Qualification Accredited

OCR
Oxford Cambridge and RSA

GCSE (9-1)

COMPUTER SCIENCE

J277

For first teaching in 2020

Exploring our question papers

Version 1

ocr.org.uk/computerscience

Contents

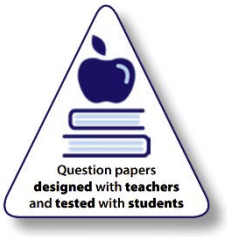
Introduction

This guide is to help you understand our examination papers. It shares the story of our assessment approach and explores our question papers with you.

It should be used to support candidates with how to access our exam papers.

Practical programming remains an **important** part of GCSE Computer Science. Candidates who do well on Component 02 are usually those who show experience of practical programming within the classroom.

This guide demonstrates how classroom practice relates to the questions your students will answer in the exam.



We tested our sample exam papers with students and teachers.

We are committed to ensuring that our papers are fair and accessible.

You can read about our accessibility principles in this document.

Question paper structure

Our GCSE (9-1) Computer Science consists of two components. Candidates must take both papers.

The qualification is marked out of a **total** of **160 marks**.

The marks, paper duration and weightings are the same for both components.

Component	Marks	Duration	Weighting
Component 01**:** Computer Systems This component focuses on the theoretical understanding of Computer Science.	80	1 hour 30 minutes	50%
Component 02**:** Computational thinking, algorithms and programming This component focuses on the practical application of Computer Science: program/problem design, writing testing and refinement.	80	1 hour 30 minutes	50%

Exam Reference Language and high-level programming languages

High-level programming languages

We do not restrict the range of programming languages you can use to teach in the classroom.

Our approach gives you control over how you deliver programming within your classroom. It allows you to teach in multiple languages.

This helps demonstrate programming techniques which may not exist in one specific language.

Teacher Tip

Some programming languages may not allow you to demonstrate all the techniques listed in the specification. It is important to ensure you cover all techniques listed in the specification to prepare your students effectively.

Exam Reference Language (ERL)

Our Exam Reference Language is the term used for 'formal' pseudocode. It allows us to write code-based questions consistently in our examinations. This helps familiarity and accessibility for students.

High-level programming languages have similar key words for core constructs. Our Exam Reference Language also adopts similar key words for core constructs.

Our ERL is given in the Specification. Examples of its use may be found in the Sample Assessment Materials and in our past papers on Teach Cambridge.

Teacher Tip

Candidates do not have to memorise or respond to questions using ERL.

They do need to be able to be familiar with it to understand the questions.

In Component 2 Section B, some questions require a response in either ERL or a high-level language. This tests that they have experienced programming in the classroom.

Between 10% and 15% of the 80 marks available for Component 02 are allocated to these types of question. This means that **8-12 marks** test candidates' practical experience of programming.

Teacher Tip

Candidates can write in a high-level language or ERL in both sections of the paper where appropriate. Stronger candidates often write their responses in the language they have used within the classroom.

Exam overview

High level assessment objectives and weightings

Each examination question tests one or more Assessment Objective.

Assessment Objectives are defined by Ofqual and have required, associated weightings.

Our published mark schemes indicate the precise Assessment Objective targeted in each question.

	Objective	Weighting
AO1	Demonstrate knowledge and understanding of the key concepts and principles of Computer Science.	30%
AO2	Apply knowledge and understanding of key concepts and principles of Computer Science.	40%
AO3	Analyse problems in computational terms: • to make reasoned judgements • to design, program, evaluate and refine solutions.	30%

Assessment Objectives by Component and Section

	Component 01	Component 02	
		Section A	Section B
AO1	✓	✓	
AO2	✓	✓	
AO3		✓	✓

Our examination

We split our papers equally between Component 01 and 02.

Each is worth 50% of the total qualification. There are 80 marks available for each examination paper.

The qualification is out of 160 marks.

Synopticity

Synoptic assessments test candidates' understanding of connections between different elements of a subject. It involves the explicit drawing together of knowledge, skills and understanding from different parts of the course.

For example, when assessing programming skills in Component 02, questions may also draw upon knowledge, skills and understanding from Component 01.

