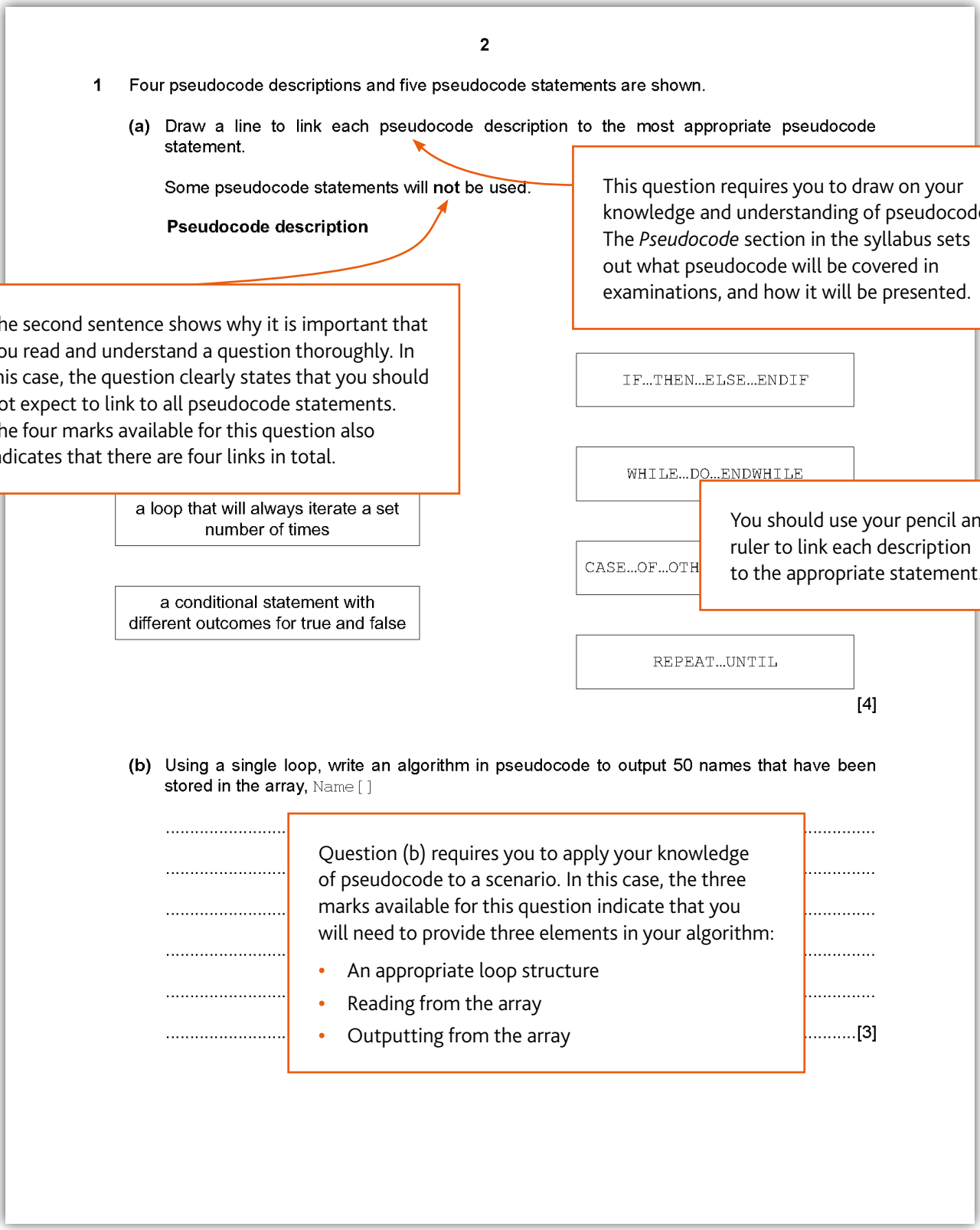
This section takes you through an example question and model answer from a Cambridge IGCSE Computer Science specimen paper. It will help you to see how to identify command words within questions and to understand what is required in your response. A command word is the part of the question that tells you what you need to do with your knowledge. For example, you might need to describe something, explain something, argue a point of view or list what you know.

