# Course report 2019

| | |
|---|---|
| Subject | Advanced Higher |
| Level | Computing Science |

This report provides information on candidates' performance. 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

## Project

For the first time, the project was externally marked at a central marking event. The average project mark decreased from 2018, giving a distribution of marks for the project more in line with the question paper.

Feedback from the marking team suggests that many candidates had failed to adhere to the candidates' guide for the project. In particular, most candidates ignored the need to make regular reflective commentary and simply provided a diary that listed tasks completed.

A reflective commentary is required at every stage of the development process. Failing to provide evidence of this had an impact throughout the project as maximum marks could not be awarded at every stage.

The process-based aspects of the project, in particular the reflective commentary, were previously supported by internal assessment. Externally marking a task that was designed to be internally assessed meant the external marker could only mark the evidence available. Previously, the internal assessor would be best placed to make a more qualitative judgement on the candidate's work, reflecting on regular meetings between candidate and assessor as well as the evidence generated by the candidate. This was taken into account when setting the grade boundaries.

## Question paper

The question paper performed largely as expected. Feedback from the marking team suggested that the question paper was accessible to candidates and fair in terms of course coverage and the overall level of demand.

The majority of candidates were able to demonstrate knowledge and understanding from all areas of the course content.

# Section 2: comments on candidate performance

## Areas that candidates performed well in

### Project

**Project plan**

Most candidates produced an outline project plan that listed each stage of the proposed development and indicated estimated time scales. In addition, most candidates listed the resources that would be required to complete the work at each stage of the development.

**Requirements specification**

A few candidates made use of a UML case diagram to help them identify user requirements that resulted from their initial proposal. Although this was not necessary, it did lead to clearer descriptions of the processes that users would expect to find in the final solution.

**Interface design**

The majority of candidates used wireframes to show the intended layout of the inputs and outputs required in their solution. Candidates who received high marks also annotated these wireframes to indicate where validation would be carried out, and any underlying processes that would be executed as a result of input.

**Program/data structure design**

A few candidates who produced a web-based solution had provided a hierarchical diagram to show the intended structure of the pages within the website. This was extremely helpful to markers as it clearly indicated how the pages would be linked and how information would be passed from one page to another. This evidence enabled markers to award marks for data flow.

**Implementation**

The majority of candidates produced very good work at the implementation stage of their project development. However, earlier planning work did not always support this.

**Final testing**

In most cases, final testing was completed well. Candidates carried out each of the tests listed in their test plan and provided evidence to demonstrate that the test had been successful.

### Question paper

Question 1(a) The majority of candidates received 1 mark for showing a declaration of the WorkMeeting class that indicated the correct data types for all instance variables additional to those in the Event super class. Many candidates received a second mark for correctly indicating in their code that the WorkMeeting class is a subclass of the Event class.

Question 1(d)(i) Most candidates received 1 mark for the correct use of the relevant instance variables needed in the calculation.

Question 1(e) The majority of candidates received at least 1 mark for an accurate description of any relevant environmental implication.

Question 2(a) Candidates demonstrated good knowledge of the action and method attributes of the HTML <FORM> element, and made appropriate use of the information provided in the question to complete the code correctly.

Question 2(d)(i) Most candidates received 1 mark for correctly assigning connection details to server-side variables, with many candidates receiving a second mark for showing how those details were used to form a connection with the database server.

Question 2(d)(ii) Most candidates received 1 mark for correct use of an SQL insert query used to add the tour details to the Tour table.

Question 2(e) The majority of candidates received 1 mark for an accurate description of any relevant legal implication.

Question 3(a)(ii) Candidates demonstrated a good understanding of the binary search algorithm, with the majority receiving at least 1 mark for indicating the correct update when the required towelID is found. However, markers noted that many candidates used the term ‘middle’ instead of ‘middle element’.

Question 3(a)(iii) Candidates demonstrated good knowledge of the use of recursion in this situation.

