GCE Examiners' Report

Computer Science

GCE

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

Executive Summary

Candidate performance across the GCE Computer Science qualification in Summer 2025 revealed a mixed profile, with strengths in procedural and structured tasks but notable challenges in abstract and theoretical areas. While engagement remained high across all units, it appears that candidates often struggled with depth of understanding and articulation in extended responses.

In Unit 1, candidates demonstrated a strong understanding of binary operations, Boolean logic and structured data tasks. However, performance dropped significantly in questions requiring comparative analysis or theoretical depth, such as operating system interfaces and database structures. Candidates struggled particularly with abstract concepts like routing, compilation stages, and variable scope. This suggests a need for greater emphasis on conceptual clarity, the ability to apply knowledge in unfamiliar contexts and more practice with extended written responses.

Unit 2 showed encouraging signs in practical programming, with Python remaining the dominant language and a notable rise in Java usage. Candidates generally showed sound problem-solving and design skills, though some submissions lacked sufficient annotation or failed to meet functional requirements. Centres are reminded of the importance submitting fully working solutions, ensuring that annotation supports the assessment objectives and ensuring that learners apply their knowledge to the given problem context rather than relying on recall alone.

Unit 3 saw strong performance in data structures, encryption and truth tables, with candidates confidently handling stack and queue operations and XOR-based tasks. However, questions involving CASE tools, algorithm efficiency and programming paradigms were less well answered. Many candidates struggled with articulating the purpose and application of these concepts, indicating a need for more targeted support in analytical writing and system-level thinking.

In Unit 4, candidates performed well in binary arithmetic, floating point conversions and SQL commands. However, descriptive questions – particularly those involving operating systems, data mining and biometric data – were often answered superficially. Candidates frequently failed to provide sufficient reasoning or examples, which limited access to higher mark bands. Greater emphasis on technical vocabulary and structured responses would benefit future cohorts.

The NEA (Unit 5) continued to offer candidates the opportunity to showcase their creativity and technical skills. High-performing candidates produced well-researched, thoroughly documented and effectively implemented solutions. However, some projects lacked alignment between investigation and final implementation, and others failed to demonstrate sufficient refinement or evaluation. Centres that advocate iterative development, used video evidence and encouraged critical reflection enabled candidates to access higher mark bands.

Common themes across all units include a need for improved depth in written responses, better understanding of abstract concepts and more consistent application of theoretical knowledge. To support future performance, centres are encouraged to make use of WJEC's professional learning events, past papers and blended learning resources, particularly in areas such as system development, database theory and algorithm design.

COMPUTER SCIENCE

GCE

Summer 2025

UNIT 1 – FUNDAMENTALS OF COMPUTER SCIENCE

Overview of the Unit

Unit 1 contributes 62.5% of the AS Level qualification and 25% of the A Level. The unit assesses a range of assessment objectives (AO) as follows:

- **AO1**
  - Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.
- **AO2**
  - Apply knowledge and understanding of the principles and concepts of computer science, including to analyse problems in computational terms.
- **AO3**
  - Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

During this series, a wide range of content was assessed, including:

- Input Devices
- Boolean Expressions and Truth Tables
- Parallel and Serial Transmission
- Network Collisions
- Routing
- Binary Addition and Conversion
- Truncation, Rounding, and Error Calculation
- Boolean Simplification
- File Design and Validation
- Types of operating systems
- Relational vs Flat File Databases
- Operating System Interfaces
- Variables and Parameters
- Scope and Lifetime of Variables
- Algorithmic Design
- Alpha vs Beta Testing
- IDE Debugging Tools
- Compilation Process

The mean total for Unit 1 this series was approximately 42.7 out of 100 marks. This represents a decrease when compared with the mean total of 46.5 out of 100 in 2024, with candidates finding the paper more demanding overall.

