Resources to support the Advanced Higher Computing Science course

The following table shows which past examination paper questions would be suitable for the Advanced Higher course.

Note: if a question does not match the course, we give advice on what you could change, or list the question as 'not suitable' on the final page.

Software design and development		**Past paper question**	**Advice**
Analysis	Identify the purpose and functional requirements of a problem that relates to the design and implementation at this level in terms of: ♦ inputs ♦ processes ♦ outputs Describe, exemplify, and implement research for: ♦ feasibility studies: — economic	2019 Q4(a) 2017 Q1(a) 2017 Q4(a)	Question can be used without editing. Question can be used without editing. Question can be used without editing (if web marks are required in an assessment, this scenario could easily be changed from software to web).

Software design and development		Past paper question	Advice
	— time — legal — technical ♦ user surveys Describe, exemplify, and implement planning in terms of: ♦ scheduling ♦ resources ♦ Gantt charts Produce requirement specifications for end-users and develop: ♦ end-user requirements ♦ scope, boundaries and constraints ♦ functional requirements Describe, exemplify, and implement Unified Modelling Language (UML): ♦ use case diagrams: — actors — use cases — relationships		

Software design and development		Past paper question	Advice
Design	Identify the data types and structures required for a problem that relates to the implementation at this level. Read and understand designs of solutions to problems at this level using the following design techniques: ♦ structure diagrams ♦ pseudocode ♦ UML	2019 Q1	Additional questions could be written around the UML, for example: ♦ Ask candidates to identify public and private elements. ♦ The UML could be rewritten to show polymorphism. A good example is to add an expenses method to the Event class, which is then over-ridden in different ways by the Work Meeting and Personal sub-classes. ♦ Multiple expenses could be claimed for a single meeting by adding an array of objects (for example, expense and reason) to the design.
	Exemplify and implement efficient design solutions to a problem at this level, using pseudocode, showing: ♦ top level design ♦ the data flow ♦ refinements	2019 Q3(a)(ii) 2018 Q4(c) 2017 Q1(b)	Question can be used without editing (see note in implementation for further editing advice). Question can be used without editing. Question can be used without editing. Additional questions could be added to the scenario, asking candidates to draw a UML using the sample data and details (method to update distances, method to return longest distance for a competitor) in the scenario.
	Describe, exemplify, and implement UML for the following: ♦ class diagrams: — class name — instance variables and data types	2017 Q4(b)	Question can be used without editing.

Software design and development		Past paper question	Advice
	— methods — public and private — inheritance — constructor — array of objects Describe, exemplify, and implement user-interface design using a wireframe, indicating: ♦ visual layout ♦ inputs ♦ validation ♦ underlying processes ♦ outputs		
Implementation	**Data types and structures** Describe, exemplify, and implement the following structures in solutions to problems at this level: ♦ parallel 1-D arrays ♦ records ♦ arrays of records ♦ 2-D arrays ♦ array of objects	2019 Q1 2019 Q3(a)(i) 2019 Q3(a)(ii)	Question can be used without editing. Question can be used without editing. Question could be edited to change it from design to implementation by including SQA reference language, rather than pseudocode. The answer would be written in a programming language of the candidate's choice.

Software design and development		Past paper question	Advice
	Describe and exemplify the operation of linked lists (double and single).	2019 Q3(c)	Question can be used without editing. This question could be changed to a double linked list with a similar question being asked.
	Computational constructs Describe, exemplify, and implement the following object-oriented constructs: ♦ object ♦ property ♦ method ♦ class ♦ sub-class ♦ encapsulation ♦ inheritance ♦ instantiation ♦ polymorphism	2019 Q4(b),(c),(d)	Questions can be used without editing. However, they could be adapted to object-oriented paradigm as follows: ♦ (b)(i) declare an object to store the seat details ♦ (b)(ii) declare a 1-D array of objects to store the bus seats ♦ (c) no change required ♦ (d) no change required
		2018 Q1(c)(ii)	Question could be edited to use a double linked list, allowing the robot to backtrack through the instructions and retrace its steps.
		2018 Q1(d)	Question can be used without editing. However, scenario could be easily adapted to an array of objects instead.
	Describe, exemplify, and implement code to: ♦ open and close connection to database server ♦ execute SQL query ♦ format query results	2018 Q3(a),(b),(c),(d)	Questions can be used without editing. Using the same scenario, additional questions could be added to include revised content, for example: ♦ A facilities property could be added to the House class as an array of strings. Two methods to update and get the facilities could also be added. Polymorphism could be used to