Question 3(c) The majority of candidates received 1 mark for accurate descriptions that demonstrated their knowledge of the need to update the pointers in the linked list when a new towel is added. However, very few candidates made use of the term ‘node’, and therefore lacked the required technical terminology to gain full marks.

Question 3(d) Most candidates received 1 mark for descriptions of beta testing that accurately referred to end users, and the feedback that they provide to the developers.

Question 4(a) The majority of candidates demonstrated good knowledge of use case diagrams, receiving 1 mark each for correctly indicating the actors and use cases required for the processes outlined in the question stem.

Question 4(c) The majority of candidates made a reasonable attempt to allocate the stated initial values for each seat, receiving at least 1 mark for the assignment of the default ‘Empty’, ‘Null’, 0.0 and false values.

Question 4(d) Candidates demonstrated good problem-solving skills in their attempts to allocate the next available seat. Many candidates received marks for incomplete solutions which did meet some of the stated requirements.

## Areas that candidates found demanding

### Project

**Reflective commentary**

Throughout the project, candidates were expected to reflect on work completed and make notes of changes that were necessary as a result of that reflection. Most candidates failed to do this and instead provided a list of work completed with no reasons given for any changes deemed necessary. Reflective commentary is required at every stage of the development process. Failing to provide evidence of this meant that marks for reflective commentary could not be awarded.

**Project plan**

Several candidates failed to provide a clear statement of their proposed development that indicated the Advanced Higher techniques to be included.

Although most candidates did produce a good outline project plan, detailed project plans often failed to list individual tasks that would need to be carried out to complete the work of each section. As a result, full marks for this aspect of the work could not be awarded.

**Requirements specification**

Despite the fact that requirements stated at this stage are integral to the rest of the project, many candidates failed to clearly describe the inputs, processes and outputs that the development would require. Many candidates failed to describe the essential process needed to validate all inputs within the context of their own development. In addition, many candidates failed to describe the processes that would make use of the two Advanced Higher techniques.

Failure to provide detailed descriptions of user requirements together with the functional input, processes and output requirements meant it was often not possible to award full marks for this aspect of a candidate’s work.

**Test plan**

Many candidates failed to refer to all the inputs, processes and outputs listed in the requirements specification in their test plans. In addition, some candidates failed to include any tests to ensure that all inputs had been validated. These omissions meant it was not possible to award full marks in this section.

**Interface design**

Although wireframes had been used to show intended layouts, a significant number of candidates omitted some of the inputs and outputs listed in the requirements specification. Full marks for this section could not be awarded where candidates had omitted necessary layouts or essential validation details.

Candidates who submitted screen shots of their final implementation labelled ‘Interface Design' received no marks.

**Program/data structure design**

A significant number of candidates did not make any attempt to indicate data flow between modules of their solution or pages of their website. This omission meant it was not possible to award full marks for this area of the work.

The majority of evidence submitted for the design of the program or server-side script was simply code that had been copied and then numbered. Since numbered code is not a recognised design technique, marks could not be awarded for this evidence.

Where candidates had attempted to describe each of the processes listed in the requirements specification in structured English, it was possible to award marks.

Many candidates whose solution required a database failed to provide a data dictionary or any query designs. No marks could be awarded where this essential evidence was missing.

### Implementation

In addition to the full code implemented in the solution, candidates were expected to provide evidence of ongoing testing. Many candidates failed to include any such evidence and therefore did not receive full marks for the implementation work.

### Final testing

Although the majority of candidates did provide good test evidence, very few commented on the results of testing. In particular, most candidates did not consider corrective maintenance and therefore did not achieve full marks.

### Evaluation

In many cases, evaluations lacked sufficient detail to receive marks at Advanced Higher level. When evaluating their solution, candidates are expected to refer to their requirements specification and provide detailed qualitative comments that explain how closely their solution matches the initial requirements. Since many evaluations were superficial, it was not possible to award full marks for this aspect of the project work.

## Question paper

Question 1(b) The majority of candidates were unable to explain the use made of the Constructor method. It was clear that many candidates understood the use made of the method, but were unable to use appropriate object-oriented terminology to explain its use.

