Questions matter

A-level
Computer Science

7517/1 Paper 1

Report on the Examination

7517
June 2024

Version: 1.0

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

General

Most students were well prepared for this exam and had made good use of the time available between the release of the Preliminary Material and the day of the exam.

The Preliminary Material is released on 1 September and centres should give students access to it soon after that date so that they have time to study this material. Centres should make sure they test the Skeleton Program on the computers to be used for the exam and ensure everything will go smoothly in advance of the exam.

Students will not receive marks for screen captures that have not been produced by running their code. There were still students who provided screen captures that were not produced by their code. Their time would have been better spent trying to get their program code to work correctly.

A copy of the Skeleton Program used by the centre should be included alongside the scripts sent to the examiner, whether or not the Skeleton Program was modified. A significant number of centres did not do this. A few centres attached a copy of the Skeleton Program to each student's Electronic Answer Document and sometimes also the exam paper, which is not required.

Some students took screen captures of code rather than copying the code out of the program editor and pasting as text into the EAD. Whilst this is allowed, it must be discouraged as it makes the code difficult to read, particularly if a dark background colour has been used. Screen captures of code also sometimes resulted in long lines of code only being partially visible when they went beyond the right-hand side of the screen which prevented some students from gaining marks.

Question1

Question 1 was about subroutines.

Most students were able to state some advantages of using subroutines, but fewer went on to explain how an advantage was achieved, which was necessary for marks to be awarded.

Question 2

Question 2 was about queue data structures.

Question 2.1 asked students to explain why a circular queue is normally a better choice of data structure than a linear queue. Some students gave answers about no need to shuffle items up when an item is deleted, others gave answers about there being no unused, but inaccessible, locations in the array after items had been deleted from the queue. Both of these approaches were acceptable.

Question 2.2 asked students to describe how an item would be removed from a circular queue. Most students were able to demonstrate some understanding of the process. A common mistake was to omit checking if the queue was empty before deleting an item.

Question 3

Question 3 was about the Halting problem.

Question part 3.1 required students to describe the Halting problem. Fewer than half of students demonstrated any knowledge of the Halting problem and less than a third got both of the available marks.

For question part 3.2 students had state the importance of the Halting problem. One common error was to get intractable and non-computable confused; some students also used computational problem instead of non-computable problem.

Question 4

Question 4 was about regular expressions and sets.

Question part 4.1 asked students to explain what is meant by the cardinality of a set. Over half of students were able to do this. One misconception that was seen a few times was that only finite sets had a cardinality.

Question part 4.2 had the lowest percentage mean mark on the paper with very few students identifying that the empty set would also be a subset. The most common wrong answers were that `{b, a}` would also be a subset (with students forgetting that there is no order to items in a set) or to say that `{b, a}` was not a subset (confusing subset with proper subset).

Question parts 4.3 and 4.4 were generally well-answered, with most students getting both parts correct. A common error was to state that the `|` symbol meant "such that" in a regular expression.

Questions 4.4 and 4.5 asked students to write regular expressions. A common misconception was that the brackets in `(bb)*` could be omitted with many students writing bb* assuming it would match with the same strings.

Question 5

Question 5 was about arithmetic expressions.

For question part 5.1 almost all students were able to state that the expression in Figure 2 was using Reverse Polish Notation (postfix notation was also a commonly seen correct answer). The most common incorrect answer was Polish Notation.

About two-thirds of students got at least one mark when representing the infix expression in Figure 2 using RPN. Students who only got one mark usually did so by writing the * after the 2 and 3. A number of students had little idea about how to do this and changed the order of the operands in the expression.

Question 6

Question 6 was about binary trees and in-order traversals.

For question 6.1 students had to state two properties of the graph in Figure 3 that made it a tree. Most students were able to give one property, but fewer were able to give both. Common incorrect answers were to state that there were no cycles (though this had been given in the question) or to give answers related to binary trees instead of trees.

Question 6.2 asked students to complete a trace table. This was generally done well with 95% of students getting at least one mark and over a third all seven marks.

Students found question 6.3 more challenging with two-thirds not getting any marks. Students who got one mark normally identified it would be a tree where every branch was in the same direction but did not state that this would be to the left.