	Algorithm specification Describe, exemplify, and implement standard algorithms including: ♦ binary search ♦ insertion sort ♦ bubble sort Read and explain code that uses constructs appropriate to this level.	 2018 Q3(e)(i) 2018 Q4(a)(i) 2018 Q4(a)(ii) 2017 Q1(c)(i),(ii),(iii)	redefine the GET method to return different parts of the array, depending on whether a house is being sold or rented. ♦ For rented properties only, an array of objects could be added to the UML, storing past details of repairs to the properties. Candidates could be asked to either explain a given addition to the UML or to add the addition onto the current UML. Question can be used without editing. Question can be used without editing. Question could be easily edited to apply to a programming language solution, rather than a scripting language. Questions could be used with an extensive rewrite. The recursive binary search is not valid for the new course, but could be easily replaced using the SQA reference language binary search example from Appendix 9, for example: ♦ (i) and (ii) must be replaced entirely, as recursion and stacks are no longer part of the revised course. ♦ (iii) could be expanded to ask additional question about the supplied SQARL code.

Software design and development		Past paper question	Advice
		2017 Q1(d)	Alternative binary search question could: ♦ supply sample data and ask how many repetitions of the conditional loop would be required to find the target ♦ ask how the code identifies a failure to find the target
		2017 Q1(e)	Question can be used without editing. Question could be edited to refer to a linked list instead. The waiting list could be implemented by adding new nodes to the end of the list (as competitors join the waiting list), and removing the first node from the list each time a competitor withdraws and is replaced by the first person in the waiting list.
		2017 Q3(a)(i),(ii),(iii)	Questions can be used without editing. However, the UML should be edited to remove "thread" and the relationship.
		2017 Q3(b)	Question can be used without editing.
Testing	Describe, exemplify, and implement the following: ♦ integrative testing ♦ usability testing based on prototypes ♦ final testing	2019 Q3(d)	Question could be edited to ask about the types of testing listed in the course specification, as the term "beta" testing is not included in the course specification.

Software design and development		Past paper question	Advice
	♦ end-user testing		
	and: ♦ component testing during the development of the solution	2017 Q3(c)	Question can be used without editing.
Evaluation	Evaluate solution in terms of: ♦ fitness for purpose ♦ maintainability — perfective — corrective — adaptive ♦ robustness	2019 Q4(f) 2018 Q3(e)(i)	Question can be used without editing. Question can be used without editing.
	and: ♦ efficiency ♦ usability		

Database design and development		Past paper question	Advice
Analysis	Identify the purpose and functional requirements of a problem that relates to the design and implementation at this level in terms of: ♦ inputs ♦ processes ♦ outputs Describe, exemplify, and implement research for: ♦ feasibility studies: — economic — time — legal — technical ♦ user surveys Describe, exemplify, and implement planning in terms of: ♦ scheduling ♦ resources ♦ Gantt charts		

Database design and development		Past paper question	Advice
	Produce requirement specifications for end-users and develop: ♦ end-user requirements ♦ scope, boundaries and constraints ♦ functional requirements Describe, exemplify, and implement Unified Modelling Language (UML): ♦ use case diagrams: — actors — use cases — relationships		
Design	Describe, exemplify, and implement entity-relationship diagrams with three or more entries indicating: ♦ entity name ♦ entity type (strong, weak) ♦ attributes ♦ relationship participation (mandatory, optional) ♦ name of relationship ♦ cardinality	2019 Q2(b) 2018 Q2(d)	Question could be used with an additional question asking more about the entity-relationship diagrams (ERD) given in the question, for example: ♦ Ask candidates to redraw the ERD showing participation and strength of the relationship — you could then ask candidates to explain their answer. Question could be used in addition to asking candidates to explain in detail the relationship between the two tables, referring to the strength and participation of the relationship. This could be expanded to also include a Venue entity.