All information and advice in this section is specific to the example question and response/ mode answer being demonstrated. It should give you an idea of how your responses might be viewed by an examiner but it is not a list of what to do in all questions. In your own examination, you will need to pay careful attention to what each question is asking you to do.

This section is separated as follows:



Question



2

1 Four pseudocode descriptions and five pseudocode statements are shown.

(a) Draw a line to link each pseudocode description to the most appropriate pseudocode statement.

Some pseudocode statements will **not** be used.

Pseudocode description

a loop that will always iterate a set number of times

a conditional statement with different outcomes for true and false

IF...THEN...ELSE...ENDIF

WHILE...DO...ENDWHILE

CASE...OF...OTH

REPEAT...UNTIL

[4]

(b) Using a single loop, write an algorithm in pseudocode to output 50 names that have been stored in the array, `Name[]`

.....

.....

.....

.....

.....

.....[3]

This question requires you to draw on your knowledge and understanding of pseudocode. The *Pseudocode* section in the syllabus sets out what pseudocode will be covered in examinations, and how it will be presented.

The second sentence shows why it is important that you read and understand a question thoroughly. In this case, the question clearly states that you should not expect to link to all pseudocode statements. The four marks available for this question also indicates that there are four links in total.

You should use your pencil and ruler to link each description to the appropriate statement.

Question (b) requires you to apply your knowledge of pseudocode to a scenario. In this case, the three marks available for this question indicate that you will need to provide three elements in your algorithm:

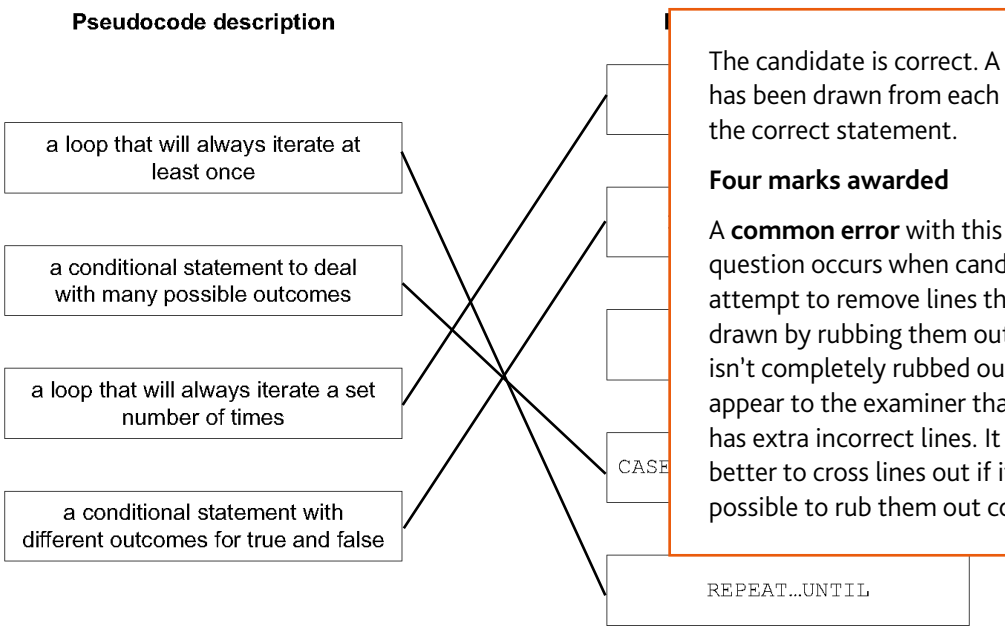
- An appropriate loop structure
- Reading from the array
- Outputting from the array

Example candidate response

1 Four pseudocode descriptions and five pseudocode statements are shown.

(a) Draw a line to link each pseudocode description to the most appropriate pseudocode statement.

