AQA

A-LEVEL
COMPUTER SCIENCE

7517/2
Report on the Examination

7517
June 2022

Version: 1.1

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

Question 1

Question 1 was about hexadecimal.

In question part 1.1, the majority of students were able to achieve both marks by correctly describing how to convert a binary number into hexadecimal. Some students simply did the conversion, which was not markworthy. Students who achieved one of the two marks usually got the mark for explaining that the value would be split up into 4-bit sections but then wrote a description of the conversion progress of each section that was imprecise or incorrect.

Question part 1.2 was also well answered, with the vast majority of students achieving the mark by stating an appropriate reason for why hexadecimal is used instead of binary. Some answers, although markworthy, were quite superficial and it would be nice to see students expanding their points more. The only common incorrect answer was that a hexadecimal number could be represented in fewer digits than binary. Whilst a hexadecimal number can be displayed in fewer digits than the corresponding binary number, the number will always be stored in binary.

Question 2

Question 2 was about data communications.

Question part 2.1 was well answered, with the vast majority of students recognising that parallel transmission would use multiple wires for transmission. Many students then went on to explain that multiple bits would be transmitted simultaneously, with one bit being sent along each wire, but some students failed to achieve this second mark as this part of the response was less clear, for example a response might refer to data or packets being sent simultaneously rather than discussing how the parallelism was achieved at the bit level.

Part 2.2 was also well answered. The most commonly stated answer was that serial transmission does not suffer from data skew. Other good responses covered issues such as crosstalk or the cost/complexity of the hardware needed. To achieve a mark relating to interference, students needed to name or describe a type of interference that would not also affect serial transmission.

Approximately three-quarters of students identified that latency was not defined correctly for question part 2.3, with incorrect responses relatively evenly split between the other alternatives.

Question parts 2.4 and 2.5 related to start and stop bits, which are not well understood.

The most commonly seen response to question part 2.4 was that the start bit identified the start of a data transmission. This was not enough to be markworthy. Good responses identified that the start bit would be used to bring the receiver's clock into synchronisation with the sender's clock. Some students wrote about clocks being synchronised, but did not identify which clocks they were referring to – this was not enough for the mark to be awarded.

The most commonly seen response to question part 2.5 was that the stop bit identified the end of a data transmission. This response was not markworthy. Good responses explained that the stop bit was sent so that the next start bit could be recognised or to give time for the receiver to process or transfer the received data.

Question 3

Question part 3.1 was very well answered, with over three quarters of students completing the truth table correctly. A similar proportion of students correctly identified that the truth table demonstrated the correctness of De Morgan's law for part 3.2.

Most students found the Boolean simplification in part 3.3 challenging, and only approximately a quarter of students achieved full marks for arriving at the correct final answer through valid working. The most common mistakes were:

(1) To simplify $A + \overline{A}$ to 1 at the start. This was not possible as the A is connected to $(B + 1)$ with an AND operator $(\cdot)$, which has higher precedence than OR $(+)$.

(2) To apply De Morgan's law incorrectly to multiple operators, and make the following incorrect simplification:

$$\overline{\overline{A} + B \cdot C + B \cdot \overline{C}} = A \cdot \overline{B} + \overline{C} \cdot \overline{B} + C$$

De Morgan's law can only be applied to multiple operators at the same time if the operators are all of the same type.

Question 4

Question 4 was about system software and the operating system.

The majority of students gave good responses to question part 4.1, which recognised that system software was used to manage the operation of a computer and to perform tasks that were only required because there was a computer. Common responses that were not markworthy were that system software was used by the operating system or was supplied with a computer.

Question part 4.2 was very well answered. The vast majority of students correctly identified that a bitmap image editor is not an example of system software.

For question part 4.3, students had to describe functions of an operating system; this was not well tackled. It was not sufficient to simply name functions, such as "memory management". Some students put the names of functions into sentences, such as "the operating system will carry out memory management", but still did not describe what this was and so did not achieve a mark. Good responses described what a function was, for example that for memory management regions of the computer's memory would be allocated to different processes. The provision of a user interface was not markworthy; to achieve a mark this point needed to be developed further to explain how the operating system was a layer of software that abstracted the user from the complexities of the hardware.

Question 5

Question 5 was about floating point numbers.

