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: project
The project is worth 90 marks.

The average project mark has slightly decreased from 2017. The number of candidates gaining very high marks has also slightly decreased from last year.

Component 2: question paper
The question paper is worth 60 marks. It has one section, with four questions worth 15 marks each.

The question paper largely performed as expected. It was accessible to candidates, and fair in terms of course coverage and overall level of demand. Most candidates understood what was required and completed all four questions in the allocated time.

Many candidates dealt with all areas of content covered in the question paper. There were several very strong responses, with those candidates demonstrating very sound knowledge of the content examined.

Although there were some areas of significant improvement in candidate performance, overall, candidate responses were below the standard expected at Advanced Higher level. Some candidates who did not perform strongly in the question paper, performed significantly better in the project.

Question 1(c)(i) did not function as intended. This was taken into account when setting grade boundaries.
Section 2: comments on candidate performance

Areas in which candidates performed well

Component 1: project
Most candidates showed excellent programming skills in a wide range of projects, which they demonstrated to a very high standard.

Component 2: question paper

Question 1(b)  Most candidates accurately explained why a queue structure would be suitable in the given situation.

Question 1(c)(ii)  Most candidates gained at least one mark for this question, by comparing the versatility of a linked list with a 1D array, when adding and removing items.

Question 1(d)  Most candidates demonstrated sound knowledge of the binary search algorithm. In particular, they used an appropriate complex condition with their conditional loop and used the selection appropriately to update the upper/lower search limits.

Question 1(e)  Candidates described an appropriate ethical implication of the increasing use of intelligent systems in driverless cars.

Question 2(c)  Most candidates gained one mark for correctly assigning the form details to the server-side variable. In addition, most candidates demonstrated correct use of selection to compare the two passwords entered and to display an appropriate error message when they did not match.

Question 3  Most candidates performed well, demonstrating good knowledge of object-oriented terminology. Candidates applied this knowledge to the code provided and also made appropriate reference to the UML class diagram. Responses to parts (a), (b), (c) and (e)(ii) were especially good.

Question 4(a)(i)  Most candidates defined an appropriate record structure to store the applicant data.

Question 4(b)  Most candidates demonstrated some knowledge of the bubble sort algorithm. They did this using an inner loop to compare adjacent last names and swap them correctly when the last names were out of sequence.
Areas which candidates found demanding

Component 1: project
Candidates did not perform at Advanced Higher level in various aspects of the project. For example, many projects had test plans that only included normal, extreme and exceptional testing and evaluations that only mentioned evaluation criteria (for example robust, reliable) which are from the Higher course. Candidates need to ensure that their project addresses the requirements of the Advanced Higher course.

It is important to stress to candidates that the record of progress and reflective commentary must be continually updated, as marks are available for this at each stage of the project. This can be detailed within the section or in the record of progress and should be distinct from the evaluation. Although multiple versions of a document show iteration, meaningful comments saying why changes/decisions were made must be included to make them reflective.

Candidates should ensure that they design the software before they write their code, and that their pseudocode is numbered and shows the main steps and refinements. Other design notations can be used, for example data flow diagrams, flow charts or structured diagrams. When databases/websites are implemented, candidates should include data dictionary, ERDs and query designs. Interface design should show an outline of the interface and not fully illustrated screens created within the implementation software.

Candidates were not always clear about the difference between the test plan and the final testing. The test plan should show a structured plan that will be used to test the software, but at this stage, should not show any testing. Screenshots of a working program should be used as evidence of final testing, not just statements saying that everything works.

Component 2: question paper

Question 1(c)(i) Most candidates did not check for stack overflow before adding a new instruction to the stack. In addition, candidates ignored the need to increment the stack pointer.

Question 1(e) Few candidates described a suitable social implication of the increase of intelligent systems in driverless cars. Most answers were very vague and did not refer to any change or impact on society.

Question 2(a) Although many candidates gained one mark for correctly identifying the primary key within their SQL statement, most candidates did not use appropriate SQL data types for the fields listed. The SQL data type text, varchar or char should be used rather than string.

Question 2(b) Although the HTML form code written by most candidates was accurate and demonstrated they could use the type attribute of the <INPUT> element correctly, few candidates correctly used the method and action attributes of the <FORM> element.

