Course Report 2018

Subject	Computing Science
Level	Higher

This report provides information on the performance of candidates. Teachers, lecturers and assessors may find it useful when preparing candidates for future assessment. The report is intended to be constructive and informative and to promote better understanding. It would be helpful to read this report in conjunction with the published assessment documents and marking instructions.

The statistics used in this report have been compiled before the completion of any Post Results Services.

Section 1: comments on the assessment

Summary of the course assessment

Component 1 — question paper

The question paper was worth 90 marks — 20 marks for section 1 and 70 marks for section 2.

Approximately 50% of the marks were allocated to ‘software design and development’ and the remainder to ‘information system design and development’.

Feedback from the markers was that candidates were better prepared this year. The full range of marks was awarded in all questions and the question paper sampled appropriately from all areas of the course, as outlined in the course assessment specification.

Based on statistical data and feedback from the markers, the question paper was more demanding than intended. This was taken into account when setting the grade boundaries.

Component 2 — assignment

The assignment was worth 60 marks.

The average mark for the assignment has slightly decreased, however, the marks for the assignment were still higher than expected, compared with candidates’ performance in the question paper. This was taken into account when setting the grade boundaries.

Section 2: comments on candidate performance

Areas in which candidates performed well

Component 1 — question paper

Question 1	The majority of candidates converted the denary value to two's complement.
Question 2	This question was answered well by most candidates.
Question 3	Most candidates named a feature of a declarative language, however, their description did not indicate any depth of understanding.
Question 5	Most candidates understood the advantage of Unicode over ASCII.
Question 9(a)	Most candidates named a trend that improves performance in modern computer architecture.
Question 11(d)(i)	Most candidates could identify a bitmap graphic when pixelated.
Question 12(b)(i)	The quality of candidate responses when defining a record structure has improved significantly.
Question 12(c)	Most candidates gained a considerable amount of marks for writing the algorithm required.
Question 13(c)	Many candidates managed to get at least 2 out of 3 marks for this question.
Question 15(e)(ii)	Most candidates could identify the type of programming error.

Component 2 — assignment

The assignment was well attempted. Most of the candidates from the sample performed extremely well in the implementation sections of the assignment and produced excellent computer programs, databases and webpages.

Areas which candidates found demanding

Component 1 — question paper

Question 3	Few candidates could describe a feature of the declarative language they had named.
Question 4	This was the first time this area of content had featured in the question paper. It was evident that very few candidates understood RAID.
Question 8	Few candidates could provide the missing steps of the fetch-execute cycle, using the appropriate technical language of the subject.

Question 11(a)(i)	Most candidates could not compare public cloud and private cloud. Many just expressed an advantage of a cloud storage and not the benefits of public cloud compared to private cloud.
Question 11(b)	Many candidates did not gain the second mark, as they did not mention username, along with password.
Question 11(c)(ii)	Most candidates did not have a good understanding of interframe compression and lacked the ability to express it well.
Question 11(d)(ii)	Many candidates struggled to explain this, even those who understood RLE.
Question 13(d)	Few candidates clearly understood how to use server-side scripting to query a remote database. Most knew that a query is run, but did not understand, for example connection to a server or selecting a database.
Question 14(e)	Very few candidates gained this mark.
Question 15(b)	Few candidates could identify formal and actual parameters, and even fewer understood the relationship between them.
Question 15(c)	Generally, candidates do well when asked about a local variable, however, they struggled when asked about it using the term ‘scope’.
Question 15(d)	This question was deliberately challenging. Candidates found it more challenging than in previous years, because it went deeper into the algorithm, and only required values when triggered by the watchpoint.
Question 15(e)(iii)	Most candidates did not understand or explain the reason for the correct output with an incorrect code.
Question 15(g)	Very few candidates clearly understood the role of memory management within the operating system.

Component 2 — assignment

Not all candidates could implement the program design stage correctly. They frequently omitted the user interface design, although candidates may have thought that their programming language had no user interface. All they needed to show were the inputs and outputs of the program.

Some candidates struggled with data-flow in programming design. This is often confused with the requirements for their programming language or with inputs and outputs to the program. Candidates only need to understand that data-flow shows which variables are passed into the sub-program and those that need to be passed out of the sub-program.

Candidates did not always remember to provide evidence of all their testing.

Candidates should show all the validation information in the data dictionary; it is not enough just to name the type of validation.

Many candidates did not provide all the required evidence for implementing the information system. Centres should remind candidates to use the checklist to ensure that they submit everything.

Section 3: advice for the preparation of future candidates

Component 1 — question paper

Candidates' written code continues to improve and it was clear that candidates knew how to use records and 1D arrays of records. Candidates were also better prepared for writing algorithms in the context of a 1D array of records. They are also improving their understanding of CSS and subsequent writing of CSS rules. Centres should continue to help candidates develop problem-solving skills in a wide variety of unfamiliar contexts.

Candidates often struggle to provide full explanations of concepts, for example compression techniques such as interframe and RLE. Centres might consider strategies to help candidates express themselves when providing explanations and descriptions.

Although writing code is improving, candidates still find understanding unfamiliar code challenging and struggle to use the appropriate subject terminology and vocabulary. Centres should ensure candidates develop problem-solving skills through understanding unfamiliar code. In particular, using actual and formal parameters, and the scope of variables.

Candidates should take care reading the questions and ensure that they provide full explanations, when required. For example if asked to compare public and private cloud, they need to provide more than just the advantage of cloud storage.

Component 2 — assignment

For 2018–19, the assignment changes from an internally assessed bank of tasks to a single assignment, issued annually. This is externally assessed by SQA.

A specimen assessment task is available on the Higher Computing Science subject page on SQA's website and centres can use this to help prepare candidates for the assignment.

Grade boundary and statistical information:

Statistical information: update on courses

Number of resulted entries in 2017	4476

Number of resulted entries in 2018	4099

Statistical information: performance of candidates

Distribution of course awards including grade boundaries

Distribution of course awards	Percentage	Cumulative %	Number of candidates	Lowest mark
Maximum mark				
A	20.5%	20.5%	840	107
B	23.4%	43.9%	959	91
C	24.8%	68.7%	1017	75
D	12.5%	81.2%	511	67
No award	18.8%	-	772	-

General commentary on grade boundaries

SQA's main aim is to be fair to candidates across all subjects and all levels and maintain comparable standards across the years, even as arrangements evolve and change.

SQA aims to set examinations and create marking instructions which allow a competent candidate to score a minimum of 50% of the available marks (the notional C boundary) and a well prepared, very competent candidate to score at least 70% of the available marks (the notional A boundary).

It is very challenging to get the standard on target every year, in every subject at every level.

Therefore SQA holds a grade boundary meeting every year for each subject at each level to bring together all the information available (statistical and judgemental). The Principal Assessor and SQA Qualifications Manager meet with the relevant SQA Business Manager and Statistician to discuss the evidence and make decisions. The meetings are chaired by members of the management team at SQA.

- The grade boundaries can be adjusted downwards if there is evidence that the exam is more challenging than usual, allowing the pass rate to be unaffected by this circumstance.
- The grade boundaries can be adjusted upwards if there is evidence that the exam is less challenging than usual, allowing the pass rate to be unaffected by this circumstance.
- Where standards are comparable to previous years, similar grade boundaries are maintained.

Grade boundaries from exam papers in the same subject at the same level tend to be marginally different year to year. This is because the particular questions, and the mix of questions, are different. This is also the case for exams set by centres. If SQA alters a boundary, this does not mean that centres should necessarily alter their boundary in the corresponding practice exam paper.