Comments on individual questions/sections

Q.1 This question asked candidates to describe input devices for different applications alongside advantages and disadvantages. With a facility factor of 57.1% and a very high attempt rate of 99.4%, it was moderately well answered. This suggests candidates generally understood input devices, though some may have struggled to contextualise them within varied applications and evaluate their practical benefits and limitations.

Q.2 This involved completing Boolean expressions in a truth table. The facility factor was a high 77.3%, with nearly universal attempts at 99.4%. This indicates confident handling of logic-based tasks and good familiarity with the structure and syntax required.

Q.3 This question was multipart, addressing transmission methods, network collision handling, and packet routing.

(a) Showed a low facility factor of 33.2%, despite 99.2% attempting it, suggesting difficulties in comparing parallel and serial transmission.

(b) Scored a stronger 66.4% and 99.1% attempted it, reflecting better understanding of collision responses on a bus network.

(c) Fell to just 29.5% with a slightly lower attempt rate of 95.2%, indicating that the routing of a packet has not been fully grasped. Many candidates included a breakdown of the parts of a packet instead of the process of routing across a network.

Q.4 This question focused on binary addition and error analysis.

(a) Showed strong performance with an 80.3% facility factor and 99% attempted, demonstrating solid procedural understanding. On occasions the addition marks were awarded but the conversion to denary was not correct.

(b) Scored only 31.3%, despite 95.6% engagement. Most were able to successfully complete the rounding and truncation element of this question, however there was a greater level of inaccuracy when calculating the relative and absolute errors, which would account for the mean average being 1.9. Candidates found the theoretical concepts error calculation less intuitive or underdeveloped.

Q.5 This asked for Boolean expression simplification using logic rules. With a 69.7% facility factor and a 98.7% attempt rate, most candidates responded well. Errors arose in application of Boolean identities or missteps in the simplification sequence, but overall familiarity with formal logic contributed to success.

Q.6 Here candidates explored file design and database structures in a payroll system.

(a) Performed strongly with a facility factor of 74% and 99.4% attempted. Candidates were comfortable with structured data tasks such as defining key fields and applying validation techniques. Some candidates lost marks for not applying effective field names and ensuring they were carried through to the second table.

(b) Dropped significantly to 41%, with only 82.3% attempting. This could be due to unfamiliar inability to effectively recognise the appropriate type of operation for payroll systems.

(c) Was a low scoring question with a facility factor of 26.8% despite a 95.2% attempt rate. This indicates candidates found relational database benefits difficult to articulate, especially in contrast to flat file systems.

Q.7 Asked candidates to discuss how an operating system provides user-hardware interface. This had the lowest facility factor overall at just 18%. Despite a 97.2% attempt rate, responses were weak due to the breadth and depth required. Many candidates did not fully comprehend the question and its requirements. Those who were awarded marks generally only considered Command Line Interface and Graphical User Interface operating systems.

Q.8 This question explored programming constructs.

(a) Was poorly answered, scoring 27.8%, although 97.9% attempted it. Candidates may have misunderstood and were unable to describe the terms variables and parameters to demonstrate more than one or two points.

(b) Scored marginally better at 41.2%, but with lower engagement at 90.6%. Many candidates were able to identify both global and local variables and discuss their lifetime, but they did not access the additional marks which required discussion of scope and lifetime of variables in general terms.

Q.9 This required candidates to write a pseudo-code algorithm with validation. It achieved a moderate facility factor of 51.8% with 94.3% attempt rate. Candidates generally managed to successfully declare and assign variables as well as accept an input. Some were able to include a form of validation on the inputted value, many struggled with the binary conversion aspect.

Q.10 Compared alpha and beta testing, saw a facility factor of 41.8% and 96.3% attempted. This result implies partial understanding, with candidates possibly confusing the contexts, objectives or examples associated with each testing phase.

Q.11 This question asked about IDE debugging tools. The facility factor was 36.7% with 96% attempted. Many candidates considered all aspects off an IDE instead of focusing specifically on the debugging tools that are available in IDEs.