About a third of students correctly identified that the algorithm used a stack data structure in their answer for question 6.4. Fewer students were able to describe the changes needed to the algorithm so that the order of the data values output would be reversed (question 6.5); the most commonly seen incorrect answer was to change the starting value of the `Pos` variable.

Question 7

Most students were able to get some marks on this programming question, with many students producing fully-working code – consistent with achievement on past section B questions.

Many students were able to get about half of the marks on this question by writing program code that got, and validated, a number from the user, setting up selection structures to output the type of number and identifying that there would be a need to repeat based on the number of digits in the number entered by the user.

Some common errors were to:

- allow the user to enter a number greater than or equal to zero instead of just greater than zero
- try and get the length of an integer value
- assume that consecutive equal digits meant a number was not increasing/decreasing
- try and compare the last character in a string with the “next” character.

For students who obtained close to full marks the most common error was to write code that did not classify a number where all the digits were the same as being not bouncy (most frequently classifying it as a perfectly bouncy number as the number of consecutive digits that were increasing was the same as the number that were decreasing – zero).

A wide variety of approaches to this question were seen that allowed students to get full marks.

Question 8

Question 8 was about the `Puzzle` class.

All three question parts were answered well by many students. Some students were confused between concatenation and encapsulation in their answers for question 8.1.

Question 9

Question 9 was about the `Cell` class.

For question 9.1 over half of students were able to explain why `IsEmpty` could have been a private method. Fewer students could explain the difference between a local variable and a private class attribute in their answers to question 9.2.

Question 10

Question 10 was about the `CheckForMatchWithPattern` method.

For question part 10.1 about 75% of students could correctly identify a method that used nested iteration.

Students found the remaining parts of question 10 harder than question 10.1.

Most students were only able to give answers that showed a fairly superficial understanding of how exception handling worked in questions 10.2 and 10.3.

Explanations for question 10.4 were often poorly written with some students just stating that this was caused by an error in the program.

Question 11

Question 11 required students to write a validation routine to check that data entered by the user met specified criteria. It was the first of the questions that required students to modify the skeleton program.

Almost all students got some marks for this question with many getting full marks. The most common mistakes were to incorrectly combine the two conditions, to omit the check for the lower bound completely or to allow a lower value of zero.

There were some students who included screen captures not produced by their code. These are never given marks, and their time would have been better spent trying to get their program code to work correctly.

Question 12

Question 12 required students to write program code that calculated the highest and average scores for a series of completed puzzles.

The most common mistakes made were:

- to write code for just one of the two requested calculations
- to always assume that there would only be two completed puzzles when calculating the average value

- to write code that would output the scores before the user had confirmed that they did not want to do another puzzle.

Over half of students got full marks for this question with almost all students getting some of the marks available.

Question 13

Question 13 required students to write a new method that would shift all the cells in a specified row one place to the left.

A significant number of students did not attempt this question even though a number of marks could be obtained by writing relatively simple bits of program code, eg creating a new method with the specified identifier and specified parameter and then calling that method would get two marks. Some of the rest of the code needed in `AttemptPuzzle`, such as decreasing the score and getting the row to shift from the user, was also fairly simple to write.

The most common issues that prevented students who made good attempts at this question from getting full marks were:

- changing the symbols in the cells being shifted instead of actually shifting the cells (the symbol is only one of the bits of data about a cell, the list of symbols not allowed is another)
- not displaying the new score
- only working with a grid of a specific size.

Question 14

Question 14 required students to write a new class that inherited from the `BlockedCell` class.

A significant number of students did not attempt this question even though a number of marks could be obtained by writing relatively simple bits of program code, eg creating a new class that inherited from `BlockedCell` was worth a mark and there was an example of creating a new class that used inheritance already in the skeleton program.

The most common issues that prevented students who made good attempts at this question from getting full marks were:

- not preventing the symbol changing to `-1` when the timer changed again after the symbol had become `@`
- inheriting from `Cell` instead of `BlockedCell`
- incorrect range for random number(s) – normally off by just one, ie starting from zero when it should have been one or a range that would never generate a cell in the first or last position of a row/the grid.

Mark Ranges and Award of Grades

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