Some pseudocode statements will **not** be used.



[4]

The candidate is correct. A single line has been drawn from each descriptor to the correct statement.

Four marks awarded

A **common error** with this type of question occurs when candidates attempt to remove lines they have drawn by rubbing them out. If the line isn't completely rubbed out, it may appear to the examiner that the answer has extra incorrect lines. It is generally better to cross lines out if it is not possible to rub them out completely.

(b) Using a single loop, write an algorithm in pseudocode to output 50 names that have been stored in the array, `Name[]`

```
FOR Counter ← 0 TO 50
    OUTPUT Name[Counter]
NEXT Counter
```

An appropriate loop structure `FOR … NEXT` has been chosen. The array index, Counter, started at zero. However, the final array index should be 49 as an index of 50 will mean that 51 names would be output. Therefore, no mark could be awarded for the loop control.

Two marks can be awarded for reading the elements from the array and outputting their values.

Two marks awarded

A **common error** with this question is not remembering that an array starting with index zero with finish, should have a final element stored at the length of the array minus 1.

Section 5: Revision

This advice will help you revise and prepare for the examinations. It is divided into general advice for all papers and more specific advice for Paper 1 and Paper 2.

Use the tick boxes ☐ to keep a record of what you have done, what you plan to do or what you understand.

General advice

Before the examination

- ☐ Find out when the examinations are and plan your revision so you have time to revise. Create a revision timetable and divide it into sections to cover each topic.
- ☐ Find out how long each paper is, how many questions you have to answer, how many marks there are for each question, and work out how long you have for each question.
- ☐ Know the meaning of the command words used in questions and how to apply them to the information given. Look at past examination papers and highlight the command words and check what they mean.
- ☐ Make revision notes. Try different styles of notes.
- ☐ Work for short periods then have a break. Revise small sections of the syllabus at a time.
- ☐ Test yourself by writing out key points, redrawing diagrams, creating key cards with the words on one side and definitions and/or examples on the back, etc.
- ☐ Make sure you practise writing selection and iteration constructs. This should include converting between, for example, two different types of loop. You can practise these with pseudocode, or use your chosen programming language to test them and make sure they work.
- ☐ Definitions must not reuse the words to be defined. E.g. *selection* means selecting which code to run. Instead it determines which code to run based on a condition.
- ☐ Make your own dictionary or draw up a glossary of key terms for each section of the syllabus.
- ☐ Look at the programs you have written during lessons and walk through what each line of code does, then working out what the whole program does. This can be tested by running the program and seeing if you are correct.
- ☐ Take the descriptions for programs you have written during lessons, and create the programs again without looking at the previous solutions. Test your programs and amend them until they meet the requirements.
- ☐ Learn to spell computer science terms correctly.
- ☐ Have a look at past questions so that you are clear of what to expect in an examination.
- ☐ Look at mark schemes to help you to understand how the marks are awarded for each question.

In the examination

- ☐ Read the instructions carefully.
- ☐ Plan your time according to the marks for each question. For example, a question worth three marks requires less time and a shorter answer than one worth 10 marks. If a question has several parts, then the parts with more marks will need more time and more developed answers.
- ☐ Do not leave out questions or parts of questions. Remember, no answer means no mark.
- ☐ Read each question very carefully.
  - Identify the command words – you could underline or highlight them.
  - Identify the other key words and perhaps underline them too.
  - Try to put the question into your own words to understand what it is really asking.
- ☐ Read all parts of a question before starting your answer. Think carefully about what is needed for each part.

- [ ] **Answer the question**. This is very important!

  Use your knowledge and understanding.

  Do not just write all you know, only write what is needed to answer the question.

