



A-level Computer Science

7517/2

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.

Question 1

Question 1 was about software classification, the operating system and levels of programming language.

For question part 1.1 students had to correctly name types of software. The vast majority of students were able to correctly name at least two types but only a third could correctly name all four types. The types of software that were most commonly not known were libraries followed by utilities.

Question part 1.2 asked students to describe two functions of an operating system. The specification lists three functions which are allocating processors, memories and I/O devices to running processes, but many other functions were described and credited. The most commonly made mistake was to simply name a function such as “security management” rather than to describe what the task involved.

Question part 1.3 asked students to describe the advantages of using a high-level language to write programs. Approximately three quarters of students were able to describe at least two advantages. Good responses covered topics such as features provided for modularity, faster development times, the availability of code libraries and support for data structures. The most commonly made point was that program code was easier to understand. Some students explained this point well but others made somewhat imprecise points such as the code being “like English”. Some answers were quite repetitive, with the point that the code was easier to understand being made in multiple ways.

Question 2

Question 2 was about representing sound and analogue to digital conversion.

For question part 2.1 students had to calculate the sample resolution that was used when a sound was sampled. This was very well answered with almost three quarters of students achieving both marks. The most common mistakes occurred when converting between units, with students forgetting to multiply by 8 to convert from bytes to bits or multiplying by 1000 an incorrect number of times.

Question part 2.2 asked students to describe the principles of an analogue to digital converter. The majority of students achieved at least one mark but only a fifth achieved both. Students often knew that samples were taken at a fixed rate and that they were encoded in binary but were imprecise in their descriptions of what a sample was, not being able to recognise that a sample was a measure of the amplitude of a wave at a fixed point in time.

For question part 2.3 students had to explain why a digital sound file might not accurately represent the original sound. Good responses recognised that the sampling rate or sample resolution might be too low and went on to explain what the impact of this would be. For example, if the sample resolution was too low then the extent of the rounding when a sample

was quantised would be too great. Students who wrote about a low sample rate often quoted Nyquist's theorem, although sometimes too vaguely to achieve both marks. In particular, students were sometimes unclear about what it was that the sample rate needed to be double of.

Question part 2.4 was about why MIDI files use significantly less storage space than sampled sound. This fact was frequently stated by students in responses to questions on previous papers, but students found it harder to explain why it was the case. Just over half of the students achieved some marks on this question. The best responses recognised that MIDI stored the information needed to generate the sound such as the properties of a note and that these properties would use far less storage than the many thousands of samples that might be stored to represent the equivalent sound using sampling.

Question 3

Question 3 was about storage devices.

There were many good descriptions of how optical storage represented and read data for question part 3.1 although few students went on to achieve all six available marks. Common issues were not knowing the terminology for pits and lands, referring to the laser itself reading the data, and omitting the idea that some type of sensor would be needed to measure the reflected light.

For question part 3.2 the issues with optical disks that have contributed to the reduction in their use were well known and students were as likely to lose marks for stating a reason imprecisely as for stating an incorrect reason. Example imprecise reasons which were not markworthy were "not much space" and "too slow". Some students made points relating to robustness, most of which were markworthy, but students should be careful to consider if points relate to the disk or the drive when making them.

Question 4

Question 4 was about the difficulties of writing and enforcing laws relating to the use of computers. This was very well answered with the most common responses relating to the evolution of technology, the difficulty of crimes being committed across international borders and the ability to conceal identity online. Approximately half of the students achieved at least two of the three available marks.

Question 5

Question 5 was about the suitability of a client-server network for use in a school.

Students typically wrote about the features of client-server networks rather than the features of the school context and missed the opportunity to gain marks for pointing out simple things

like the fact that a school was likely to have many devices connected to its network and that it would not be possible to trust all of the users.

The most commonly achieved marks were for recognising that security could be centrally managed, allowing appropriate access rights to be assigned to students and teachers, that backups could be easily made as the data was stored on the server and that software updates might be more easily rolled out.

Some students confused the comparison between client-server and peer-to-peer networking with a comparison between thin-client and thick-client computing.

Question 6

Question 6 was about relational databases.

For question part 6.1 students had to recognise that in the database system it would not be possible for the same teacher to award a merit to the same student more than once in a day. Nearly half of the students recognised this and achieved the mark.

Question part 6.2 asked students to complete an entity-relationship diagram. This was well done. Many students drew a many-to-many relationship between the Student and Teacher entities. This was not necessary as it was not modelled directly in the database, but it did not prevent the award of the mark.