Question part 5.1 was well answered; the majority of students were able to convert the fixed point binary number into decimal. Common mistakes were to treat the bit that should have had the value -32 as +32, giving an answer of 44.6875, or to sum the bits to the left of the binary point to -20, and the bits to the right to 0.6875 and give the answer -20.6875 instead of adding the 0.6875 to the -20 to give the correct answer of -19.3125.

Question part 5.2 was not answered well, with many students giving vague response such as "increased accuracy", "easier to tell if positive or negative" or "easier to understand". Good responses recognised that normalising floating point numbers maximises precision within a given number of bits and ensures that there is one unique representation for each number, which makes it simpler to test two numbers for equality.

Part 5.3 was very well answered with most students giving the correct response as the fraction 27/4096. The most common mistakes were to move the binary point seven places to the right instead of to the left or to give the exponent as +9 instead of -7.

The majority of students were able to give the correct floating point representation of -23.25 for question part 5.4. Mistakes were often made by students who tried to write out -23.25 directly in fixed point, instead of writing 23.25 in fixed point first and then converting this to -23.25. Students who did this often wrote out -23 followed by +0.25 and gave the fixed point binary as 101001.01, which is in fact -22.75 in decimal. Students should clearly show their working in this type of question, as this will be credited if the final solution is incorrect. Some students arrived at an incorrect but almost correct final answer but had not shown sufficient working for this to be credited.

Question part 5.5 was the best answered part of question 5; the vast majority of students correctly calculated that the absolute error, to 2 decimal places, was 0.43%. Common mistakes were to calculate the absolute error to be a negative value, resulting in a relative error of -0.43% or dividing the absolute error by 0.2265625 instead of 0.22558594. Students who arrived at incorrect final answers often did the correct calculation but made a calculating error at some point.

There was an error in question part 5.5. The question stated that the value provided in Figure 4 was a normalised binary representation, however, the value provided was not normalised. This error did not have any impact on students answers or the marks awarded.

Question 6

For this question, students needed to describe how bar code and RFID input devices operate and to discuss the ethical and legal implications of a supermarket loyalty card scheme that used these.

Almost all students scored some marks for this question and the majority of students achieved a mark in Level 3 or 4. Some students did not cover one of the three areas of the question and so limited the maximum mark that they could achieve.

Good responses about ethical and legal issues recognised that the consent of the customers would be necessary for the system to be used and that this would be regulated by GDPR, which would place a range of requirements on the supermarket. Ethical issues discussed included who should have access to the data and what the data should be used for. Some students assumed that the data was being collected without the consent of the customers, which was unlikely in this scenario. A discussion about the need for consent was markworthy, but a statement that the customers had not consented was not.

Many good descriptions of how bar codes are RFID work were seen. A common misconception about bar codes was that the colour of a bar represented a bit, for example black = 0 and white = 1. In reality, the bars are of different widths and a specific bit value is not associated with each colour.

Question 7

Question 7 was about relational databases and SQL.

For question part 7.1, the majority of students correctly identified that in a fully normalised database the primary key does not have to consist of only one attribute.

For part 7.2, the majority of students were able to correctly identify the degree of at least one of the direct relationships that existed, between Animal and AnimalLocation or AnimalLocation and Zoo. The most common mistakes were to draw a one-to-one relationship between Animal and AnimalLocation or a one-to-many relationship between Zoo and Animal. Whilst at any one time, an animal could only have one location and could only be at one zoo, the system captured historical data as well so an Animal could have more than one AnimalLocation and also have been at more than one Zoo.

Most students achieved some marks for writing SQL code to create the Animal table in question part 7.3 but less than a quarter achieved all three marks. Common mistakes were writing the data type before the fieldname and missing out commas between the definitions for individual fields. As these marks were for programming, the code had to be syntactically correct for full marks to be achieved. A small number of students wrote SQL code for a different table. Although there were only two options, it was not appropriate to choose Boolean as a data type for the sex of an animal.

Writing the SQL query for question part 7.4 was challenging, particularly determining the conditions that could be used to identify animals that had been at the zoo during the required time period. Pleasingly, almost all students knew the basic structure of an SQL query and were able to achieve some marks. The vast majority recognised that there needed to be a condition to link the Animal and AnimalLocation tables. The hardest part was writing the date conditions. Many students were able to write conditions to identify some animals that would have been at the zoo during the required period, but only a small number were able to write conditions that identified all animals that would have been there.