- [ ] Plan your answers. Clear, concise, well-ordered, well-argued, well-supported answers get more marks than long, rambling, muddled, repetitious answers. Quality is better than quantity.

- [ ] Make sure your writing is clear and easy to read. It is no good writing a brilliant answer if the examiner cannot read it!

Paper 1 advice

- [ ] Check the command word that is used, for example describe requires more than a statement. *Explain* requires more detail as to why something happens, or why you have selected the answer you have.

- [ ] Check whether tick box questions require one, or possibly more ticks per row. If it is one tick per row then make sure you don't put two. If there could be more check each possibility, it is very likely that at least one of the rows requires two or more ticks.

- [ ] If a question gives a context, and the wording of the command refers back to this context, then make sure you use it in your answer. Don't just give a generic answer, use the context in each point you make.

- [ ] If a question asks for differences between two or more items, then make sure you give all sides. For example just stating that parallel transmission has multiple wires is not enough without also stating that serial has just one wire.

- [ ] When converting numbers between different types (e.g. binary, denary, hexadecimal), always check your answers by working it the reverse. For example, if converting binary to denary, convert your denary answer back to binary and make sure that they both match.

Paper 2 advice

- [ ] When drawing logic gate diagrams repeat the process backwards to make sure your answer is correct. For example draw the diagram for the statement, then write the statement or your diagram and check that they match.

- [ ] Make sure you have answered the algorithm questions in the way they want the answer. If the question asks for a flowchart, make sure you draw a flowchart. If a question asks for a description, then only drawing a flowchart or writing code will not be acceptable.

- [ ] Split your time appropriately between the scenario question at the end of the paper and the rest of the questions. You should spend approximately 30 minutes on the scenario question. Do not got too much over this otherwise you may not have enough time for the rest of the exam paper.

- [ ] If you get stuck on an algorithm, don't spend a long time trying to fix it. Move on and you can always come back later after you've answered the other questions, in that time away you might be able to look at it afresh.

- [ ] Test run any algorithms you write. Use some sample data and work through each line of your algorithm to make sure it works. If it doesn't, then you should be able to find out where the problem is.

- [ ] Check your spelling and case (lower or upper) especially with any database questions, e.g. queries. Spelling and case must be exact, if in the database table it is called HOUSE, then writing house is inaccurate. After you have answered each question check the words you have used against those in the question.

- [ ] If a question asks for an example, make sure you give one as there will be at least 1 mark available for this.

Revision checklists

In the next part of this guide we have provided some revision checklists. These include information from the syllabus that you should revise. They don't contain all the detailed knowledge you need to know, just an overview. For more detail see the syllabus and talk to your teacher.

The table headings are explained below:

Topic	You should be able to	R	A	G	Comments
These are the core topics from the syllabus	Content in the syllabus you need to cover	You can use the tick boxes to show when you have revised an item and how confident you feel about it. R = RED means you are really unsure and lack confidence; you might want to focus your revision here and possibly talk to your teacher for help A = AMBER means you are reasonably confident but need some extra practice G = GREEN means you are very confident. As your revision progresses, you can concentrate on the RED and AMBER items in order to turn them into GREEN items. You might find it helpful to highlight each topic in red, orange or green to help you prioritise.			You can: • add further information of your own, such as names of case studies needed • add learning aids, such as rhymes, poems or word play • pinpoint areas of difficulty you need to check further with your teacher or textbooks • include reference to a useful resource

Note: the tables below cannot contain absolutely everything you need to know, but it does use examples wherever it can.

