



GCSE

Computer Science

8525/1 (1A/B/C)

Report on the examination

Version: 1.0
June 2025

Further copies of this report are available from aqa.org.uk

AQA retains the copyright on all its publications. However, registered centres for AQA are permitted to copy material from this booklet for their own internal use, with the following important exception: AQA cannot give permission to centres to photocopy any material that is acknowledged to a third party even for internal use within the centre.

Introduction

In general, the students seemed well prepared for this examination paper and showed a good application of the problem solving and computational thinking skills they have been taught.

The coding questions were well answered, on the whole, in all three languages. However, there is still some confusion shown when using the indentation grids that have been provided for coded responses. It is especially important for Python centres to work with their students on the correct use of the indentation grids within the answer spaces as these are provided to support them in making their indentation clear but, in many cases, it is not being used as intended and the student's indentation is not always clear as a result. VB.NET and C# centres can also use the grids to keep their code inline and tidy which helps the students to see where their iteration and selection structures begin and end.

With the coding questions, students should also be careful that they understand the difference between outputting a value and returning a value. Where a question states that they must do one of these things marks cannot be awarded if they confuse them. A common error continues to be using the word `output` instead of a correct `print/writeline` statement in all languages.

Students continue to use mathematical symbols incorrectly (such as $\geq$) which would not work in programming languages. Also, the order in which the symbols are appearing would cause issues if programmed in an IDE (e.g. `=>` instead of `>=`). Some students were not gaining marks due to confusing the symbols (using greater than instead of less than and vice versa).

This year we saw an increase in students answering in a different language to the one they were entered for or using pseudocode when a HLL was required. This means that not all of the marks for that question are available to them. Students should answer using the method stated in the question.

Students also spent time writing out input and output statements which were not required. Students can omit superfluous I/O statements unless the question specifically asks for messages to be output or for input messages to be used. If the student prefers to include them for clarity, they should keep the messages brief as longer messages may inadvertently introduce errors.

There is a coding guide available to show teachers how the examiners mark the coding questions and centres are encouraged to refer to it.

Question 1

This question was about operators. Just over 75% of students were able to identify where the arithmetic operator was for the algorithm, this dropped to only 60% being able to correctly identify the relational operator. A common error included writing the whole of line 5, which didn't show that the student understood which part was the answer. As previously mentioned, a lot of students answered using "greater than symbol" which was the wrong way around and couldn't gain the marks.

This question block also had the first of the trace tables for this year's paper and it was pleasing to see over 80% of students gaining at least two of the four marks. A lot of the incorrect responses didn't identify that the variable was output inside the loop, choosing to only output once the algorithm had finished

Question 2

This set of questions was the first of the coding questions on the paper. The first question asked students to complete three lines of code. Students need to be careful when answering this type of question and that they only write the code that has been asked for. Over 80% gained two of the three marks with syntax errors being the main reason the third mark wasn't accessed. (50% gained all three).

Question 2.2

This should have been a fairly straightforward programming question where two marks were available for taking user input and assigning it to three variables. It was pleasing to see approximately 80% of the students gaining three of the available marks. Unfortunately a lot of students made errors with datatypes either forgetting to cast, using integer values for values that multiply for a decimal or using incorrect/none existent datatypes which would cause a syntax error (e.g. real).

Question 3

Question 3 was a second straightforward programming question, and whilst over 60% of students gained 7 of the available 8 marks, only 30% gained full marks. It was errors in logic that seemed to cause the most problems for that final mark. A lot of students checked for the number of students being greater than 15 before they checked it for 25. Another common error was getting the boundaries wrong. Students should check their solutions carefully to catch these logic errors in their code.

On questions 2.2 and 3, there were 3 marks available for taking user input correctly and storing it in a variable – as a starting point for coding I would expect all students to be able to write this line of code however there were a number who could not do this successfully.

Question 4

Question 4 examined students understanding of validation and only 40% of students chose the correct option for 4.1 with a lot of students focusing on the part of the question stating name rather than identifying the purpose of validation. However, over 90% were able to identify the definition of boundary test data for Q4.3, and 60% were able to give a correct example. Wrong answers often included 2 or 14, which demonstrated knowledge of boundary test data but showed that they had not read the question properly.

Just over 50% identified erroneous data as being the third type of validation.

Question 4.2 focused on a validation program, and it was pleasing to see 60% of students gaining 4 of the 6 available marks. Errors occurred with trying to correctly find the length of the string and logic errors (a number cannot be <1 AND >15).

Question 4.6 required the students to put some given labels onto a flowchart – some students wrote their own pseudocode which was not correct instead of using the statements given to them.