Q.12 This focused on compilation stages in relation to a flawed program, had a low facility factor of 27.6% and 92% attempt rate. While many candidates might recall the names of compilation phases, fewer could effectively link them to the specific context of debugging a program with errors. This resulted in them being unable to access the higher mark bands of this question.

Overall Observations reveal that structured, procedural tasks such as binary addition and Boolean simplification saw higher facility factors, while abstract, comparative or essay-style responses, such as those involving system development, operating systems or databases, had lower performance, often despite high engagement. Some drop-off in attempt rate appears in questions involving more technical jargon or less familiar operational terms.
For improved outcomes, it would be beneficial to provide targeted support in abstract and theoretical areas, such as database comparisons, compilation processes and system-level programming concepts.

COMPUTER SCIENCE

GCE

Summer 2025

UNIT 2 – PRACTICAL PROGRAMMING TO SOLVE PROBLEMS

Overview of the Unit

Unit 2 continues to provide a valuable insight into learners' ability to apply programming skills in a practical context. It was pleasing to see a broader range of programming languages being used this series, with a notable increase in the use of Java, which suggests a growing confidence in object-oriented programming paradigms. However, Python remains the most widely used language, likely due to its accessibility and strong support resources. Overall, candidate performance was encouraging, with many demonstrating sound problem-solving strategies, effective planning and design skills, and appropriate testing techniques. That said, some submissions lacked sufficient annotation or evidence application (AO2) of knowledge to given problems, which limited the marks awarded. Centres are encouraged to continue supporting learners in applying their knowledge to given scenarios and not purely relying on basic recall (AO1) skills.

Comments on individual questions/sections

Part A: Problem Decomposition and Algorithm Design

Most candidates attempted to break down the problem into manageable components and demonstrated a good understanding of ER diagrams and practical programming.

Stronger candidates used structured pseudocode or flowcharts, while others jumped straight into implementation, often leading to disorganised solutions that did not match the program brief.

Part B: Practical Programming

Python was the dominant language, with increasing use of Java. Most candidates demonstrated familiarity with basic constructs (loops, conditionals, functions).

Annotation was generally well presented with candidates annotating a good range of their code. Candidates are encouraged to annotate their entire programs when asked to annotate their code.

Candidates using Python often relied on built-in functions, while Java users tended to write more verbose but structured code. There was sadly some programming provided that did not function and would not run. Centres are reminded that candidates need to ensure they provide fully working solutions that can be loaded and run without modification by the examiner. Broken solution files were an issue once again in Visual Basic and Java based centres.

COMPUTER SCIENCE

GCE

Summer 2025

UNIT 3 – PROGRAMMING AND SYSTEM DEVELOPMENT

Overview of the Unit

Unit 3 contributes 20% of the A Level qualification and assesses a range of assessment objectives (AO) as follows:

- **AO1**
  - Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.
- **AO2**
  - Apply knowledge and understanding of the principles and concepts of computer science, including to analyse problems in computational terms.
- **AO3**
  - Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

During this series, a wide range of content was assessed, including:

- Queue and stack data structures
- Linked list manipulation
- Truth tables and Boolean simplification
- Logical operations and XOR encryption/decryption
- CASE tools in system development
- Waterfall methodology and documentation
- Insertion sort and algorithm design
- Algorithm efficiency and Big O notation
- Backus-Naur Form (BNF) for email validation
- Data compression types and calculations
- Code of conduct and employee monitoring
- Binary tree construction and traversal
- Programming paradigms

The mean total for Unit 3 this series was approximately 37.2 out of 100 marks. This represents a decrease when compared with the mean total of 39.3 out of 100 in 2024, with candidates finding the paper slightly more demanding overall.

Comments on individual questions/sections

Q.1 (a) This question focused on describing queue data structures with contextual examples. With a facility factor of 70.6% and a very high attempt rate of 99.1%, it was reasonably well answered. This suggests that candidates were broadly familiar with linear data structures and could relate queues to practical usage, though deeper explanation or varied examples were lacking in some instances.