Candidates will also benefit from a wider understanding of the impacts of Computer Science within society and the emerging development of technology. This experience can help candidates relate examination questions to real-life issues.

Showing working

Some questions specifically state to show working out. This helps assess the thinking of the candidate. A common example of this is when converting between denary, binary and hexadecimal.

Candidates can gain marks for showing working where asked. This is true even if the final answer may be incorrect.

Teacher Tip

It is good practice to show your working, even if not required to do so.
This can help prevent mistakes.

Component 01

Component 01 is a 'traditional' theory paper.

It is made up of a range of question types. It follows the same style as the legacy J276 Component 01 papers. This means that many of the historical questions from J276 may be used as past-paper questions. However, bear in mind that there have been minor content updates.

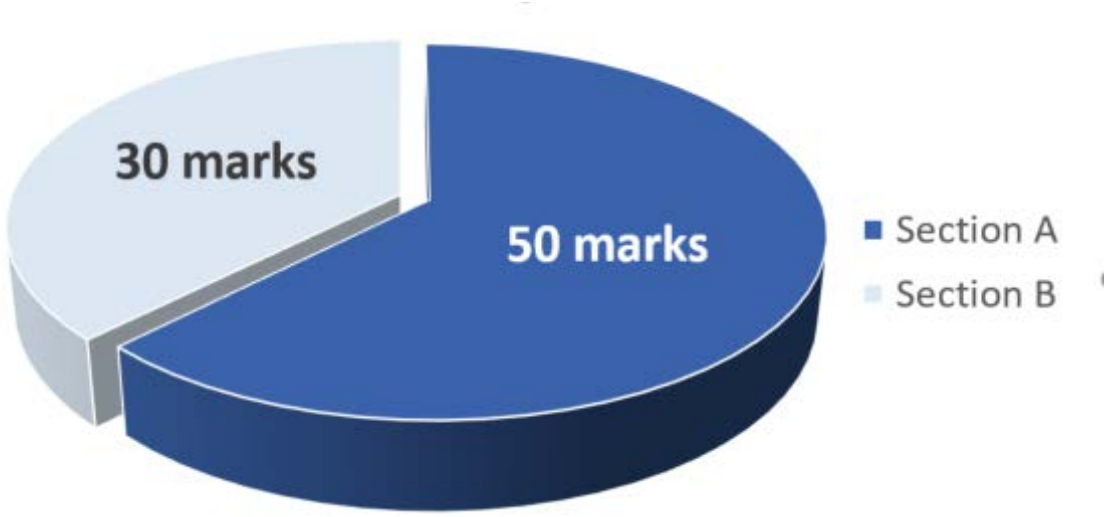
Component 01 also contains a single 8 mark extended response question. This is marked using a slightly different approach to other questions. We use 'levels' to assess this question.

Each level contains a description of key things we are looking for. The answer is judged using a 'best fit' approach.

Component 02

We have divided Component 02 into two sections.

The breakdown of marks between both sections can be seen below:



Section A: 50 marks

Section A covers general questions around computational thinking, algorithms and programming.

Section A mostly consists of short and medium answer questions. These are marked using a points-based mark scheme.

Section A also contains one six-mark question. We mark this using a points-based mark scheme. A points-based mark scheme suits the algorithmic nature of these questions.

Candidates have flexibility and choice in how they present their answers in this section.

Unless stated, candidates may respond using:

- ✓ Pseudocode
- ✓ Flowcharts
- ✓ Bullet points
- ✓ OCR Exam Reference Language
- ✓ A high-level programming language

This flexibility allows us to focus on assessing a candidate's ability to structure answers logically without a focus on syntactic precision.

Section B: 30 marks

Section B tests a candidate's programming skills. It tests their ability to **Design**, **Write**, **Test**, and **Refine** programs. Some questions test a candidate's precision when writing algorithms.

Candidates apply their knowledge of programming to problems in computational terms. Some questions may require an algorithmic approach.

Teacher Tip

Candidates often appear to find the programming aspects more challenging.

Candidates who perform better in Component 02 are typically those who have had significant experience of programming within the classroom.

Our approach to marking Section B

When we mark Section B, we test a candidate's ability to form an answer using precise programming syntax.