Approximately three-quarters of students achieved some marks for question part 7.5. Good responses recognised that the main advantage of storing the additional attribute would be that the current location of an animal could be found by querying the Animal table alone; it would not be necessary to perform a cross-table query including the AnimalLocation table. It was not enough to state that it would be easier to find out where an animal was. The most commonly given disadvantage was that data redundancy would be introduced, and that this would require additional storage space. Some responses developed this further by explaining that this could give rise to the possibility of data inconsistency and make updates more complex.

Question 8

More than three quarters of students achieved at least one mark for question part 8.1 which required students to compare peer-to-peer and client-server networks. Good responses recognised that in a peer-to-peer network, each computer would have equal status and could share resources with other computers and use the resources from other computers. In contrast, in a client-server network, each computer would act as either a client or a server and resources would be shared by the server which would receive requests from clients to access them. Many students also went on to discuss security and the reliance on the server in the two models.

A common misconception was that in a client-server network, all data would have to be transmitted through the server. Whilst all requests for resources would be sent to a server, it is possible for other devices on the network to communicate with each other independently of the server. Another misconception was that if the server failed on a client-server network all of the network would go down, or every computer would fail. Whilst the failure of a server would prevent its resources being accessed (which might prevent logins taking place), it would not cause the network or other computers to fail.

Approximately three quarters of students achieved some marks for question part 8.2 which was about whether a peer-to-peer or client-server network would be best suited to a student house. The most frequently awarded mark was for recognising that there would be very few devices on the network. Good responses also recognised that it was likely that the users would trust each other and that they would likely want the ability to choose for themselves which files they shared and with whom.

It was possible to obtain a mark by explaining that it might be cheaper as there was no need to purchase a server, but students who just stated that a peer-to-peer system would be cheaper did not receive a mark as this was a statement rather than an explanation.

In this type of question, it is important that students address the context when giving their response, rather than just listing general advantages of a peer-to-peer system.

For question part 8.3, most students stated that the purpose of DNS was to translate domain names into IP addresses. A mark was awarded for this response, but it should be noted that it is actually fully qualified domain names that are used. Some students referred only to addresses, rather than IP addresses which was not markworthy.

Good responses went on to discuss the organisation of DNS, that servers are organised into a hierarchy which store a distributed database of mappings and that if one server does not know the IP address for a fully qualified domain name, the request for a mapping will be forwarded to another server.

Frequently, students went on to explain that it was easier for a human to remember a domain name than an IP address, which was not required for this question.

Question 9

Question 9 was about assembly language programming. For question part 9.1, the majority of students recognised that the assembly language instructions in the figure used direct addressing. The most common incorrect responses were immediate, indirect and add.

Most students achieved at least two marks for completing the trace table of the assembly language program in question part 9.2, but less than a quarter achieved full marks. The most common mistakes were:

- Shifting the value in R1 left instead of right.
- Storing a value with a decimal part in R1 when it was shifted right.
- Not copying the final value 115 from R3 into memory location 122.

Approximately a third of students gave a correct response to question part 9.3, which required students to have correctly completed the trace table in part 9.2. Students who had not successfully completed the trace table sometimes guessed at what the program did, but relatively few of these guesses were correct. Good responses recognised that the program multiplied together the numbers stored in memory locations 120 and 121.

For question part 9.4, just under half of the students recognised that the machine code that was output might execute more quickly or use less memory if the original program was written in assembly language instead of a high-level language. It was not markworthy to state that the program would have better access to the hardware of the computer as the purpose of the program was to perform multiplication, which did not require this.

A common misconception was that assembly language would not need to be translated. It is also unlikely that the speed of the translation process would be a factor in deciding whether to use assembly language or a high-level language.

Most students achieved the mark for question part 9.5. Good responses either stated explicitly that there was a 1-to-1 relationship between assembly language and machine code instructions or described this relationship in more detail, for example by stating that each mnemonic in assembly language corresponded to a unique binary pattern in machine code. Some responses were too vague or did not refer directly to instructions, such as stating that they were both low-level languages.