(b) (i) Candidates were asked to illustrate a stack with six entries. The facility factor was very high at 95.3%, with 99.1% attempting. Responses indicate strong understanding of the LIFO structure and candidates handled the diagrammatic representation with confidence.

(ii) This part required manipulation of the stack after popping and pushing operations. Facility factor was also high at 85.4%, with 98.3% attempting. Most candidates correctly followed stack rules, although some made minor errors in diagram layout or order resulting in lower marks awarded.

Q.2 (a) This required redrawing a linked list after deletion. The facility factor was 91.3% with 99.5% attempts, showing candidates were confident in editing node pointers and maintaining list integrity.

(b) Extending the linked list with an added node saw an even stronger facility factor of 93.4%. The near-universal attempt rate (99.3%) further supports the view that linked lists are well understood and taught effectively.

Q.3 (a) Candidates completed a truth table. With a facility factor of 78.6% and 98.3% attempt rate, this was moderately well answered. The Boolean expression's structure was accessible, but some candidates struggled with combining logical operations correctly.

(b) (i) This required simplification of a Boolean expression. The facility factor stood at 78.8%, with 97.9% attempts. Most candidates used the identity laws appropriately, although a few misapplied simplification steps.

(ii) The second expression proved more challenging. Facility factor was just 69.1%, with 97.7% attempts. Errors tended to stem from operator precedence confusion and redundancy in simplification along with not effectively applying De Morgan's law.

Q.4 (a) Candidates had to identify a logical operation from a truth table. With a low facility factor of 61.5% but a strong attempt rate of 98.8%, responses were mixed. Many correctly matched the output pattern, but some failed to fully justify the operation chosen.

(b) (i) This encryption task using XOR achieved a facility factor of 89.7% and 96% attempts. Strong scores indicate candidates were confident with bitwise manipulation and encryption steps.

(ii) Decryption using XOR followed similarly, with a facility factor of 87.1% and 92% attempts. High scores suggest consistent understanding of reversible encryption.

Q.5 This question asked candidates to discuss CASE tools in system design. With a facility factor of 41.7% and 94.3% attempt rate, it was challenging. Candidates struggled to move beyond generalisations and often omitted key features such as automated diagram generation or version control.

Q.6 **(a)** Documentation benefits in Waterfall development were explored. A facility factor of 47.8% and 96.9% attempt rate indicates mixed responses. Many referenced clarity and traceability, but fewer articulated stage-specific relevance or long-term advantages.

(b) Maintenance stage documentation had a facility factor of 29.5% and 91.2% attempt rate. Responses were often vague, lacking discussion of the various documentation associated with the maintenance stage.

Q.7 **(a)** Insertion sort suitability was poorly understood, scoring a facility factor of just 28.1% with 91.7% attempts. Many failed to distinguish its comparative advantage for small or nearly sorted datasets.

(b) The algorithm writing task achieved a facility factor of 34.9%, with only 82.5% attempting. Candidates often misstructured pseudo-code or omitted key iterative details, resulting in a lower mean score.

Q.8 **(a)** **(i)** Efficiency evaluation via Big O scored 51.2%, with 92.5% attempts. While many identified linear, the responses often lacked depth, with a mean response of 1.5 out of a possible 3 marks.

(ii) Time growth rate determination saw weaker performance: facility factor of 46.2%, with only 86.1% attempts. Candidates appeared less confident translating performance observations into formal notation.

(b) Graphical illustration scored moderately at 65.7%, though only 89.8% attempted. Many candidates could label axes accurately but failed accurately sketch trends.

Q.9 Candidates wrote a Backus-Naur Form to define email syntax. The facility factor was 47% with a 94.1% attempt rate. Most responses included structure, though uppercase rules and domain differentiation were often misapplied or missed.

Q.10 **(a)** This part compared lossless and lossy compression. With a facility factor of 70% and 95.7% attempts, candidates generally distinguished both types, though examples and implications were sometimes superficial.