However, we do not penalise candidates for minor errors in their programming language syntax.

For example, a missing colon would not normally be penalised because this does not stop an examiner from understanding the steps the program should take. However some syntax may be penalised. For example, writing an assignment the wrong way around: 7 = x instead of x = 7.

Use of scenarios

Some questions may use contexts or scenarios. Candidates will then make links and connections between different strands of knowledge and understanding and these contexts/scenarios. Their responses should link to the context/scenario where required.

Teacher Tip

10-15% of marks are awarded for writing algorithms with precision.

That's between 8 to 12 marks in the paper.

Section B has been designed to mimic a **real-life programming experience** as far as possible under examination conditions.

Each question generally builds on the one before. However, questions are not dependent on each other.

Section B only assesses AO3. Section B lends itself to scenario-based, problem-solving questions.

Candidates will draw on their knowledge and experiences from across the full course of study. The specification allows significant time for practical programming. Programming time in the classroom should be used to maximise candidates' familiarity with programming concepts and techniques.

Teacher Tip

Effective programming experiences within the classroom should include understanding of techniques and algorithms. It is also important to develop their ability to combine these techniques to create solutions to more challenging tasks.

Section B is scaffolded to support accessibility. Low demand questions appear earlier in the section. The level of demand increases as candidates progress through this section.

Responding to Section B questions

Section B has four areas of question focus. These are taken from the DfE Subject Requirements of 'Design, Write, Test and Refine'.

This table shows how candidates may respond to each question focus.

Question Focus	Questions asked in:	Candidates respond using:
Design	• Natural English	✓ Pseudocode ✓ Flowchart ✓ Tick box responses ✓ Natural English
Write	• Pseudocode • Natural English • Flow chart	✓ OCR Exam Reference Language ✓ High-level programming language
Test	• OCR Exam Reference Language	✓ Trace tables ✓ Program Logs
Refine	• OCR Exam Reference Language	✓ OCR Exam Reference Language ✓ High-level programming language

Design questions test a candidate's experience of creating ideas for algorithms. We accept the use of flowcharts, pseudocode and natural English when creating algorithm designs. Candidates may also use a high-level language or ERL to respond to these questions.

Write questions require candidates to write responses in either OCR Exam Reference Language or a high-level programming language they are familiar with.

Other response (e.g. flowcharts or natural English) will not be awarded marks.

Test questions assess a candidate's ability to identify how a program is tested to ensure that it works as intended. These questions may include:

- Completing trace tables
- Identifying suitable test data
- Identifying suitable test cases

Refine questions require candidates to write responses in either OCR Exam Reference Language or a high-level programming language they are familiar with.

Other responses (e.g. flowcharts or natural English) will not be awarded marks.

The requirement for precision in **Write** and **Refine** questions test a candidate's experience of using programming language within the classroom. Marks are given for correct use of common elements of syntax which are found across all programming languages.

Questions which require the use of OCR Exam Reference Language, or a high-level programming language will be clearly marked. An example is shown here:

> You must use **either**:
> - OCR Exam Reference Language, **or**
> - A high-level programming language that you have studied.

Accessibility Principles

Accessibility is the term used to define the ease with which a candidate understands a question, and what is required of them.

It does not infer that a question is easy, or hard!

Our Accessibility principles and rationale are outlined below.

These make sure we're always assessing understanding of computer science without letting the language or formatting of our questions be an obstacle for understanding what is needed.

No.	Accessibility Principle	Why?
Look and feel of the paper		
1	**Layout** Clear for all ✓ Arial font will be used except when we write programming code which will use `Courier New`. ✓ Adequate space for responses and room for working in calculations.	To make it easy for candidates to add their responses/do their working. `Courier New` is used for our Exam Reference Language to ensure consistent formatting of spaces and characters.
2	**Tone** Assessing GCSE Computer Science without the language of questions being an obstacle to understanding what is needed ✓ Avoid use of overly complicated language and grammatical constructions. ✓ Contexts and vocabulary will be considered for currency and appropriateness to students, (e.g. glasses not spectacles, dice not die, formulas not formulae). ✓ Language used will be consistent. For example, language use in the stem of a question matches the rest of the question and any titles given diagrams. ✓ Technical words are used appropriately to underpin the GCSE Computer Science being assessed.	To make it as clear as possible what response is expected.
3	Command words used are taken from the defined list of command words for GCSE Computer Science.	To ensure clarity as to what can be assessed and how all command words will be used. These may be found in the specification.
4	Negative questions will be kept to a minimum.	Used well, negative questions can be a good way of testing understanding but can also easily lead to confusion. We will only ever use negatives where it is the most appropriate approach.