Question 1(d)(ii) Although the majority of candidates made a good attempt at calculating the total travel expenses, most ignored the fact that instance variables are encapsulated and therefore cannot be accessed directly; instead, methods must be used. As a result, very few candidates received the mark available for making use of methods.

Question 2(c)(i) Very few candidates were able to demonstrate knowledge of the SQL aggregate COUNT() function and the need for a GROUP BY clause in a SELECT query.

Question 3(a)(i) Many candidates provided explanations that made use of bubble sort or selection sort, rather than the insertion sort the question required.

Question 4(b)(ii) The majority of candidates ignored the instruction to define a 2D array to store the seat data. Instead, most responses made use of an array of records.

Question 4(e) Most candidates provided very generic descriptions of deadlock, concurrency or resource starvation and failed to apply that knowledge to the scenario to discuss a problem that could potentially arise as a result of using multiple threads.

Question 4(f) The majority of candidates either named the incorrect type of maintenance or failed to provide an accurate description of adaptive maintenance.

# Section 3: preparing candidates for future assessment

### Project

Centres must ensure they are using the correct version of the coursework assessment task. It was clear in the submissions that a significant number of centres had been referring candidates to guidelines issued for the former Advanced Higher Computing or Information Systems courses. This meant that candidates spent a lot of time completing work that was unnecessary, and received no marks.

Centres should advise candidates on their choice of problem to ensure that candidates' chosen problem meets the criteria for an Advanced Higher project and is achievable. From session 2019-20, the project must allow the candidate to demonstrate at least two Advanced Higher techniques from their chosen area of development and integration with at least one other area of the course content. Projects that are overly simplistic will fail to meet these essential criteria and cannot receive the full range of marks available in each section of the project. Similarly, projects that are overly ambitious may make it extremely difficult for candidates to complete all sections of the project to the required standard.

Centres must ensure that all candidate evidence is submitted to SQA in paper-based format. This includes all aspects of a candidate's implementation. Prior to submission, it would be extremely helpful if centres could ensure that all candidate evidence is presented in the same sequence as the marking instructions. Centres should encourage candidates to annotate their code to highlight where in the solution the Advanced Higher techniques have been applied, and where integration with at least one other area of the course content has been achieved. The revised coursework assessment task, valid from session 2019-20, includes a candidate checklist. This evidence checklist details all evidence to be gathered and centres should encourage candidates to use it to ensure they submit all necessary evidence to SQA.

Centres must ensure that the development environment candidates use to implement their solution provides the structures and facilities needed to demonstrate the essential Advanced Higher techniques and integration required in candidates' solution. Several project submissions included implementation evidence generated using tools such as GameMaker and Unity. This evidence had not been annotated to indicate the parts of the solution developed by the candidates themselves. As a result, it was very difficult for markers to award marks for the two Advanced Higher techniques that are necessary for any Advanced Higher project.

Candidates must independently generate all project evidence they submit. Centres are advised to meet regularly with individual candidates to authenticate their evidence and ensure the implementation is indeed their own work.

### Question paper

It was encouraging that, overall, candidates engaged well with problem-solving questions that required them to cope with the challenges of unseen programming tasks.

Centres should continue to ensure that candidates have opportunities to apply the standard algorithms for binary search, bubble sort and insertion sort to a range of data structures to solve this type of problem-solving task. The data structures should include all data structures introduced at Advanced Higher level: 2D arrays, objects and array of objects. This work is essential preparation for Advanced Higher candidates and can take the form of practical work and written problem-solving tasks.

Centres should note that the binary search algorithm provided in Question 3(a)(ii) made use of the variables startPointer, endPointer and middle. Many candidates did not realise that these variables are indices used to identify individual array elements rather than individual array elements. As a result, the conditions in their comparisons were incorrect. Centres should encourage candidates to pay closer attention to the use of any variables used in standard algorithms in the question paper.