(b) **(i)** Compression ratio calculation scored 60.5%, with 97.6% attempts. Most could use the correct formula.

(ii) Saving percentage reached 69.2% with a 96.3% attempt rate. Errors were typically mathematical rather than conceptual.

(iii) Compression time calculation performed worse, with a facility factor of 49% and 97.7% attempts. Candidates misapplied the rate per byte or misunderstood unit conversions.

Q.11 **(a)** Code of conduct content scored a low 38.4%, with a 94.6% attempt rate. Many answers remained general, omitting specifics like legal expectations, professional integrity or equality.

(b) Monitoring techniques had a facility factor of just 30.7%, with 91% attempting. While some named software and network-level checks, fewer linked them to policy enforcement or staff accountability.

Q.12 **(a)** Binary tree construction had a facility factor of 66.1% and 97.7% attempt rate. Candidates generally inserted nodes correctly but occasionally misapplied left-right pointer rules.

(b) In-order traversal scored 51%, with a 95% attempt rate. Many completed the sequence, but mis-ordered nodes.

(c) Pre-order traversal saw poorer results: 35.2% facility and 94.1% attempted. Misalignment with traversal definition caused errors.

(d) Post-order traversal had a similar outcome, with a facility factor of 35.9% and 93.9% attempts. Sequence logic was frequently misapplied.

Q.13 This final question discussed programming paradigms. Despite being attempted by 95.3%, facility factor was just 41.8%. Candidates often repeated definitions without exploring practical application contexts, especially for specific-purpose languages.

Overall Observations: Performance was strongest where candidates dealt with structured procedural tasks, such as stack and queue diagrams, encryption methods and truth tables. More abstract or multi-stage reasoning task, especially those involving CASE tools, monitoring and programming paradigms were weaker.

Despite high engagement rates across the board, lower facility factors in long-form responses suggest candidates struggled with depth, scope and articulation. Future support should focus on improving conceptual clarity, analytical writing and multi-part logic reasoning.

COMPUTER SCIENCE

GCE

Summer 2025

UNIT 4 – COMPUTER ARCHITECTURE, DATA, COMMUNICATION AND APPLICATIONS

Overview of the Unit

The unit 4 examination is well established and produces consistent outcomes which result in stable grade boundaries. The examination addresses each of the three Assessment Objectives (AOs) described in the specification, as follows: -

	Description	Weighting
AO1	Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.	50/100
AO2	Apply knowledge and understanding of the principles and concepts of computer science, including to analyse problems in computational terms.	38/100
AO3	Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.	12/100

The balance between the AOs is set and adhered to, which produces a structure for a consistent assessment.

For this series the majority of candidates were able to demonstrate a wide knowledge of the topics in the specification, as required at A2 level, with questions requiring precise responses to programming or mathematical problems generally producing high marks.

As in previous series the questions requiring more descriptive answers often produced responses that were correct but lacked adequate reasoning or detail to gain high marks. For these types of questions, it is suggested that candidates should concentrate on the technical aspects of the topic and include examples to illustrate and help structure their responses.

Comments on individual questions/sections

Example AO1 Questions

Q.1 Limiting factors to parallelisation. Mean mark of 1.7 out of 6 with a facility factor of 29.1.

It was thought that this would be a familiar topic, but the question did not provide the intended accessible start to the paper, with many candidates stating two of the required three factors but losing marks by omitting to provide any associated descriptions.

Q.4 **(a)** Types of operating system. Mean mark of 1.6 out of 4 with a facility factor of 39.5.

A topic which proved to be more challenging than expected, with many candidates describing networking rather than explaining the terms ‘multi-access’ and ‘multi-tasking’, as required.

(b) Partitioning of main memory. Mean mark 1.0 out of 4 with a facility factor of 16.7.

Candidates also found this to be a challenging topic, with many confusing main memory with secondary storage and basing a response on file organisation.

Q.6 Data mining and predictive analytics. Mean mark of 1.9 out of 6 with a facility factor of 32.2.

