GCSE Examiners' Report

Computer Science

GCSE

Summer 2025

Introduction

Our Principal examiners' report provides valuable feedback on the recent assessment series. It has been written by our Principal Examiners and Principal Moderators after the completion of marking and moderation, and details how candidates have performed in each unit.

This report opens with a summary of candidates' performance, including the assessment objectives/skills/topics/themes being tested, and highlights the characteristics of successful performance and where performance could be improved. It then looks in detail at each unit, pinpointing aspects that proved challenging to some candidates and suggesting some reasons as to why that might be.[1]

The information found in this report provides valuable insight for practitioners to support their teaching and learning activity. We would also encourage practitioners to share this document – in its entirety or in part – with their learners to help with exam preparation, to understand how to avoid pitfalls and to add to their revision toolbox.

Further support

Document	Description	Link
Professional Learning / CPD	WJEC offers an extensive programme of online and face-to-face Professional Learning events. Access interactive feedback, review example candidate responses, gain practical ideas for the classroom and put questions to our dedicated team by registering for one of our events here.	https://www.wjec.co.uk/home/professional-learning/
Past papers	Access the bank of past papers for this qualification, including the most recent assessments. Please note that we do not make past papers available on the public website until 12 months after the examination.	Portal by WJEC or on the WJEC subject page
Grade boundary information	Grade boundaries are the minimum number of marks needed to achieve each grade. For unitised specifications grade boundaries are expressed on a Uniform Mark Scale (UMS). UMS grade boundaries remain the same every year as the range of UMS mark percentages allocated to a particular grade does not change. UMS grade boundaries are published at overall subject and unit level. For linear specifications, a single grade is awarded for the subject, rather than for each unit that contributes towards the overall grade. Grade boundaries are published on results day.	For unitised specifications click here: Results, Grade Boundaries and PRS (wjec.co.uk)

[1] ***Please note that where overall performance on a question/question part was considered good, with no particular areas to highlight, these questions have not been included in the report.***

Exam Results Analysis	WJEC provides information to examination centres via the WJEC Portal. This is restricted to centre staff only. Access is granted to centre staff by the Examinations Officer at the centre.	Portal by WJEC
Classroom Resources	Access our extensive range of FREE classroom resources, including blended learning materials, exam walk-throughs and knowledge organisers to support teaching and learning.	https://resources.wjec.co.uk/
Bank of Professional Learning materials	Access our bank of Professional Learning materials from previous events from our secure website and additional pre-recorded materials available in the public domain.	Portal by WJEC or on the WJEC subject page.
Become an examiner with WJEC.	We are always looking to recruit new examiners or moderators. These opportunities can provide you with valuable insight into the assessment process, enhance your skill set, increase your understanding of your subject and inform your teaching.	Become an Examiner | WJEC

Contents **Page**

Executive Summary

Candidate performance in the Summer 2025 series of GCSE Computer Science assessments demonstrated a generally positive trend across all units, with notable improvements in accessibility and engagement compared to previous years. The assessment effectively tested a broad range of skills and knowledge, with candidates showing particular strength in object-oriented programming and foundational Computer Science concepts.

In Unit 1, candidates responded well to questions on CPU architecture, storage types and number system conversions. The paper was considered more accessible than in 2024, with an increase in the mean total score. However, areas such as environmental impacts of hardware disposal, DNS sequencing and Boolean logic simplification revealed gaps in understanding. While most candidates could identify syntax errors in code, fewer were able to explain logical flaws or provide detailed reasoning for corrections. Responses to cybersecurity and compression topics varied, with misconceptions around lossy and lossless methods persisting.

Unit 2 saw encouraging performance in computational thinking and programming, particularly in Greenfoot-based Java tasks. Candidates demonstrated sound understanding of object-oriented principles and attempt rates across the paper were high. Facility factors indicated that while some questions were well understood, others posed significant challenges, especially those requiring deeper conceptual reasoning. The extended programming task revealed a wide spread of ability, with more able candidates producing well-structured, modular solutions and less able candidates struggling with logic and clarity.

In Unit 3, the NEA component, most candidates successfully developed functional software solutions for the Eisteddfod booking system scenario. Interfaces were generally effective and many candidates implemented validation routines, though usability issues were common. The refinement logs varied in quality, with some candidates providing reflective entries while others relied heavily on copied content. Analysis and design stages were often underdeveloped, and retrospective design work was frequently submitted, which cannot be credited. Testing strategies lacked depth, with limited evidence of logic-based testing and evaluation against objectives. Suggestions for further development were typically brief and lacked technical insight.

