Questions matter

AS

Computer Science

7516/1 Paper 1

Report on the Examination

7516
June 2024

Version: 1.0

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

General

The quality of student responses across the piece was indicative of a similar level of difficulty to last year's exam. One notable difference was that in the 2023 exam, students who were short on time prioritised the final question over the penultimate question. That was not the case this year, which indicates a good self-awareness among the cohort as to the nature of both the questions and individual ability. A student who performed well on the previous paper would have performed well on this year's paper.

The problem of unreadable screen evidence exists to a lesser extent than in previous years. Nevertheless, it should be reiterated that students will not receive marks for captures of screens that have not been produced by running their code. Students should also make sure, when pasting in screen captures, that they are readable. The height of a capital letter should be at least 2mm when the EAD (Electronic Answer Document) is printed. Students should copy and paste code from their editor/IDE rather than take screen captures of the code.

When answering a question that requires an identifier students should not include anything other than the identifier (a sequence of letters and underscore characters alone).

Question 1

This question required completion of a trace table based upon a pseudocode algorithm that involved concatenation of binary digits. Although the modal score was full marks, many of those who picked up fewer marks did so as a result of a very common error, namely concatenation of a binary string with an additional bit. Specifically, a new bit was appended to the wrong end of the string. This error occurred when parsing the only instruction that required supplementary explanation in the exam. This suggests that students were relatively well-prepared for commonly encountered pseudocode expressions, but less well equipped for new and unexpected instructions.

Question 2

This was a question on the differences between global and local variables, and responses varied in quality significantly. A common error was to refer erroneously to the calling of variables, which was the most common means by which a mark point was missed. A minority of students, below 10%, provided answers of extremely good quality, which provided far more detail than needed for either of the two-mark parts to this question. This may have had an impact upon available time in subsequent questions.

Question 3

Most students - almost 90% - demonstrated some knowledge of the advantages of using structured programming, the most common single-mark attempt (on a three-mark question) being to address understandability. This question was a good discriminator of ability.

Question 4

Students generally performed very well on the pseudocode-to-code question, with almost 90% securing at least six of the eight available marks in part 1. The most commonly dropped mark point was the final one, which required an output built up of individual characters to appear on a single line. Approximately half of the students failed to correctly implement this, with their (otherwise correct) output being generated one character per line. This was the most common reason for the screen capture mark not being awarded.

Questions 5, 6 and 7

These questions required identification of a data type and identification of a corresponding variable from the skeleton code. Most students picked up all marks here, although there was a higher level of success in identifying a data type (this being knowledge) than in identifying a variable (this being basic analysis).

Question 8

This was the first question that required students to demonstrate meaningful analytical capability in the context of the skeleton program. This question asked about the purpose of a particular line of code. Slightly more than 50% of students picked up the mark for this question, which would have required close analysis of the skeleton code in advance of the examination.

Question 9

Like question 8, the focus here was to explain why the skeleton code was designed and developed in a particular way. Success in this question was likely to have been determined by the nature of any exam preparation as much as by exam technique. Levels of success in this question were comparable to the previous question, which had a similar focus.

Question 10

This question, comprising two parts, required students to identify data structures for user-defined and built-in data types respectively. Most students gained no marks here, although some understanding was demonstrated in that the vast majority of responses were data structures. A common mistake was to transpose user-defined and built-in, although a small number of possibly risk-averse students provided the same answer to each of the two questions, ensuring that one mark was gained.

Question 11

This was a four-mark question on composition, with two marks available for a general description, and two for addressing why composition was used in the skeleton program. This was a departure from questions on previous papers, with students more likely to expect a question on decomposition than on composition, and the quality of responses varied widely. A common mistake was to describe subroutines or modularity in a more general sense, and only around 1% of students gained full marks here.

Question 12

This was a variation on a standard question, focusing on the benefits of using named constants. Slightly more than 50% of students picked up at least one of the two available marks. The generic aspect of the question - namely the benefits of using named constants - was better answered than the specific aspect of the question, in which students were required to identify the purpose of a specific named constant.

Question 13

This question addressed the purpose of a particular loop within the skeleton program, which proved a challenge for most students, with just over 50% gaining no marks on this question. Like questions 8 and 9, this question required students to address why a particular piece of code was written the way that it was written, and this proved to be a challenge.

Question 14

This was another question that addressed the code from an analytical standpoint, the focus here being 'how does this work' rather than 'why was this approach taken'. This proved to be more of a challenge than question 13 and exposed a commonly held misconception: although a variable was involved in the calculation of an average the variable itself was not an input to the calculation – the variable was an index to an array element which was then the input to the calculation. This additional level of complexity explains why fewer than a quarter of students gained full marks.

Question 15

Here, students were required to suggest - but not implement - a change to the skeleton program. For students who would have been able to implement the change, this question presented a different challenge in that they were required to explain rather than program, demonstrating a different skillset.

Question 16

The first question of section C, in which modifications were to be made to the skeleton code. Of the students who attempted this question, over 90% picked up some marks, and more than 50% of students secured over half of the marks. Given the nature of any programming task, a wide variety of mistakes and misconceptions was demonstrated, but most fell into one of two categories. One common mistake was, when opening more tills in the simulation, to write code that would allow more tills to open than actually existed. The instruction in the question paper that provides this constraint was "...until the maximum number of tills available have opened". Another equal common mistake was to use hard-coded values rather than variables.

Question 17

This question was aimed at varying the speed at which different tills in the simulation operated, and great diversity of quality was demonstrated among student answers. The most commonly missed mark required students to provide code that wasn't explicitly requested, but that was indicated in the question by way of "...and any code you have changed or added to the Skeleton Program". Students generally included the required change to the parameter list of a subroutine, but did not include evidence of the updated call to the subroutine.

By this point in the exam, a significant number of students appeared to be running low on time. More than half of students did not submit the screen evidence required for question 17.2, which was typically indicative of a partial attempt at a question.

Question 18

No student gained full marks in the final question of the paper, although every mark point on the mark scheme was awarded. Approximately 5% of students gained 11 of the available 12 marks on question 11.1 but the missing mark point was not the same for all. The most commonly missing mark was one for which students were required to call a subroutine under two conditions, whereas only one condition appeared in their code.

Nearly 75% of students provided no screen capture evidence for question 18.2, which again was indicative of time pressure.

Mark Ranges and Award of Grades

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