Most candidates referred to the investigation of ‘big data’, but few were able to explain the concept of data mining and analytics – “the discovery of hidden patterns in data which leads to the analytical interpretation of the discovered relationships to help predict trends and support decision making”

Q.10 **(a)** DBMS. Mean mark of 1.5 out of 6 with a facility factor of 24.2.

Many candidates were able to identify tasks such as making backups and setting access levels, but as with other AO1 questions, few gained the marks available for providing detailed descriptions of the tasks identified.

Q.12 **(b)** Biometric data. Mean mark of 2.9 out of 8 with a facility factor of 36.8.

An open topic within the context of benefits and concerns arising from the collection, storage and uses of biometric data. The question was band marked with a range of points identified in the indicative content of the mark scheme. Most candidates were able to describe uses of biometrics, but few presented a balanced discussion weighing the benefits of the identified uses against the associated concerns arising from possible errors, data security or mass surveillance.

Example AO2 Questions

Q.2 Binary arithmetic. Mean mark of 6.3 out of 10 with a facility factor of 60.1.

Most candidates were able to convert denary numbers into binary, calculate the total correctly and produce a correct representation of a negative binary number in twos complement. with many examples of full marks seen for these items.

Q.3 Floating point numbers. Mean mark of 5.3 out of 9 with a facility factor of 59.5.

The parts of the question which involved converting floating point numbers were well done, again with many examples of full marks. Candidates tended to lose marks when addressing the part of the question which involved a comparison between truncating and rounding.

Q.8 Database design. Mean mark of 5.0 out of 11 with a facility factor of 45.7.

Most candidates were able to interpret the given scenario and produce reasonable entity relationships. Data tables also tended to be well done, although the allocation of foreign keys caused some difficulties, particularly where ERDs had not been well developed.

AO3 Questions

Q.5 Assembly language. Mean mark of 2.1 out of 4 with a facility factor of 53.1.

Many candidates were able to interpret the given scenario and successfully translate the logic of the given algorithm into assembly code, using the instruction set provided.

Q.7 SQL commands. Mean mark of 4.8 out of 8 with a facility factor of 60.3.

This is a well taught and practised topic, and most candidates were able to produce at least two of the three required commands correctly.

UNIT 5 NEA – PROGRAMMED SOLUTION TO A PROBLEM

Overview of the Unit

This unit requires the learners to investigate, design, prototype, implement, test and evaluate a computer solution to a substantial problem of their own choice. The learner's chosen problem must provide sufficient scope for them to access the marks available for each section of the work.

Learners need to investigate their chosen problems in sufficient detail to identify how data is collected, processed and output currently. The current system may be either paper-based or electronic.

Following the identification of their problems, learners should prepare sufficient documentation to allow them to take part effectively in the discussion with their teachers and/or peers.

Notionally this task will require 72 guided learning hours, which includes teaching time.

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

Discussion:

This section offers candidates the opportunity to present their proposed problem to their teacher, peers or other competent third parties. Candidates should receive detailed and informed feedback regarding the scope and suitability of their chosen problem and reflect deeply on this input. Reflection should result in refinements to their ideas, ensuring unsuitable topics are revised or replaced. Many centres have supported this process effectively, with candidates often enhancing the depth of their proposals or where appropriate, discarding their original ideas in favour of more viable ones—an approach that is highly effective and commendable.

Investigation:

Many candidates selected original problems that did not involve existing end users. To access the higher mark bands, candidates in this situation should research similar commercial systems, identifying common characteristics and features. A comprehensive working specification—complete with measurable objectives—should follow the research. However, in some cases, there is a disconnect between the investigation and the final specification, with candidates introducing new ideas not supported by their research.