For question part 6.3 students had to write an INSERT INTO query. This was well answered with approximately a third of students achieving both marks. The most common mistakes were to miss out the VALUES keyword or the brackets around the list of values and to write INSERT INTO in the incorrect place, for example putting a field list between the INSERT and the INTO.

Question part 6.4 required students to write an SQL query to retrieve data about merits that were awarded. The majority of students achieved at least three of the available six marks but only a tenth achieved full marks. The most common mistakes were not including the Term table, which was needed to obtain the term start and end dates, and using > and < instead of >= and <=.

Good responses to question part 6.5 about concurrent editing recognised that the update made by the first user who saved their edits would be overwritten when the second user saved. Students often made less developed points such as “some data being lost” which were not precise enough to be markworthy.

Question 7

Question 7 was about the processor.

Question part 7.1 asked students to state the stored program concept. This was not well tackled and many students wrote about the von Neumann architecture instead, often incidentally

achieving a mark owing to the overlap of the concepts. Good responses recognised that machine code instructions are stored in main memory and that instructions are fetched and executed serially by a processor that performs arithmetic and logical operations.

For question part 7.2 students were asked to describe the role of the control unit, general purpose registers and the status register. The role of the control unit was described best with many students recognising that it would manage the overall operation of the computer, including sequencing the Fetch-Execute cycle and fetching and executing instructions. Many students recognised that general purpose registers were fast access memory locations but few went on to describe what they would be used to store. The role of the status register was least well known. Some students stated that the register would be checked once during each cycle, but this is not the role of the register. Better responses identified that the register stored information about the result of executing the previous instruction, such as if overflow had occurred, and that this information would affect the behaviour of the next instruction.

For question part 7.3 students had to calculate the number of general-purpose registers that could be addressed given the format of an instruction. A third of students calculated the correct answer of 16. The most common incorrect answers were calculated using 2^{12} instead of 2^4 or using $2^4 - 1$, not realising that 0 was a valid register number.

For question part 7.4 students had to calculate the range of integer values that could be represented in 12 bits using two's complement. The correct answers could be calculated as $2^{11} - 1$ for the most positive value and -2^{11} for the most negative value. The most common mistake was to give 2^{12} as the most positive value and -2^{12} for the most negative value.

Question 8

Question 8 was an extended response question about networking. It focused on the change from IPv4 to IPv6, public and private IP addresses, NAT and DHCP. Three quarters of students achieved at least three of the available twelve marks but only a tenth achieved a mark of ten or above.

Most students were aware that IPv4 uses 32 bits to represent the address of a computer and that IPv6 has more addresses because it uses more bits, but relatively few knew that IPv6 used 128 bits. A commonly stated misconception was that IPv6 could store more addresses because it used hexadecimal instead of decimal.

Most students had some understanding of the difference between public and private IP addresses, but many did not achieve all the available marks because they did not make enough valid points or because they expressed points imprecisely. Good responses recognised the scope of the uniqueness of each type of address and explained whether or not they could have data directly routed to them or not. The most common misconception was to believe that public and private IP addresses were security measures and some students stated incorrectly that data could not be sent directly to a computer with a private IP address, not recognising that this could be done within the LAN or subnet that the device with the private IP address belonged to.

Network Address Translation (NAT) has featured on a number of recent papers and most students were familiar with the basic concept, however the level of technical detail about how NAT works varied considerably, and some students did not link back to the question and did not explain how NAT was used to help with the problem of there not being enough IPv4 addresses. Good responses recognised that this was because an entire network would only require one public IP address instead of one per connected device.

The purpose of DHCP was well known with most students being able to describe that it would allocate a network configuration including an IP address to a device as it joined a network. As with NAT, whilst the basic purpose was well known many responses contained only limited details about how the allocation process works, with many students knowing how this started with a request but did not explain the later stages.

Question 9

Question 9 was about logic circuits, Boolean expressions and character sets.

Question part 9.1 was very well tackled with the majority of students achieving full marks for completing the truth table, despite the complexity of the circuit.

Just under half of the students then went on to correctly describe the purpose of both outputs for question part 9.2. Some leniency was applied, but for output Q2 it is worth noting that the opposite of an uppercase letter is not a lowercase letter as the ASCII character set includes other symbols too.

Three quarters of students were able to select the correct Boolean expression for one of the circuit's outputs for question part 9.3.

Question 10

Question 10 was about compression.

The purpose of compression was well known with the majority of students achieving full marks. The most commonly seen response that was not markworthy was that it would allow data to be transmitted more easily. Responses that linked to transmission time did achieve the mark.