Across all units, common themes for improvement include deeper conceptual understanding, more precise technical language and stronger evaluative and reflective skills. Candidates would benefit from clearer guidance on planning, testing and documenting their work, particularly in the NEA. Centres are encouraged to make greater use of annotation and detailed feedback during internal marking to support moderation.

To support future series, WJEC offers a range of resources including blended learning materials, exam walk-throughs and professional learning events. These resources are available via the WJEC Educational Resources portal.

COMPUTER SCIENCE

GCSE

Summer 2025

UNIT 1 – UNDERSTANDING COMPUTER SCIENCE

Overview of the Unit

Unit 1 contributes 50% of the qualification assesses a range of assessment objectives (AO) as follows:

- AO1 – 52 marks
  - Demonstrate knowledge and understanding of the key concepts and principles of computer science

- AO2 – 48 marks
  - Apply knowledge and understanding of key concepts and principles of computer science

During this series, a wide range of content was assessed, including:
- Computer hardware
- Secondary storage
- Truth tables and Boolean algebra
- Networks
- Routing traffic
- Representation of numbers
- Programming errors
- Data structures and security
- Integrated Development Environment (IDE) tools
- Cybersecurity
- Digital storage of text, graphics and sound.

The mean total for Unit 1 this series was approximately 48.5 out of 100 marks. This represents an increase when compared with the mean total of 39.7 out of 100 in 2024, with candidates finding the paper more accessible.

Comments on individual questions/sections

Q.1 Most candidates correctly matched the CPU components to their descriptions. Many were able to state the purpose of the program counter, while only a minority named two additional CPU registers accurately. Descriptions of RISC versus CISC were often superficial, with fewer candidates providing two clear differences. The definition of an embedded system was generally well answered, though environmental impacts of hardware disposal attracted only brief, generic responses.

Q.2 The true/false comparison of solid-state and magnetic storage was well handled, with over half of candidates selecting all three correct statements. Descriptions of optical storage characteristics varied in depth; while several candidates could describe read/write processes, few discussed capacity or durability in detail.

Q.3 Around half of candidates completed the truth table correctly and applied Boolean simplification accurately, often gaining full marks or none. The main pitfalls were misunderstanding the XOR operator and omitting the NOT operation in two cells.

Q.4 Tick-box identification of TCP/IP layers was straightforward for most, but completion of protocol names (e.g. HTTP, SMTP) saw many spelling errors. Labelling of packet contents was generally accurate. The DNS sequence question was poorly answered; few described the sequence correctly.

Q.5 This routing question was the best answered on the paper. Most candidates determined the lowest-cost paths correctly, although a small number mis-computed cumulative costs on longer routes.

Q.6 Conversions between number systems were handled competently, with most scoring full marks on denary and hexadecimal calculations. Identification of overflow in 4-bit addition was well answered, but explanations of why overflow occurs often lacked reference to bit-width limitations. Descriptions of division by four via bit shifts were mostly correct, though some candidates mixed up shifts left with shifts right.

Q.7 Candidates were asked to identify and correct both syntax and logical errors in a short code snippet. While most spotted basic syntax mistakes (mis-spelled variables), far fewer recognised subtler logic flaws - such as errors in calculations. Many gave only the corrected code lines without explaining why each change was needed.

Q.8 Candidates generally found the data-structure table straightforward, choosing field names, data types and validation methods appropriately. Where marks were lost, it was due to mismatches (e.g. text validation applied to numeric fields). Security methods – access levels and XOR encryption – were described in broad terms, but detailed examples were rare.

Q.9 IDE tool identification scored well overall. Around half of candidates named three distinct facilities (such as breakpoints, code completion, debugging consoles) with accurate uses. The remainder often repeated similar tools.

Q.10 Definitions of spyware, brute-force and Trojans were generally correct when using key terms. Penetration-testing methods (targeted and blind testing) attracted limited detail, with most candidates confusing the two approaches.

Q.11 Explanations of how text, graphics and sound are stored varied widely in technical accuracy. Compression discussions often mentioned formats such as ZIP and JPEG, but candidates rarely contrasted lossless and lossy methods correctly. In particular, far too many stated that only lossy compression can reduce quality, implying that lossless never impacts quality - which overlooks cases where metadata stripping or algorithmic limits can still alter fidelity. Answers should explicitly note that lossy methods intentionally discard data, whereas lossless schemes preserve all original data, but may still affect perceived quality.

COMPUTER SCIENCE

GCSE

Summer 2025

UNIT 2 COMPUTATIONAL THINKING AND PROGRAMMING

Overview of the Unit

Unit 2 of the GCSE Computer Science course assesses the following objectives:

AO1 – Demonstrate knowledge and understanding of the key concepts and principles of Computer Science.

AO2 – Apply knowledge and understanding of key concepts and principles of Computer Science.

