GCSE
COMPUTER SCIENCE

8520/1: Computational thinking and Problem Solving
Report on the Examination

8520
June 2019

Version: 1.0

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

Introduction

This year saw the second set of examinations for the reformed GCSE Computer Science specification, with some areas being examined for the first time. On the whole, students seemed well prepared for this examination paper and showed good application of the problem solving and computational thinking skills they have been taught.

Once again key points to take forward from this year's examination paper would be to ensure students read the question carefully as some responses indicated that this was not always happening.

Mistakes in the use of flowchart symbols are normally ignored, except where they make the algorithm unclear, for example unannotated arrows emerging from a decision symbol.

Another issue which was evident this year concerned the use of arrows in pseudo-code for assignment statements when some students reversed the assignment flow, i.e., `variable → USERINPUT`.

A number of students used ← for equality as well as assignment. As this is not clear and unambiguous the marks cannot be awarded. It was good to see that many pseudo-code solutions showed good use of indentation which frequently made the algorithms easier to read and follow. This is to be encouraged.

Question 1

This question was about string manipulation and ASCII/Unicode. Although it was multiple choice it was pleasing to see over half of the students getting these questions correct, the exception being question 1.3 where most students were unable to identify a substring.

A small number of students left some of these questions unanswered.

Question 2

This is the first time we have seen an in-depth understanding question on types of programming language and it is pleasing to see that some centres have taught this topic well.

Question 2.2 was a four mark question on the difference between high and low level languages. Over 80% of the students were able to get one mark with over 50% getting half marks. The most common correct answers given were that high-level languages were easier to maintain/understand.

It was disappointing that less than half of the students were able to identify either a compiler or an assembler.

Question 3

Question 3 was the first of the algorithms on the paper and on the whole was answered well by students. The multi-choice questions were answered correctly by over 80% of the students and nearly 70% gained 2 marks on the algorithm. The biggest point that was missed was (as in previous series) the indefinite loop with lots of students using a selection statement instead thus

limiting the marks. Another common issue was students who did not use the variable name they were given in the question.

Question 4

Question 4 focused on another algorithm and contained the only trace table for this paper.

This was well answered with over half the students showing that they can trace through code thoroughly enough to gain full marks and over 90% of students gaining at least one mark. Over 80% of the students were able to identify the input and outputs but fewer were able to identify the constants or all three pieces of code in the algorithm.

One point noticed by the examiners was students adding spaces into the variable names which was not accepted as a correct answer.

Question 5

Question 5 was on logic gates and truth tables.

Students showed a good understanding of the logic gates with about 80% being able to correctly name them. The specification clearly shows the correct logic symbols that students should be using when drawing logic gates. Marks were not awarded if the drawings were incorrect or missing input and output lines. In a lot of cases it was difficult to tell whether the drawing was of an AND or an OR gate.

The truth table was completed fairly well with over 60% gaining full marks and over 85% picking up at least one mark.

Question 6

Question 6 looked at a program for run length encoding.

Question 6.6 was well answered with over 80% of students getting full marks.

For Question 6.7 students were asked to state 3 further tests that the developer could use. This question was not answered well with only just over 20% gaining full marks. The most common incorrect answer talked about the different ways of testing which didn't answer the question or gave three tests which fell under the same mark scheme point.

Question 7

Question 7 was another question focused on a piece of code.

As noted in the examiner’s report for question 2 on the 2018 paper the first two parts were answered well and students showed a good understanding of key terms. However, more than two thirds were unable to trace through the algorithm for part 7.3; it is clear that students need to practise working through this type of question. Less than half could identify both series of inputs needed for question 7.4.

With question 7.5, although many students were able to identify that the algorithm asked for 'yes' and 'no' they were not able to clearly explain what happened in the code with a lot referring to the program crashing (which it doesn't). Only just over 30% got two or three marks for this question.

Question 8

Question 8 was the first time that the binary search algorithm has been tested in depth.

This question showed a gap in knowledge for a lot of the students as those that answered the question using the correct algorithm were able to gain full marks by simply stating the three comparisons (21, 31, 27). Over 55% did not gain any marks. The biggest mistakes made by students were describing a different type of search (e.g. linear) and not understanding what happens when the result isn't found.

Over 75% of students were able to identify that the list needed to be sorted for the search to work correctly.

Question 9

Question 9 was the second of the algorithm design questions and the first time, on the current specification, that a question has required an answer in pseudo-code. The specification clearly requires all students to be able to express algorithms in both pseudo-code and flowcharts. As this was the first occasion where an explicit method of response was required answers given as flowcharts were credited and next summer the following statement will appear on the front cover: “Unless the question states otherwise, you are free to answer questions that require a coded solution in whatever format you prefer as long as your meaning is clear and unambiguous.”

From the responses seen there was evidence that students did not have a good understanding of 2D arrays/lists. Additionally, marks were lost by poor indexing methods and for also incrementing variables in the wrong place.

Examiners noticed an increase in the number of students reproducing the code given to them in the question. This is not worthy of credit and wastes time in the exam. Students should be encouraged not to reproduce the code given to them.

Question 10

Question 10 required students to understand a piece of given code and then to write an algorithm.

As with question 7 it was encouraging to see that the first two parts were answered well and students showed a good understanding of key terms (over 75% being able to identify datatypes and decomposition). However, 60% were unable to state a more suitable variable name with some students simply stating another letter, but over 50% were able to state at least one property of a local variable.

In question 10.5 students were asked, for the first time in this specification, to create an algorithm for a sub-routine. It was tackled very well with over 30% getting six marks or more and over 75% gaining at least one mark.

A small number of students did not attempt the question. Teachers should encourage students to attempt every question as often marks can be gained for straightforward variable assignment and

Boolean conditions; in this question one mark could be gained for declaring a subroutine (with a suitable name) and a second for passing in the given parameter. An example was given at the beginning of the question. A tip for students might be to tick off each item on the list when they have included it in their answer as the list of requirements is very concise.

Use of statistics

Statistics used in this report may be taken from incomplete processing data. However, this data still gives a true account on how students have performed for each question.

Mark Ranges and Award of Grades

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