This question was about sampled sound and MIDI.

For question part 10.1, the majority of students correctly calculated that the duration of the sampled sound was 195 seconds. The most common mistake made by those who did not arrive at the correct answer was to confuse megabytes and kilobytes and convert 17.199 megabytes into 17199 bytes instead of 17199000 bytes.

Question part 10.2 was well answered with the majority of students being able to give two advantages of using MIDI, although full mark responses were quite rare. The most commonly given correct responses were that a MIDI file would take up less storage space and that MIDI files are easier to edit. It is worth noting that it is also possible to edit sampled sounds, it is just that certain modifications such as changing the instrument that music is played on cannot be changed if sampling has been used.

Question 11

This question was about thin-client and thick-client systems. It was not well answered, with just over a third of students achieving any marks.

Many students used very vague terminology when writing about the hardware requirements of thin-client and thick-client systems. Phrases such as “high-end server”, “powerful server”, “server with lots of hardware”, “lots of power”, “cheaper computer”, “weak hardware and “good communications” were often seen and were not markworthy. Whilst they did convey some understanding, at A-Level, students are expected to be more precise, for example to discuss the amount of RAM required, or that processors with multiple cores might be necessary. Some students also failed to achieve marks as, when giving hardware requirements, they did not state whether these related to a server or to clients.

Question 12

This question was about functional programming.

The majority of students correctly identified that function `fv` included a higher-order function in its definition for question part 12.1. For part 12.2, over three-quarters of students correctly identified that functions `fw` and `fx` used recursion.

For question part 12.3, most students achieved two marks for writing the results of the calls to `fu` and `fv` (which involved the use of `map`) on the top two rows of the table. To achieve the mark for the result of the call to `fv`, it was necessary to write the list in brackets. Fewer students were able to apply the recursive functions `fw` and `fx` correctly to achieve the final two marks, but approximately a quarter of students did achieve all four marks.

The number of correct responses to question part 12.4 was relatively small, reflecting the fact that students who were unable to calculate the results of the function calls in part 12.3 were unlikely to be able to explain the purpose of the function `fz`. Good responses explained that the function calculated the average temperature in degrees centigrade from a list of temperatures in degrees Fahrenheit. Some students gave responses that were not quite enough for the mark, such as that it calculated the average of some values in a list or that it converted temperatures from Fahrenheit to centigrade.

Question part 12.5 was well tackled. The best responses explained that the conversion to centigrade would only be carried once with the new version of function `fz`, which would mean less processing steps were required.

Question 13

This question was about computer hardware.

For question part 13.1, the majority of students correctly identified that the definition was of the stored program concept. The most frequently selected incorrect response was the von Neumann architecture.

Most students achieved a mark for question part 13.2, but relatively few achieved both marks. The most frequently seen correct response was that secondary storage is needed to store programs and data whilst a computer is turned off. Responses sometimes also include the facts that secondary storage devices could be used to transfer data between computers or an explanation that the contents of RAM would be cleared when the power was turned off.

A commonly seen response was that secondary storage would be used to increase the amount of storage available. This was not enough to be markworthy by itself, but with further explanation such as that files larger than the size of the addressable memory could be stored, it was sometimes developed into a markworthy point. It was also not markworthy to simply state that secondary storage was non-volatile, without going on to explain what this meant.

A common misconception was that main memory is volatile and would lose its contents when there is no power. Whilst this is true of RAM, it is not true of ROM and therefore not true of main memory. Another common misconception was that secondary store was used to keep a backup of main memory.

Most students achieved at least two marks for question part 13.3 and there were many detailed descriptions of how an SSD worked that achieved full marks.

The most commonly achieved mark was for stating that an SSD did not have moving parts or stored data purely electronically. Good responses went on to explain how floating gate transistor were used and that 0s and 1s were represented by the presence of absence of a charge. Students often went on to explain how data was structured into pages and blocks and how individual words of data could not be overwritten, instead a whole block would need to be read, edited and then written back.

A small number of students wrote about magnetic hard disk drives or optical disc drives instead of SSDs or listed advantages of SSDs but did not describe how they worked.

Mark Ranges and Award of Grades

Grade boundaries and cumulative percentage grades are available on the Results Statistics page of the AQA Website.