AO3 – Analyse problems in computational terms to make reasoned judgements and to design, program, evaluate and refine solutions.

It was pleasing to see that the Greenfoot programming work using the Java language was of a good standard, with many learners demonstrating a sound understanding of object-oriented programming concepts such as classes, objects, inheritance and methods. Overall, performance across the unit was encouraging, with learners generally able to apply their knowledge effectively in practical programming tasks. For future teaching, continued emphasis on object-oriented principles and their application in real-world contexts will support learners in developing robust programming skills and a deeper understanding of computational thinking.

Comments on individual questions/sections

General Trends and Patterns
Facility Factors (FF) ranged from 34.2% to 78.6%, indicating a wide variation in question demand for candidates.
Highest FF: Question 4(a) (78.6%) – suggesting it was well understood and accessible.
Lowest FF: Question 2 (34.2%) – indicating it was the most challenging for candidates.

Attempt rates were generally high (above 94% for most questions), showing good engagement across the paper and notably that the higher demand questions were also well attempted indicating good candidate familiarity with question types despite certain topics demand of difficulty.

Question-Level Insights

Q1 (FF: 69.8%)
High facility factor and attempt rate (97.9%) suggest this was a well-answered, accessible question where candidates could apply their knowledge of markup languages.
This tested foundational knowledge or straightforward application.

Q2 (FF: 34.2%)
Lowest FF despite a high attempt rate (96.4%).
Indicative of the conceptual or interpretive challenge; candidates may have misunderstood the requirements or lacked sufficient depth of knowledge in the topic.

Q3 (FF: 58.3%)
Moderate performance; suggests a mix of partial and full responses.
Candidates may have had varied success depending on how they approached the question.

Q4(a) (FF: 78.6%)
Strongest performance overall.
Indicates clear understanding and confident application of the topic assessed.

Q4(b) (FF: 48.4%)
Lower FF and attempt rate (89.6%) suggest this sub-question was more demanding or less well understood. May have required more precise knowledge or reasoning.

Q5 (FF: 63.2%)
Solid performance: suggests candidates were generally comfortable with the content.
Could reflect effective teaching of this topic area.

Q2 (12-mark question on programming an interactive computer game, FF: 52.5%)
Moderate FF and high SD (4.7) indicate a wide spread of performance.
Suggests good differentiation in candidate ability or approach to extended response.

COMPUTER SCIENCE

GCSE

Summer 2025

UNIT 3 NEA – SOFTWARE DEVELOPMENT

Overview of the Unit

This unit requires learners to produce a programmed solution to a problem. They must analyse the problem, design a solution to the problem, develop a final programmed solution, test the solution and give suggestions for further development of the solution. Throughout the production of the solution learners are required to produce a refinement log that evidences the development of the solution.
Examples of good work were seen during moderation this summer. It should be noted that it is essential that candidates undertake the correct scenario for the current series.

This unit represents 20% of the qualification and comprises of the below AOs:

- AO2 - 2%
  Apply knowledge and understanding of key concepts and principles of computer science.
- AO3 - 18%
  Analyse problems in computational terms to make reasoned judgements and to design, program, evaluate and refine solutions.

Tasks
Comments on tasks/questions relating to candidate performance/meeting assessment criteria

The scenario has a bullet pointed list of requirements. To access full marks for the implementation of the solution to the given problem, all bullet points should be covered. However, many candidates were not able to produce a solution that covered all bullet points of the scenario.

The scenario was based on the creation of an order and invoice system for the Eisteddfod. The brief required candidates to create software that allowed for the booking of stalls at the festival as well as other requirements, i.e.:

- Enter and store vendor details.
- Enter and store details of the vendor's booking
  - choose a stall size (small, medium, large)
  - choose whether the stall is indoors (only small stalls are available indoors) or outdoors
  - choose the duration of the booking (up to 6 days)
  - choose whether the vendor requires electricity and/or water at their stall
- Calculate the total cost of the stall booking including any additional costs for electricity and water.
- Provide an invoice for the vendor.
- Search vendor details by booking number.
- Produce a list of all stalls for the site team to build the stalls for the festival.

Most candidates could create an interface that allows users to enter the vendor's details when booking the stalls. Many candidates were able to produce code that could carry out the required functions. Candidates also created validation routines. However, these routines often caused issues when running the code as the interfaces did not provide clear enough instructions for the end user.

Task marking
Comments on approaches to internal marking

Most centres made good use of the centre comments section of the candidate declaration forms, however, this could be further improved with more detailed comments. Several centres include the band marks on a per candidate basis that have been highlighted to demonstrate where the candidate has obtained the marks. Very few centres add annotation to the candidates' work.