Design:

The design phase allows candidates to plan and develop the technical aspects of their functional solution. This stage should be forward-thinking and include:

- Detailed screen layout designs
- Consideration of all required inputs and outputs
- Evidence of data structures and methods of access
- Consideration of appropriate data validation
- Program routine designs using recognised conventions (e.g., pseudocode or flowcharts)

Candidates should ensure that all elements of their design are relevant and clearly linked to their specific project. In some cases, candidates included generic descriptions (e.g., validation types or sorting algorithms) without applying them to their solution, which reduces their value. Most centres, however, provided excellent, detailed designs that could feasibly be implemented by a third party.

Prototype:

At this stage, candidates should demonstrate a partially complete system, self-reflect on progress and identify any changes in strategy required. The prototype should highlight key components of the system in development, with justification for the sections chosen. Many candidates effectively used screenshots or video evidence to show their prototype in action, video being particularly valuable for demonstrating system functionality. Submitting screenshots of code alone is discouraged, as it is difficult to ascertain the functionality of the submitted code.

Centres are encouraged to submit programming code that the moderator can run as evidence for this section.

Post prototype Refinement of Design:

Following third-party feedback, candidates should refine their design and provide justifications for accepting or rejecting suggestions. This section must focus on changes to the **design** and not include completed code. Centres that show "before and after" comparisons of designs provide strong evidence here. However, the inclusion of final programmed changes is inappropriate in this section and will not gain any credit, this should instead be assessed under Software Development.

Software Development:

Some centres have submitted outstanding work, showcasing a wide range of creative and effective solutions. To access the highest mark bands, candidates should:

- Fully exploit the programming language, using advanced features and normalised data structures (to 3NF)
- Demonstrate effective searching and sorting techniques
- Implement a system that meets all defined objectives and user requirements
- Include extensive input validation and error handling
- Use self-documenting identifiers and detailed annotations, making the code easy to follow for third parties

In some cases, code was difficult to execute due to missing packages or libraries. In such instances, candidates must include all necessary files or provide clear instructions in a README file.

Centres that included video demonstrations of the completed systems helped streamline the moderation process.

Developmental Testing:

Most centres presented detailed developmental testing throughout the software development process. Candidates showcased their problem-solving and decision-making skills effectively. However, in some cases, this testing was either absent or incorrectly merged with final testing. Developmental testing should be ongoing and clearly separate from final testing evidence.

Testing:

Centres that included well-structured test tables (with clearly identified **typical**, **extreme**, and **invalid** test cases) provided strong evidence. However, some candidates misunderstood **extreme testing**, which should involve inputs at the boundaries of allowable values—not beyond them.

Many centres supported this section with video evidence linked to each test. Where multiple tests were shown in a single video, timestamps were useful in guiding moderators to relevant sections.

Testing should focus on the functionality of the system in terms of:

- Input mechanisms, including validation for reasonable data entry
- Processing routines, ensuring correct and accurate output
- Output presentation, both on-screen and paper-based

Each test should be traceable to a specific objective. Candidates should provide thoughtful commentary and suggest refinements where test results do not meet expectations.

Evaluation:

It is encouraging to see many centres accurately assess evaluations. High-performing candidates provide detailed, reflective analyses of their work across all sections of the project. These evaluations include observations on performance, lessons learned, and improvements for future projects.

To achieve higher marks, evaluations must go beyond simply confirming that objectives have been met. While this can be included in the testing section, the evaluation should focus on critical self-reflection and analysis of the development process, including consideration of alternative approaches and personal development as a programmer.

Task marking
Comments on approaches to internal marking

It would aid the moderation process if centres would ensure that candidates' work and documentation are saved with filenames that clearly identify the centre number, candidate number and candidate name. As detailed in the specification for this qualification, "For example Diane Smith (centre number 68999, candidate number 12345) would store her work in a folder named 68999_12345_SM_D. In addition, candidates should ensure that they have linked their work to the GCE Computer Science Unit 5 Task sheet. The centre should make notes on the Centre Mark Sheet the nature of any assistance given and the extent to which the solution actually works as stated in the report there should also be specific reference to the assessment objectives in the comments written on the work and coversheets.

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: AS/A Level 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