Question 2(d) Very few candidates demonstrated they knew the SQL COUNT() function, or used the GROUP BY clause of a SELECT query.
Question 3(e)(i) Candidates were unable to explain the type of maintenance needed by the change described. Instead, many candidates simply stated the name of a type of maintenance, often incorrectly.

Question 4(a)(ii) This question was not answered well. Candidates who formed a connection with an online database, did not use a SELECT query to read the database details.
Section 3: advice for the preparation of future candidates

Component 1: project
Assessors should ensure that candidates select projects that meet the criteria at Advanced Higher and that they can implement them. Many candidates chose projects that were overly ambitious meaning that they could not complete them to the required standard.

Candidates and assessors should follow the Appendix 1: Instructions and guidance for candidates section of the coursework assessment task to ensure the project meets the requirements of the Advanced Higher course.

Candidates should keep early versions of their project as evidence of development and these should form part of their reflective commentary. Candidates should use reflective commentary to explain how and why they have altered their project. It may be helpful if candidates separate their reflective commentary for each stage of the project.

Assessors should have candidate evidence for every item within each section, to support their marking decisions.

Component 2: question paper

Overall, candidate performance for object-oriented programming improved greatly this year. Similarly, candidate knowledge of the standard algorithms examined in the question paper (binary search and bubble sort showed improvement). Centres should encourage candidates to apply these algorithms (and other standard algorithms covered in the Advanced Higher course) to a variety of data structures.

Responses to the question paper demonstrated that many candidates knew the algorithms, but were unable to adapt them to solve the problems posed. For example:

Question 1(d) Although most candidates coped well with the binary search algorithm, they did not refer to a 2D array structure in their responses.

Question 3(d) When calculating the average house value for a given town, most candidates did not refer to the object-oriented methods getTown() and getValue() in their responses.

Question 4(b) Candidates handled the bubble sort algorithm reasonably well, but many responses did not make any reference to the lastName field of the array of records.

Candidates should be familiar with the content from all aspects of the course. Although the number of questions with no response has fallen, there remains a marked difference in how prepared candidates are to attempt questions from across the course.

In general, candidates who respond well to software development questions (that require knowledge of standard algorithms and data structures), have large gaps in their knowledge when it comes to HTML form processing and SQL queries. The reverse is also true.
Centres should ensure that candidates have experience of integrating their knowledge from different aspects of the course content, including processing data retrieved from an online database within their server-side code. This type of classroom activity will help prepare candidates for problem-solving questions that require integrated solutions.

Candidates’ knowledge of SQL queries could be improved. Centres should ensure that candidates are familiar with SQL SELECT, UPDATE, DELETE and INSERT queries. In addition, candidates should know how these queries are used to process HTML form data. In question 2(c), most candidates did not notice how an UPDATE query was used and the subsequent need to execute the query using server-side script. Similarly, in question 4(b)(ii) most candidates did not notice the need for a SELECT query.

Candidates must pay attention to the wording of each question. On many occasions, candidates provided only partial answers to the question asked. For example:

Question 1(a)  Many candidates named a design methodology rather than a development methodology and several did not describe any benefit of the method named.

Question 1(d)  Although most candidates demonstrated good knowledge of the binary search algorithm, they did not address the full requirements of the search facility to be designed. Many designs did not include the name of a team or display the team’s details.

Question 4(a)(i)  Many candidates defined a suitable record structure but did not define an array with an appropriate dimension.

Question 4(b)  Many candidates did not pay attention to the wording, which specifically referred to a bubble sort algorithm that makes use of a Boolean variable.
Grade boundary and statistical information:

Statistical information: update on courses

<table>
<thead>
<tr>
<th>Number of resulted entries in 2017</th>
<th>641</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of resulted entries in 2018</td>
<td>636</td>
</tr>
</tbody>
</table>

Statistical information: performance of candidates

Distribution of course awards including grade boundaries

<table>
<thead>
<tr>
<th>Distribution of course awards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>Maximum mark</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
<tr>
<td>No award</td>
</tr>
</tbody>
</table>
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.