1 Data representation

Topic	You should be able to	R	A	G	Comments
1.1 Number systems					
1	Understand how and why computers use binary to represent all forms of data	☐	☐	☐	
2	a. Understand the denary, binary and hexadecimal number systems b. Convert between i) positive denary and positive binary ii) positive denary and positive hexadecimal iii) positive hexadecimal and positive binary	☐	☐	☐	
3	Understand how and why hexadecimal is used as a beneficial method of data representation	☐	☐	☐	
4	a. Add two positive 8-bit binary integers b. Understand the concept of overflow and why it occurs in binary addition	☐	☐	☐	
5	Perform a logical binary shift on a positive 8-bit binary integer and understand the effect this has on the positive binary integer	☐	☐	☐	
6	Use two's complement to represent positive and negative 8-bit binary integers	☐	☐	☐	
1.2 Text, sound and images					
1	Understand how and why a computer represents text and the use of character sets, including American standard code for information interchange (ASCII) and Unicode	☐	☐	☐	
2	Understand how and why a computer represents sound, including the effects of the sample rate and sample resolution	☐	☐	☐	
3	Understand how and why a computer represents an image, including the effects of the resolution and colour depth	☐	☐	☐	
1.3 Data storage and compression					
1	Understand how data storage is measured	☐	☐	☐	
2	Calculate the file size of an image file and a sound file, using information given	☐	☐	☐	
3	Understand the purpose of and need for data compression	☐	☐	☐	
4	Understand how files are compressed using lossy and lossless compression methods	☐	☐	☐	

2 Data transmission

Topic	You should know and understand	R	A	G	Comments
2.1 Types and methods of data transmission					
1	a. Understand that data is broken down into packets to be transmitted b. Describe the structure of a packet c. Describe the process of packet switching	☐	☐	☐	
2	a. Describe how data is transmitted from one device to another using different methods of data transmission b. Explain the suitability of each method of data transmission, for a given scenario	☐	☐	☐	
2.2 Methods of error detection					
1	Understand the need to check for errors after data transmission and how these errors can occur	☐	☐	☐	
2	Describe the processes involved in each of the following error detection methods for detecting errors in data after transmission: parity check (odd and even), checksum and echo check	☐	☐	☐	
3	Describe how a check digit is used to detect errors in data entry and identify examples of when a check digit is used, including international standard book numbers (ISBN) and bar codes	☐	☐	☐	
4	Describe how an automatic repeat query (ARQ) can be used to establish that data is received without error	☐	☐	☐	
2.3 Encryption					
1	Understand the need for and purpose of encryption when transmitting data	☐	☐	☐	
2	Understand how data is encrypted using symmetric and asymmetric encryption	☐	☐	☐	

3 Hardware

Topic	You should know and understand	R	A	G	Comments
3.1 Computer architecture					
1	a. Understand the role of the central processing unit (CPU) in a computer b. Understand what is meant by a microprocessor	☐	☐	☐	
2	a. Understand the purpose of the components in a CPU, in a computer that has a Von Neumann architecture b. Describe the process of the fetch–decode–execute cycle including the role of each component in the process	☐	☐	☐	
3	Understand what is meant by a core, cache and clock in a CPU and explain how they can affect the performance of a CPU	☐	☐	☐	
4	Understand the purpose and use of an instruction set for a CPU	☐	☐	☐	
5	Describe the purpose and characteristics of an embedded system and identify devices in which they are commonly used	☐	☐	☐	
3.2 Input and output devices					
1	Understand what is meant by an input device and why it is required	☐	☐	☐	
2	Understand what is meant by an output device and why it is required	☐	☐	☐	
3	a. Understand what is meant by a sensor and the purposes of sensors b. Identify the type of data captured by each sensor and understand when each sensor would be used, including selecting the most suitable sensor for a given context	☐	☐	☐	
3.3 Data storage					
1	Understand what is meant by primary storage	☐	☐	☐	
2	Understand what is meant by secondary storage	☐	☐	☐	
3	Describe the operation of magnetic, optical and solid-state (flash memory) storage and give examples of each	☐	☐	☐	
4	Describe what is meant by virtual memory, how it is created and used and why it is necessary	☐	☐	☐	

