AQA

A-LEVEL
COMPUTER SCIENCE

7517/1A-E Paper 1 Options A - E
Report on the Examination

7517/1A-E
June 2023

Version: 1.0

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 as soon as possible after that date so that they have time to study it. Centres must test the Skeleton Program on the computers to be used for the exam and ensure everything will go smoothly in advance of the exam.

There were still many students who provided screen captures that were not produced by their code: students will not receive marks for such screen captures and their time would have been better spent trying to get their program code working 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 may result in long lines of code being only partially visible and so prevent some students from gaining marks.

Question 1

Question 1 was about adding an item to a circular queue implemented as a static data structure.

Most students were aware that a queue is a FIFO data structure and could describe the idea of adding the new item to the position in the array indicated by a rear pointer.

A number of students omitted the check for a full array. Another common error was to confuse the start of the array and the start of the queue when checking if the new item needed to be inserted at the start of the array (rather than at the next position in the array).

Question 2

Question 2 was a logic puzzle.

All three question parts were generally well-answered by students. There was no requirement for students to explain their answers to this question; students who did this were not making effective use of their time in the exam.

Question 3

Question 3 was about binary trees, binary search and time complexity.

Question part 3.1 required students to state two characteristics of binary trees. The most common error was to give properties of trees in general that did not differentiate binary trees from other trees.

For question part 3.2 students had to complete the gaps in a binary tree search algorithm. Students found it harder to give code for the parts labelled 4 and 5 than for the other parts.

Question parts 3.3 and 3.4 were generally well-answered.

Question part 3.5 asked students to explain why searching in a list is always a tractable problem. Many students were able to explain clearly the idea that there were algorithms that could do this in a polynomial time complexity meaning that the problem was tractable. An equally valid answer was that it was solvable by algorithms that had a better than exponential time complexity.

Answers for question part 3.6 were often vague and lacked the detail needed for two marks. Some answers were clearly describing decomposition rather than the use of heuristics.

Many students gave answers for question part 3.7 stating that constant time complexity meant that the time complexity for the algorithm, rather than the time taken by the algorithm, did not change.

Question 4

Question 4 was about regular languages and sets.

Question part 4.1 gave examples of languages represented using syntax diagrams, a regular expression, set notation and Backus-Naur Form (BNF).

Most students recognised that Language B, represented by a regular expression, was a regular language but the other languages were more challenging to categorise correctly.

Question part 4.2 asked students to represent a language represented by the union of two given sets as a regular expression. Not many students were able to give a fully correct answer but a number of students were able to describe one of the two sets using a regular expression. Some students gave their answers in either set notation or BNF rather than as a regular expression as asked for in the question.

Question part 4.3 was generally well-answered. A common misconception was to state in answers that an infinite set cannot have a cardinality (countable infinite sets do have a cardinality, uncountable infinite sets do not).

Question 5

Question 5 was about Turing machines.

For question part 5.1 students were required to trace the operation of a Turing machine and many students did well on this question. Some students did not include the current state in their answers and some made mistakes when indicating the position of the read/write head.

Not many students were able to give a good answer to question part 5.2. A common error was not to read the question carefully and give one or more answers that had already been provided in the question.

Question parts 5.3 and 5.4 were about theoretical concepts related to Turing machines. Answers for question part 5.3 were not always very precise, and while many students were able to demonstrate some understanding of the topic few were able to give a two mark description of a Universal Turing Machine. For question part 5.4 a higher proportion of students could describe why a UTM could be considered more powerful than any computer, recognising that a UTM has infinite memory which is impossible for a real computer.

Question 6

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 a string from the user, checked the length of the string, set up iteration to repeat until the string was valid, iterate through every character in the entered string and check if two characters in the string were the same.

Some students wrote programs that had incorrect conditions for checking the length, often by using `AND` instead of `OR` to combine the two conditions or by excluding lengths of 5 and 7.

Other common errors were:

- not using the correct function to get the ASCII code for a character
- not resetting variables used with validation checks at the start of the loop meaning the program would only work correctly for the first iteration
- only checking if consecutive characters were the same as each other instead of checking if a character was the same as any other character in the string
- checking if all letters were upper case instead of checking that all characters were upper case letters

Some creative answers to this question were seen that showed good problem solving and programming skills. One of the more frequently seen examples of this was using dictionaries when checking for duplicate characters.

Question 7

Question 7 was about the `PlayGame` method. Both question parts were answered well by many students.

Question 8

Question 8 asked about the difference between virtual and abstract methods. This topic has not been asked before and very few students were able to demonstrate any understanding of it. Many students either did not attempt this question or gave answers related to abstraction rather than abstract methods.

There were a number of students who got the two types of method the wrong way round and gave answers such as "Virtual methods must be overridden whereas abstract methods can be overridden".

Question 9

Question 9 was about the `MoveOptionQueue` class.

For question part 9.1 the most commonly-seen correct answer was to state that the `Queue` attribute had been made private.

Many students were able to answer both question parts 9.2 and 9.3 correctly. A common misconception for question part 9.2 was to state it was not being used as a FIFO data structure because an item would be added to the end of the queue after it had been removed.

Question 10

Question 10 was about the `CheckIfGameOver` method.

For question part 10.1 students were required to explain why the first selection structure was needed. Many good answers were seen for this question part but some students explained larger parts of the method and did not make clear the purpose of that particular selection structure.

Question part 10.2 was also well-answered. A number of students did not read the question carefully and gave answers that used `AND`, `NAND` or `XOR`.

Question 11

Question 11 asked students to state the names of identifiers from the Skeleton Program.

Many students got both parts correct. The most common error was to include additional code and not just the identifier.

Question 12

Question 12 was about the `CreateMoveOptions` method.

Many students were able to explain the reason why `-1` was used but some answers were too vague to get the one mark available.

Question 13

Question 13 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, not to include an error message, to use selection instead of iteration or only to check the upper boundary.

Some students got full marks for developing correctly working solutions that used recursion instead of iteration and some used exception handling instead of setting up validation checks.

Question 14

Question 14 required students to write a new method that randomly swapped the locations of five pairs of squares in the game.

The most common mistakes made were:

- generating random numbers in an incorrect range (often a range of 35, rather than 36, values)
- omitting the check that the two random numbers were not the same
- changing the pieces in the two selected squares instead of changing the positions of the two squares within `Board`

A number of ways of generating the random numbers were seen including getting four numbers between 1 and 6 then combining them into square references, generating random indexes between 0 and 35 and selecting random items from the `Board` data structure.

Question 15

Question 15 required students to write a new class that inherited from the `Square` 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 code, eg creating a new class that inherited from `Square` was worth a mark and is something already done for the `Kotla` class in the Skeleton Program.

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

- not returning the correct value from `GetPointsForOccupancy` under the correct circumstances
- adding the new `Gacaka` square to `Board` as an additional square instead of as a replacement for one of the other squares (ie having 37 items in `Board`)

Question 16

Question 16 required students to write a new method that calculated how many possible moves there are for any given state of the game.

As with question 15 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 code, eg creating a new method with the specified identifier and specified parameter and then calling that method would get two marks. Having the method return an integer and ensuring that the returned value was displayed would get a third mark.

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

- not checking if the square a piece could be moved to contained a piece belonging to the current player
- having conditions on iterative structures which meant that moves involving the last position in `Board` would not be included

Mark Ranges and Award of Grades

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