Question 5

This was another question looking at definitions and it was disappointing to see that only 20% of students could correctly describe what a data type was. There were very few answers mentioning variables storing the data and a lot of incorrect answers referred to the “type of data.”. This number increased to 35% when they were explaining why the program needed integers in order to run. A lot of students wrote about the variables not being strings because they were cast to an int which didn’t answer the question asked. Over 65% were able to identify the data type of a value with a decimal point with too many students incorrectly answering Real which is not a data type in any of the three languages within the specification.

For question 5.4, 60% of students gained 3 or more marks. Some students stored the number in an array, which was unnecessary but full marks were given for this if it was written correctly. However, with more complicated answers such as these, there is more scope for errors occurring and this is what happened in a number of cases, particularly with array indices.

The `range` function proved to be problematic with many students using `range (4)` instead of `range (5)`, other solutions contained `while counter <= 5` rather than `while counter < 5`.

Question 6

This question block focused on sorting and searching algorithms. Students showed a better understanding of completing the bubble sort than the merge sort with over 64% gaining full marks on the former. Some students completed a full pass rather than showing each change which limited the marks available.

Less than 40% gained full marks on the merge sort with a noticeable error being merging and then sorting the values as two separate steps.

There were some well written answers for the advantages/disadvantages of the bubble sort and the majority of students were able to identify at least one. Over 80% of students gained at least one mark when explaining why the binary search was better with the sorted list. Some students are still not qualifying their answers which means it is not clear what they mean, and so marks cannot always be given.

Question 7

This question tested understanding of a standard subroutine that checked if a value is even. It was pleasing to see over half the students were able to identify the purpose, and over 86% were able to trace at least one of the two given values through the algorithm. Whilst over 66% were able to give the return value for Q7.3, only 40% could name the return value. This was surprising to see as this is a common type of question on the paper and a key term from the specification. Less than 10% of students were able to give 2 valid characteristics of the structured approach to programming and the majority of students gave the benefits instead.

Centres should remind their students to read the question carefully as there were a few places in this paper where the answer given was technically correct but not an answer for the question asked.

Question 8

Question 8 was another trace table and tested the students ability to work with string indexes. Over half of the students gained three of the five available marks, with the values being assigned to the string new causing some issues for students. Nearly 50% were able to show they understood 0 based indexing and length.

Question 9

This question gave the students 5 algorithms to check, and it was pleasing to see that over 70% were able to categorise at least 3 correctly, nearly 40% could categorise them all.

Question 10

Question 10 was testing the student's knowledge of records and how they are formed. As with previous years this posed some issues for the students. Teachers are advised to use the resources that are available to them through the AQA website to support the teaching of records, as it was clear that few students were prepared for this type of question with only 17% gaining marks on Q10.2

We saw some really well thought out solutions for the coding question 10.3, and nearly 40% of students accessed at least half-marks. Similar issues affected student's code with loops taking the list/array out of bounds.

Another issue seen was that only a small number of students were able to initialise the shortest value to one of the array elements or a large number, instead choosing to initialise it as zero, which logically wouldn't work.

Question 11

Question 11 gave a subroutine to the students to work through; the multi-choice questions showed a good understanding of the algorithm with only 11.2 (following through the else option) causing issues for over half the students.

Q11.5 asked the students to extend a program and include the given subroutine as part of their solution. Again, we saw some interesting solutions, but students were unable to correctly call the subroutine. A significant number also thought that the local variable `newhidden` would be accessible outside the `findletter` function and used that, instead of storing the function's return value in the variable `hidden`.

Over 40% of students gained over half marks but disappointingly 17% of students did not attempt the question. A common error was not including iteration in the solution or using built in subroutines to check for letters/asterisks when the question specifically asked them not to. As with previous questions the Python `range` method caused issues with a lot of students using `for i in range (7)` which would run one too few times.

The errors in logic and boundaries that were present in this year's papers are errors that we are seeing each year and across all three languages. This is alongside the misuse of quotation marks (either missing them on strings or adding them to integers).

Final note

A small number of students did not attempt the coding questions. Teachers should encourage students to attempt every question as often marks can be gained for straightforward variable assignment and Boolean conditions. There are also design marks available for the coding questions which means that some marks can be gained even if the students code is syntactically incorrect. A tip for students might be to tick off each item on the bullet list in the question when they have included it in their answer as the list of requirements is very concise.

AQA has provided resources on their website which show the coding guidance that the examiners use as well as programming guides and pseudocode guides to support the teaching of this paper which covers the areas that will be tested on the exam.

Mark Ranges and Award of Grades

Grade boundaries and cumulative percentage grades are available on the [Results Statistics](#) page of the AQA Website.