No.	Accessibility Principle	Why?
Look and feel of the paper		
5	Where a large context is provided (e.g. a scenario), sentences will be grouped by content rather than separately. Bulleted lists or numbering will be used to help indicate stages in a process/practical method.	To ensure information is presented in the clearest possible way.
6	Names will not be used, unless avoiding their use leads to a complicated question layout.	To avoid imparting cultural/gender bias into questions through choice of name or confusing candidates through choices of names they are unfamiliar with.
7	All text is left aligned (text in table headings will be centred except for row headings, which will be left aligned).	To align with the principles applied to our modified question papers (left alignment is easier to understand for a range of visual impairments).
8	Italics will not be used in questions. Latin abbreviations such as, i.e., e.g. and etc. will not be used. English terms will be used instead. If a specific word requires emphasis, bold font will be used.	Italics can be hard to read if overused.
9	If a question requires an answer to a certain number of decimal places, then we will always ensure this is clearly stated.	To avoid confusing candidates who may be concerned about the required precision needed.
10	Images, diagrams, and data will only be used where they directly support the question. We will avoid candidates needing to turn pages by having any images or diagrams required for a question on the same page, or the facing page.	Avoid distracting images for candidates that do not relate to what is required in the question. To avoid unnecessary page turning.
11	All tables, graphs, images and diagrams will be left aligned.	To align with the principles applied to our modified question papers (left alignment is easier to understand for a range of visual impairments).
12	Text will not be wrapped around images/diagrams/graphs.	To retain clarity.
13	If candidates are required to do something with an image/diagram/graph, it will be centred with sufficient space around it for them to do their working.	To avoid candidates struggling to fit in their response.

Command words

We have defined command words for use within J277 GCSE (9-1) Computer Science.

Command Words are used consistently throughout the examination papers. This allows for:

- ✓ consistent understanding on what style of response is required in a question
- ✓ clarity for teachers in understanding assessment purpose
- ✓ support for teachers when designing internal tests
- ✓ familiarity and consistency within an examination paper
- ✓ clarity for candidates in the expectations of an examination question.

The exact response expected to a command word will be dependent on the context.

We advise candidates to read the full question carefully to be sure of what they are being asked to do.

The list of command words can be found in the latest copy of the specification. This can be downloaded from the Computer Science web page.

Teacher Tip

Understanding command words helps candidates know how to respond to each question. Command words are stated in the specification.

Exploring our question types

We use a range of different question types to allow broad specification coverage, helping us keep our examinations at a manageable length.

Types of questions possible:

		Short answer	Medium answer	Extended response
Component 1		✓	✓	✓ + 8-mark question
Component 2	**A**	✓	✓ + 6-mark question	✗
	B	✓	✓	✗

Short answer questions

Short answer questions may:

- be 1, 2, or 3 mark questions
- test any Assessment Objective (AO1, AO2, AO3)

Short answer questions allow us to:

✓ have broad specification coverage

✓ keep our examinations at a manageable length.

We increased the number of tick box, multiple choice and 'fill in the blank' questions we use in our examinations.

This allows candidates to show knowledge and understanding in a range of accessible response styles.

These types of questions allow candidates to show understanding of both simple and more complex ideas, without necessarily requiring a written response, or multi-line answer.

Examples of short answer questions

Tick box questions

1 Tick (✓) **one** box in each row to identify if each operator is a comparison operator or an arithmetic operator.

Operator	Comparison	Arithmetic
==		
+		
DIV		
>		

[4]

Fill in the blanks

(c) A pilot's flying experience is validated. An algorithm checks that the experience is between 0 and 20 years.

```
exp = input("Enter number of years")
if exp >= 0 and exp <= 20 then
    print(True)
else
    print(False)
endif
```

Complete this test plan for the algorithm.