Topic	You should know and understand	R	A	G	Comments
5	Understand what is meant by cloud storage	☐	☐	☐	
6	Explain the advantages and disadvantages of storing data on the cloud in comparison to storing it locally	☐	☐	☐	
3.4 Network hardware					
1	Understand that a computer needs a network interface card (NIC) to access a network	☐	☐	☐	
2	Understand what is meant by and the purpose of a media access control (MAC) address, including its structure	☐	☐	☐	
3	a. Understand what is meant by and the purpose of an internet protocol (IP) address b. Understand that there are different types of IP address	☐	☐	☐	
4	Describe the role of a router in a network	☐	☐	☐	

4 Software

Topic	You should know and understand	R	A	G	Comments
4.1 Types of software and interrupts					
1	Describe the difference between system software and application software and provide examples of each	☐	☐	☐	
2	Describe the role and basic functions of an operating system	☐	☐	☐	
3	Understand how hardware, firmware and an operating system are required to run applications software	☐	☐	☐	
4	Describe the role and operation of interrupts	☐	☐	☐	
4.2 Types of programming language, translators and integrated development environments (IDEs)					
1	Explain what is meant by a high-level language and a low-level language, including the advantages and disadvantages of each	☐	☐	☐	
2	Understand that assembly language is a form of low-level language that uses mnemonics, and that an assembler is needed to translate an assembly language program into machine code	☐	☐	☐	

Topic	You should know and understand	R	A	G	Comments
3	Describe the operation of a compiler and an interpreter, including how high-level language is translated by each and how errors are reported				
4	Explain the advantages and disadvantages of a compiler and an interpreter				
5	Explain the role of an IDE in writing program code and the common functions IDEs provide				

5 The internet and its uses

Topic	You should know and understand	R	A	G	Comments
5.1 The internet and the world wide web					
1	Understand the difference between the internet and the world wide web				
2	Understand what is meant by a uniform resource locator (URL)				
3	Describe the purpose and operation of hypertext transfer protocol (HTTP) and hypertext transfer protocol secure (HTTPS)				
4	Explain the purpose and functions of a web browser				
5	Describe how web pages are located, retrieved and displayed on a device when a user enters a URL				
6	Explain what is meant by cookies and how they are used, including session cookies and persistent cookies				
5.2 Digital currency					
1	Understand the concept of a digital currency and how digital currencies are used				
2	Understand the process of blockchain and how it is used to track digital currency transactions				
5.3 Cyber security					
1	Describe the processes involved in, and the aim of carrying out, a range of cyber security threats				
2	Explain how a range of solutions are used to help keep data safe from security threats				

6 Automated and emerging technologies

Topic	You should know and understand	R	A	G	Comments
6.1 Automated systems					
1	Describe how sensors, microprocessors and actuators can be used in collaboration to create automated systems				
2	Describe the advantages and disadvantages of an automated system used for a given scenario				
6.2 Robotics					
1	Understand what is meant by robotics				
2	Describe the characteristics of a robot				
3	Understand the roles that robots can perform and describe the advantages and disadvantages of their use				
6.3 Artificial intelligence					
1	Understand what is meant by artificial intelligence (AI)				
2	Describe the main characteristics of AI as the collection of data and the rules for using that data, the ability to reason, and can include the ability to learn and adapt				
3	Explain the basic operation and components of AI systems to simulate intelligent behaviour				

7 Algorithm design and problem-solving

Topic	You should know and understand	R	A	G	Comments
7 Algorithm design and problem-solving					
1	Understand the program development life cycle, limited to: analysis, design, coding and testing				
2	a. Understand that every computer system is made up of sub-systems, which are made up of further sub-systems b. Understand how a problem can be decomposed into its component parts c. Use different methods to design and construct a solution to a problem				
3	Explain the purpose of a given algorithm				