Database design and development		Past paper question	Advice
	Identify relationship participation from an entity-occurrence diagram. Describe, exemplify, and implement surrogate keys. Describe and exemplify a data dictionary, in relation to SQL, with three or more entities for the following: ♦ entity name ♦ attribute name ♦ primary and foreign key ♦ attribute type: — varchar — integer — float — date — time ♦ attribute size ♦ validation: — presence check — restricted choice — field length — range	2017 Q2(a)	Question can be used without editing. However, it could be easily expanded by: ♦ asking candidates to complete the entire data dictionary for the table ♦ supplying a partially completed data dictionary for candidates to complete

Database design and development		Past paper question	Advice
	Exemplify a design of a solution to a query using: ♦ tables and queries ♦ fields ♦ search criteria ♦ sort order ♦ calculations ♦ grouping ♦ having		
Implementation	**SQL** Implement relational database using SQL Data Definition Language (DDL) and Data Manipulation Language (DML) to match the design. Describe, exemplify, and implement the following SQL operations: ♦ CREATE statement: — CREATE DATABASE — CREATE TABLE — constraints: ○ primary key ○ foreign key ○ not null	2019 Q2(b) 2019 Q2(c)	Question can be used without editing. Parts (i) and (ii) contain only Higher level SQL. These could be rewritten to include Advanced Higher concepts: ♦ Part (i) could require that a sub-query is used to count and display the number of tours conducted by each guide on 16/04/2019, if a tour had five or more participants. ♦ Part (ii) could be rewritten to display the largest number of tours completed in a single day between two dates. Many other examples of questions using NOT and IN could be easily constructed for this two-table database.

Database design and development		Past paper question	Advice
	○ check ○ auto increment ♦ DROP statement: — DROP DATABASE — DROP TABLE ♦ HAVING clause of the SELECT statement ♦ subqueries used with the WHERE clause of SELECT statements ♦ data types: — varchar — integer — float — date — time ♦ logical operators: — IN — NOT — BETWEEN — ANY — EXISTS Read and explain code that uses the SQL at this level.	2018 Q2(a) 2018 Q2(d) 2017 Q2(b)(i),(ii),(iii) 2017 Q2(c)	Question can be used without editing. Table and scenario could be re-used to write an Advanced Higher SQL question. The current question contains only Higher content. You could: ♦ count all the reviews by each user that were NOT written in the last calendar year ♦ use a sub-query to display all the users who have more than an average number of reviews Questions can be used without editing. Question could be used but requires editing, as the SQL query described is National 5 level. A few simple changes could be to: ♦ ask for reservations between two dates (or two times) ♦ create a list that excludes one or more of the restaurants ♦ display a list of reservations for each day, where a day is only displayed if a group of six people has reserved a table for that day

Database design and development		Past paper question	Advice
Testing	Describe, exemplify, and implement the following: ♦ integrative testing ♦ usability testing based on prototypes ♦ final testing ♦ end-user testing		
	and: ♦ SQL implemented tables match design ♦ SQL operations work correctly at this level		
Evaluation	Evaluate solution in terms of: ♦ fitness for purpose ♦ maintainability — perfective — corrective — adaptive ♦ robustness		
	and: ♦ accuracy of output		

Web design and development		Past paper question	Advice
Analysis	Identify the purpose and functional requirements of a problem that relates to the design and implementation at this level in terms of: ♦ inputs ♦ processes ♦ outputs Describe, exemplify, and implement research for: ♦ feasibility studies: — economic — time — legal — technical ♦ user surveys Describe, exemplify, and implement planning in terms of: ♦ scheduling ♦ resources ♦ Gantt charts		

Web design and development		Past paper question	Advice
	Produce requirement specifications for end-users and develop: ♦ end-user requirements ♦ scope, boundaries and constraints ♦ functional requirements Describe, exemplify, and implement Unified Modelling Language (UML): ♦ use case diagrams: — actors — use cases — relationships		
Design	Describe, exemplify, and implement wireframe designs showing: ♦ visual layout ♦ navigation ♦ consistency ♦ underlying processes Describe, exemplify, and implement low-fidelity prototype from wireframe design.		