For question part 10.2 students had to explain why run length encoding (RLE) is more suitable for compressing images and dictionary-based methods are more suitable for compressing text. Good responses recognised that RLE was suitable for images as some types of images were likely to contain many repeated adjacent pixels of the same colour and that it was not suitable for text as whilst text often contains repeated substrings these are repeated throughout the data and are unlikely to be next to each other. Students who did not receive marks had frequently did not contextualise their response, for example not making clear that it would be the colour of pixels that would be repeated and that the pixels would have to be adjacent in a line.

Question 11

Question 11 was about functional programming.

Question part 11.1 asked about the features of the Fibonacci function given on the question paper. Just under a quarter of students correctly identified that the function had one argument and was a first-class object. The vast majority of students correctly identified that the function had one argument but it was less well known that all functions in functional programming languages are first-class objects.

For question part 11.2 students had to identify the function type of the given function. Over three quarters of students were able to identify that the correct type was $\mathbb{N} \rightarrow \mathbb{N}$.

Question part 11.3 asked students to explain why the implementation of the Fibonacci function given was not efficient. The best responses recognised that the function called itself twice and that this would result in the same terms in the sequence being calculated multiple times. Some responses went on to link this to Paper 1 content, explaining that the time complexity would be exponential. This was not required but was also creditworthy.

Question part 11.4 was about partial application of a function. This part was not tackled well with only a fifth of students achieving any marks. Many responses appeared to be guesses at what the term might mean based upon the word “partial”. Good responses recognised that one or more arguments to a function are fixed creating a new function with fewer arguments. Students who achieved only one of the two marks most commonly did so by stating that a new function would be created.

Question 12

Question 12 was about floating-point numbers.

For question parts 12.1 and 12.2 students had to identify a value that was not normalised and the most negative normalised value from a list of numbers, which the vast majority of students did correctly.

Question part 12.3 asked students to state why numbers are normalised. This was not well tackled and only approximately a quarter of students achieved the mark. Good responses recognised that normalising produced a unique representation of every number which made it simpler to test for equality or that it would maximise precision within the available number of bits.

Question part 12.3 asked students to convert a negative number from floating point to decimal which the majority of students were able to do correctly. The most common errors were shifting the binary point four places in the wrong direction or correctly shifting it left but not knowing

how to deal with a negative number when the point was shifted left, arriving at an answer of +21/256 instead of the correct answer of -11/256.

For question part 12.5 students had to convert a very large number that could not be represented exactly from decimal to floating point. Approximately half of the students achieved at least two of the three available marks. The most common mistakes were to give the correct mantissa but to get the exponent wrong (most commonly 25 instead of 26) or to make arithmetic errors.

For the final question part students had to calculate the relative error that occurred in question part 12.5 and almost every student who correctly answered that part was able to go on to calculate the relative error correctly.

Question 13

Question 13 was about Boolean algebra. Almost all students knew how to tackle a simplification and the majority achieved at least one mark but only a very small number went on to achieve all four.

Students who delayed applying De Morgan's laws until later in their solutions were more successful than those who did so as their very first step. Students who applied De Morgan's laws first often immediately arrived at the expression $\bar{A} + \bar{B} \cdot A + \bar{C} \cdot \bar{B} + \bar{C} \cdot B + \bar{A}$ but this is incorrect owing to the precedence of AND being higher than OR. For the expression to be correct, brackets needed to be introduced like this $(\bar{A} + \bar{B}) \cdot (A + \bar{C}) \cdot (\bar{B} + \bar{C}) \cdot (B + \bar{A})$.

Students who applied more straightforward simplification steps first, such as factorising and recognising that $X + \bar{X} = 1$ and $X \cdot 1 = X$ quickly picked up method marks. Students should show as much of their working as possible as if a student makes too big a leap in one step at best they will miss out on some method marks that they might have achieved and at worst an examiner may not be able to see a convincing route by which the final solution was derived.

Some students achieved three out of the four marks because they did not see that a final application of De Morgan's laws could be made to simplify $\bar{A} \cdot \bar{C}$ to $\overline{A + C}$.

Question 14

Question 14 was about assembly language. This was a new style of question on this topic where students had to describe what the assembly code was achieving in the context of a robot being driven, with much of the code using masking. Two thirds of students achieved some marks but only a small number achieved all six.

The most commonly achieved marks were for recognising that the code in Section 3 would stop the robot moving and that the code in Section 5 would activate the robot's arm. The hardest section of code to describe was Section 4. Good responses for this section recognised that the number of touch sensors that were active would be counted and that if exactly two sensors

were active the program would carry on, otherwise it would end. Students sometimes did not achieve marks in this section by not recognising that it was the touch sensors that were being checked or by just stating that the robot would stop rather than correctly recognising that the entire program would end.

Mark Ranges and Award of Grades

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