Experience in years	Type of test	Expected output
	Normal	True
20	Boundary	
32		

[4]

Fill in the blanks

4 Complete the description of computational thinking using the given list of terms.

Not all terms will be used.

abstraction	algorithm	computation	decomposition
evaluation	flowchart	origin	program
pseudocode	research	sequence	thinking

Computational thinking is the process of analysing problems so that they can be solved in a logical way.

The process of ..... breaks down a problem into smaller, more manageable parts.

The process of ..... removes unnecessary detail from the problem, so that the main components can be focused on.

Algorithmic ..... identifies the main steps needed to solve the problem and the ..... that the steps are completed.

[4]

Short answer questions

(c) Identify **three** devices, other than a Sat Nav, which contain embedded systems.

1.....

2.....

3.....

[3]

Medium response questions

Medium response questions may:

- ✓ be 3, 4, 5, or 6 mark questions
- ✓ test any Assessment Objective (AO1, AO2, AO3).

Component 2 Section A will always contain **one** larger tariff, 6 mark question.

This will be marked using a point-based mark scheme. The algorithmic nature of these questions and responses suit this style of mark scheme, rather than a levels of response mark scheme.

Examples of medium response questions

Example 1: Algorithm writing

Here candidates are free to choose how they respond as there is no requirement to use either a high-level programming language or ERL.

(b) The program design is updated to create usernames as follows:

- If the person is male, their username is the last 3 letters of their surname and then the first 2 letters of their first name.
- If the person is female, their username is the first 3 letters of their first name and then the first 2 letters of their surname.

(i) What would be the username for a male called Fred Biscuit using the updated process?

..... **[1]**

(ii) Write an algorithm for the updated program design shown in question **4(b)(i)**.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....**[6]**

Example 2: Algorithm writing

Here the candidates are required to write in a high-level programming language or ERL. This is clearly marked in the question.

(i) Complete the algorithm to:

- Calculate the total pay for the pilot for that day

You must use either:

- OCR Exam Reference Language, or
- A high-level programming language that you have studied.

```
experience = input("Enter years of experience")

miles = input("Enter miles flown")

totalPay = 0

.....

.....

.....

.....

.....

.....

.....

.....

print(totalPay)
```

[4]

Extended response questions

Component 1 will always have **one** extended response question worth 8 marks.

Extended response questions test a candidate's ability to construct and develop a sustained line of reasoning.

Extended response questions use a levels of response mark scheme. Candidates are placed initially in 'bands', and then given a mark within that particular band.

Example

11 * People often want to buy the most up-to-date smartphones, even though the smartphone they own still works.

Discuss the impact of people wanting to upgrade to the latest smartphone.

In your answer you might consider the impact on:

- smartphone users
- cultural issues
- ethical issues
- environmental issues

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... **[8]**

Teacher Tip

These questions will often require candidates to use their knowledge and understanding to evaluate or create discussion based on a context.

Answers should be presented in a logical format.

Meet the team

Meet the team that supports you through your delivery. Learn more about who they are, and how they will be able to help you.

To discover more about our Computer Science team please visit: https://teach.ocr.org.uk/meet-the-subject-team-computing

Need to get in touch?

If you ever have any questions about OCR qualifications or services (including administration, logistics and teaching) please feel free to get in touch with our customer support centre.

Call us on
01223 553998

Alternatively, you can email us on
support@ocr.org.uk

For more information visit
ocr.org.uk/qualifications/resource-finder
ocr.org.uk
facebook.com/ocrexams
twitter.com/ocrexams
instagram.com/ocrexaminations
linkedin.com/company/ocr
youtube.com/ocrexams

We really value your feedback

Click to send us an autogenerated email about this resource. Add comments if you want to. Let us know how we can improve this resource or what else you need. Your email address will not be used or shared for any marketing purposes.

I like this **I dislike this**

Please note – web links are correct at date of publication but other websites may change over time. If you have any problems with a link you may want to navigate to that organisation's website for a direct search.

CAMBRIDGE
UNIVERSITY PRESS & ASSESSMENT