Centres should ensure that candidates are able to state the correct data types required to store data. In database design and development, candidates are expected to make use of appropriate SQL data types. In software design and development, candidates are required to do this in a programming language of their own choice. Responses in languages that use dynamic typing, such as LiveCode, JavaScript, PHP and Python, are acceptable but centres should encourage candidates who use these languages to add internal commentary to their code to indicate the data type that is intended.

Candidates should be familiar with the specialist features of object-oriented programming languages and should be able to use appropriate object-oriented terminology to explain the operation and effect of code written in the SQA Reference Language. It was clear that candidates had knowledge and awareness of this terminology, but were often unable to apply that knowledge appropriately. It was also clear that many candidates failed to read any of the code in lines 1 to 21 at the start of the question. For example:

Question 1(c)(i) Although candidates could identify that a new object called sales, that belongs to the WorkMeeting subclass, had been instantiated, they failed to acknowledge that any such object would be assigned the default values at lines 8 and 9 in addition to all the values stated in the DECLARE statement.

Question 1(c)(ii) Most candidates were able to identify that the addParticipants() method of the sales object had been invoked, but hadn't read the code in lines 17-20. As a result, most responses made no mention of the array index which would be incremented by the code in this question.

Question 1(d)(ii) In their responses to Question 1(c)(iii), it was clear that most candidates understood that encapsulation had been used to protect instance variables of the sales object. This had been indicated in the UML class diagram.

However few candidates made use of the getTravelExpenses() and getDate() methods to access the encapsulated instance variables travelExpenses and date, which were necessary to perform the calculation required.

Centres should ensure that candidates are familiar with all SQL statements required at Advanced Higher level. In addition, candidates should be aware that all SQL statements can be executed using server-side script.

Candidates must pay more attention to the wording of questions. In several questions, candidates ignored important details or instructions that had clearly been stated. For example:

Question 1(d)(ii) Several candidates ignored the need to calculate the total expenses for April.

Question 2(a) Some candidates failed to pay attention to the requirement to submit details securely (method attribute) and include the name of the script (action attribute).

Question 2(b) The form used to gather the Tour details indicates the mandatory fields of the Tour table. In addition, the question refers to the tourID field which uses auto increment. Some candidates failed to make use of these facts and, as a result, omitted important details from their code.

Question 3(a) The question clearly asks candidates to explain how insertion sort algorithm could be used. Despite this, many candidates provided descriptions of how bubble sort or selection sort would be used.

Question 4(b)(ii) Candidates were asked to declare a suitable 2D array. The majority of candidates ignored this instruction.

Question 4(f) Many candidates failed to read the instruction to ‘name and describe’ the type of maintenance required.

# Grade boundary and statistical information:

## Statistical information: update on courses

| | |
|---|---|
| **Number of resulted entries in 2018** | 636 |

| | |
|---|---|
| **Number of resulted entries in 2019** | 614 |

## 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** | 24.1% | 24.1% | 148 | 101 |
| **B** | 20.7% | 44.8% | 127 | 84 |
| **C** | 20.7% | 65.5% | 127 | 68 |
| **D** | 9.9% | 75.4% | 61 | 60 |
| **No award** | 24.6% | - | 151 | - |

## 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 that allow:

- a competent candidate to score a minimum of 50% of the available marks (the notional C boundary)
- 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 head of service and statistician to discuss the evidence and make decisions. Members of the SQA management team chair these meetings. SQA can adjust the grade boundaries as a result of the meetings. This allows the pass rate to be unaffected in circumstances where there is evidence that the question paper has been more, or less, challenging than usual.

- The grade boundaries can be adjusted downwards if there is evidence that the question paper is more challenging than usual.
- The grade boundaries can be adjusted upwards if there is evidence that the exam is less challenging than usual.
- Where standards are comparable to previous years, similar grade boundaries are maintained.

Grade boundaries from question 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 question papers set by centres. If SQA alters a boundary, this does not mean that centres should necessarily alter their boundary in the question papers that they set themselves.