Requirements for the Report:
The specification states that the candidates should produce a report that:
- Analyses the given information.
- Includes a design of a solution to the given problem.
- Programming of a solution to the given problem.
- Testing and refinement of the application, noting the refinements in the refinement log.
- Gives an evaluation of the application.

Refinement log

Candidates are required to complete a log of their activities during the twenty-hour controlled assignment. Almost all candidates presented a completed log. However, a significant number of candidates submitted logs that included many copied and pasted entries. Where candidates had made effective use of their logs, entries included discussion of problems encountered and solutions to these problems. Many candidates were able to identify action points for the following sessions that would enable them to make more effective use of their time.

Scope of the problem

A minority of candidates presented effective analysis of the given scenario while many either restated the problem or copied and pasted the contents of the brief. Most candidates were able to outline the objectives for their solution to the given problem.

Design of Solution

In a significant minority of instances, candidates were neither able to justify their choice of programming language using appropriate technical terminology nor relate the features and facilities of the language to their proposed method of solution. Many candidates were able to describe some of the process stages required for their solutions in pseudo code and/or flowcharts. However, fewer candidates covered all processing stages for their proposed solution. In several cases it was not clear that this work had been completed before implementation. Retrospective designs will not be given any credit at moderation.

Effectiveness of solution

Most candidates had produced a solution that allowed the user to enter the details of the businesses and customers into the system. A large percentage of candidates developed solutions using the Python programming language with an increasing number making use of SQLite3, while others produced solutions using a visual programming language. The quality of the interfaces produced varied considerably. Centres are reminded that the use of online software development tools is prohibited and that 'drag and drop' block type IDEs is not adequate. Most solutions were modular and included the required authentication routines.

Technical quality

Many candidates produced code that was self-documenting and there were instances of the code being well structured. Where candidates had a good understanding of the language they were using, there was evidence of the use of consistent style throughout including indentation and use of white space. In general candidates produced code that used meaningful identifiers and appropriate constants and had coded some validation routines.

Many candidates included some annotation of their code with more able candidates included annotation that demonstrated their understanding of the problem and solution.

Test strategy

Most candidates were able to describe some type of testing strategy and some evaluation criteria. In future candidates may benefit from considering their objectives when describing their testing strategies and evaluation criteria, ensuring that they plan to test and evaluate against each objective.

Testing

Most candidates were able to design tests that would demonstrate the functioning of parts of their solution. However, some test plans would have benefited from focusing on the logic of the solution rather than repeatedly testing the less complex parts of the system.
In future, candidates would benefit from using their objectives and success criteria as a framework for their test plans and ensure that these are met by their solution.

As solutions should be able to carry out particular functions, candidates should ensure that the data entered produces the correct result and the output is correctly formatted. The scenarios provided many opportunities for candidates to test the logic of their solutions using text-based data to produce a mathematical result to their calculations.

All tests from the test strategy should be evidenced with screenshots and discussed rather than stating if a particular test was a pass or fail. This aspect needs to be improved upon by candidates and centres.

Further development

The specification calls for candidates to:

- Consider the outcomes of the testing process in terms of how well the application meets the objectives set at the beginning of the project.
- Describe the good features of the application and identify areas for further development.
- Provide detailed suggestions for specific extensions to the application.

Many of the discussions produced by the candidates were brief and tended to be narrative rather than reflective and evaluative in nature.

Few candidates offered valid and detailed suggestions for future improvements. However, a minority were able to discuss their solutions in the light of their structure and suggest viable improvements that could be created using their chosen language.

Supporting you

Useful contacts and links

Our friendly subject team is on hand to support you between 8.30am and 5.00pm, Monday to Friday.
Tel: 02920 265 401
Email: CS@wjec.co.uk
Qualification webpage: GCSE Computer Science

See other useful contacts here: Useful Contacts | WJEC

CPD Training / Professional Learning

Access our popular, free online CPD/PL courses to receive exam feedback and put questions to our subject team, and attend one of our face-to-face events, focused on enhancing teaching and learning, providing practical classroom ideas and developing understanding of marking and assessment.

Please find details for all our courses here: https://www.wjec.co.uk/home/professional-learning/

WJEC Qualifications

As Wales' largest awarding body, WJEC supports its education community by providing trusted bilingual qualifications, specialist support, and reliable assessment to schools and colleges across the country. This allows our learners to reach their full potential.

With more than 70 years' experience, we are also amongst the leading providers in both England and Northern Ireland.

WJEC
245 Western Avenue
Cardiff CF5 2YX
Tel No 029 2026 5000
Fax 029 2057 5994
E-mail: exams@wjec.co.uk
website: www.wjec.co.uk