Web design and development		Past paper question	Advice
	Read and understand designs of server-side processes at this level, using the following design techniques: ♦ structure diagrams ♦ pseudocode Exemplify and implement the design of server-side processes using pseudocode.		
Implementation	**CSS** Describe, exemplify, and implement responsive pages using the following media queries: ♦ media type: — print — screen ♦ media feature: — max-width **HTML** Describe, exemplify, and implement form elements including: ♦ FORM element: — action	2019 Q2(a) 2019 Q2(d) 2018 Q2(b) 2018 Q2(c) 2018 Q4(a)(ii)	Question can be used without editing. Question could be used, but should be edited to state “PHP” rather than the generic “server-side script”. Question can be used without editing. However, the marking instructions should be edited to focus on action, method and name. If this was done, you could supply the Higher content of the form in the question, as no marks would be awarded for this. Question can be used without editing. However, it would be more accurate to refer to “PHP” in the question rather than the generic “server-side scripting language”. Question can be used without editing. Further questions could be included to query this table.

Web design and development		Past paper question	Advice
	— method (get and post) ♦ INPUT, SELECT and TEXTAREA elements: — name — value ♦ TABLE element: — th, tr, td **PHP** Describe, exemplify, and implement coding of server-side processing to: ♦ assign form data to server-side variables: — $_get() — $_post() ♦ open and close connection to database server: — die() — mysqli_connect() — mysqli_close() ♦ execute SQL query: — mysqli_query() ♦ format query results: — echo — mysqli_fetch_array()		Additional questions could: ♦ ask for the PHP code required to execute a simple query and to format the results in an HTML table ♦ focus on Advanced Higher SQL queries

Web design and development		Past paper question	Advice
	— mysqli_num_row() and: ♦ assignment, repetition and selection using server-side local and global variables ♦ sessions: — session_start() — session_destroy() Read and explain code that uses constructs appropriate to this level.		
Testing	Describe, exemplify, and implement the following: ♦ integrative testing ♦ usability testing based on prototypes ♦ final testing ♦ end-user testing		
Evaluation	Evaluate solution in terms of: ♦ fitness for purpose ♦ maintainability — perfective		

Web design and development		Past paper question	Advice
	— corrective — adaptive ♦ robustness		
	and: ♦ usability		

Computer systems		Past paper question	Advice
Data representation	Describe and exemplify the use of hexadecimal to represent positive integers. Convert hexadecimal numbers to binary/denary and vice-versa. Add two 8-bit two's complement numbers leading to possible overflow errors.		
Computer structure	With reference to the fetch-execute cycle, describe: ♦ the role of the: — memory address register (MAR) — memory data register (MDR) — instruction register (IR) ♦ the use of pipelining to increase throughput Describe the role of the flag registers (overflow, carry, sign, and zero) following Arithmetic Logic Unit (ALU) operations.		
Environmental impact	Describe the environmental impact of data centres.	2017 Q1(f)	Question does not specifically mention data centres, but fits well with the revised course.
Security risks and precautions	Describe and identify the security risks of SQL code injections and how to protect against them.		

Past paper questions not suitable for the Advanced Higher Computing Science course

The following past paper questions are not suitable for the Advanced Higher Computing Science course, as we have removed the topic/term from the course:

2019

- 2(e) — legal concerns
- 3(a)(iii) — recursion
- 3(b) — queues (and stacks)
- 4(e) — multi-threading

2018

- 1(a) — iterative prototyping (although candidates may experience this while they work on their projects)
- 1(b) — queues
- 1(c) — stacks
- 1(e) — social and ethical implications
- 2(e) — legal and economic implications

2017

- 1(c)(i) — recursion
- 1(c)(ii) — stacks
- 1(e) — queues (this question could be easily edited, as described earlier in the SDD section)
- 2(d) — quicksort algorithm
- 3(d) — implications of big data