Topic	You should know and understand	R	A	G	Comments
4	Understand standard methods of solution				
5	a. Understand the need for validation checks to be made on input data and the different types of validation check b. Understand the need for verification checks to be made on input data and the different types of verification check				
6	Suggest and apply suitable test data				
7	Complete a trace table to document a dry-run of an algorithm				
8	Identify errors in given algorithms and suggest ways of correcting these errors				
9	Write and amend algorithms for given problems or scenarios, using: pseudocode, program code and flowcharts				

8 Programming

Topic	You should know and understand	R	A	G	Comments
8.1 Programming concepts					
1	Declare and use variables and constants				
2	Understand and use the basic data types				
3	Understand and use input and output				
4	a. Understand and use the concept of sequence b. Understand and use the concept of selection c. Understand and use the concept of iteration d. Understand and use the concepts of totalling and counting e. Understand and use the concept of string handling f. Understand and use arithmetic, logical and Boolean operators				
5	Understand and use nested statements				
6	a. Understand what is meant by procedures, functions and parameters b. Define and use procedures and functions, with or without parameters c. Understand and use local and global variables				

Topic	You should know and understand	R	A	G	Comments
7	Understand and use library routines				
8	Understand how to create a maintainable program				
8.2 Arrays					
1	Declare and use one-dimensional (1D) and two-dimensional (2D) arrays				
2	Understand the use of arrays				
3	Write values into and read values from an array using iteration				
8.3 File handling					
1	Understand the purpose of storing data in a file to be used by a program				
2	Open, close and use a file for reading and writing				

9 Databases

Topic	You should know and understand	R	A	G	Comments
9 Databases					
1	Define a single-table database from given data storage requirements				
2	Suggest suitable basic data types				
3	Understand the purpose of a primary key and identify a suitable primary key for a given database table				
4	Read, understand and complete structured query language (SQL) scripts to query data stored in a single database table				

10 Boolean logic

Topic	You should know and understand	R	A	G	Comments
10 Boolean logic					
1	Identify and use the standard symbols for logic gates	☐	☐	☐	
2	Define and understand the functions of the logic gates	☐	☐	☐	
3	a. Use logic gates to create given logic circuits from a: ii) problem statement iii) logic expression iv) truth table b. Complete a truth table from a: i) problem statement ii) logic expression iii) logic circuit c. Write a logic expression from a: i) problem statement ii) logic circuit iii) truth table	☐	☐	☐	

Section 6: Useful resources

The resources listed below will help you to revise and study for your Cambridge IGCSE and O Level Computer Science course.

These resources have not been through the Cambridge quality assurance process but have been found suitable for use with various parts of the syllabus. This list includes website links providing direct access to internet resources. Cambridge is not responsible for the accuracy or content of information contained in these resources. The inclusion of a link to an external website should not be understood to be an endorsement of that website or the site's owners (or their products/services).

www.bbc.co.uk/bitesize/subjects/z34k7ty
Includes several theory and programming aspects relevant to this syllabus with quizzes.

www.khanacademy.org/
Website allows learners to sign up and practise their programming techniques independently, particularly useful for Paper 2.

www.101computing.net/LMC/
Simulation of a processor including the different registers. Learners can enter pre-written programs and watch how and when the values changes.

www.computerscience.gcse.guru
Website with theory and quizzes, written for a range of syllabuses so not all is relevant.

You can find a resource list, including endorsed resources to support Cambridge IGCSE and O Level Computer Science on our public website [IGCSE] [O Level]

Endorsed resources have been written to be closely aligned to the syllabus they support, and have been through a detailed quality assurance process. All textbooks endorsed by Cambridge International for this syllabus are the ideal resource to be used alongside this Learner Guide.

In addition to reading the syllabus, you should refer to the past and specimen papers.

Cambridge Assessment International Education
The Triangle Building, Shaftesbury Road, Cambridge, CB2 8EA, United Kingdom
t: +44 1223 553554
e: info@cambridgeinternational.org www.cambridgeinternational.org