National Qualifications 2024

2024 Computing Science

Advanced Higher

Question Paper Finalised Marking Instructions

These marking instructions have been prepared by examination teams for use by SQA appointed markers when marking external course assessments.

General marking principles for Advanced Higher Computing Science

This information is provided to help you understand the general principles you must apply when marking candidate responses to questions in this paper. These principles must be read in conjunction with the detailed marking instructions, which identify the key features required in candidate responses.

(a) Marks for each candidate response must **always** be assigned in line with these general marking principles and the detailed marking instructions for this assessment.

(b) Always use positive marking. This means candidates accumulate marks for the demonstration of relevant skills, knowledge and understanding; marks are not deducted.

(c) If a candidate response is not covered by either the principles or detailed marking instructions, and you are uncertain how to assess it, you must seek guidance from your team leader.

(d) Award marks regardless of spelling, as long as the meaning is unambiguous. This applies to all responses, including code. Award marks as per the detailed marking instructions, regardless of syntax errors, if the intention of the coding is clear.

(e) For questions where candidates are asked to design or write code, a sample response is shown in the detailed marking instructions. This will not be the only valid response. You must use the detailed marking instructions and additional guidance to ensure that you consider alternative approaches and nuances of different programming languages. If in doubt you should refer to your team leader.

(f) If a candidate scores through a response and makes a further attempt, you should only mark the further attempt. If no further attempt is made and the original is legible, you should mark the original response.

(g) Where an incorrect response is carried forward and used correctly in a following part of the question, you should give credit for subsequent responses that are correct with regard to the original error. Candidates should not be penalised more than once for the same error.

(h) Only award marks for a valid response to the question asked. Where candidates are asked to:

- **Identify**, **name**, **give** or **state,** they need only name or present in brief form.
- **describe**, they must provide a statement or structure of characteristics and/or features. This will be more than an outline or a list. It may refer to, for example, a concept, process, experiment, situation, or facts, in the context of and appropriate to the question. Candidates must make the same number of factual/appropriate points as there are marks available in the question.
- **explain**, they must relate cause and/or effect and/or make relationships between things clear, in the context of the question or a specific area within the question.
- **write code,** they must write recognisable code, not prose nor a diagram.
- **design,** they must use a design technique appropriate to the problem. Award marks as per the detailed marking instructions, regardless of errors in the exemplification of the technique, if the intention of the design is clear.

(i) In the detailed marking instructions, if a word is underlined then it is essential; if a word is in brackets() then it is not essential. Words separated by / are alternatives.

Marking instructions for each question

Section 1 – Software design and development

<table>
<tr><th colspan="3">Question</th><th>Expected response</th><th>Max mark</th><th>Additional guidance</th></tr>
<tr><td>1.</td><td></td><td></td><td>Compare with Oak.
Compare with Willow.
Compare with Sycamore.</td><td>1</td><td>1 mark for all comparisons correct.</td></tr>
<tr><td>2.</td><td>(a)</td><td></td><td>519</td><td>1</td><td></td></tr>
<tr><td></td><td>(b)</td><td></td><td><table><tr><td>Node</td><td>Pointer</td></tr><tr><td>523</td><td>514</td></tr><tr><td>517</td><td>523</td></tr><tr><td>519</td><td>517</td></tr><tr><td>514</td><td>null</td></tr></table></td><td>2</td><td>1 mark for pointers 523, 517 and 519 correct.

1 mark for pointer 514 correct.</td></tr>
<tr><td>3.</td><td>(a)</td><td></td><td><pre>DECLARE grid AS ARRAY OF ARRAY OF INTEGER [ ]
<9 x 9 array, all set to zero></pre>OR<pre>DECLARE grid TO [[]] * 9
FOR count FROM 0 TO 8 DO
SET grid [count] TO [0] * 4
END FOR</pre></td><td>1</td><td>1 mark for definition of 2D array of integer values with dimensions 9 × 9.

Also accept solutions in other programming languages
eg <code>DIM grid (8,8) AS INTEGER</code></td></tr>
<tr><td></td><td>(b)</td><td></td><td><pre>SET grid[3,4] TO 9</pre></td><td>1</td><td>Also accept:<pre>SET grid[4,3] TO 9
SET grid[4,5] TO 9
SET grid[5,4] TO 9</pre></td></tr>
<tr><td>4.</td><td>(a)</td><td></td><td>This is the description of a persona and test case that will be used during final testing.

The characteristics of the persona will be adopted by developers who perform the test case to find out how easy it was to use or on any problems encountered.</td><td>2</td><td>1 mark for description of persona and test case.

1 mark for description of how the developers would use the persona and test case during final testing of the system.</td></tr>
<tr><td></td><td>(b)</td><td></td><td>Perfective.

This maintenance is adding additional features to a project that was not part of the initial requirements.</td><td>1</td><td>1 mark for perfective maintenance with appropriate explanation.</td></tr>
</table>

Question			Expected response	Max mark	Additional guidance
5.	(a)		Overriding is used to define the `calcArea()` method for each of the 3 subclasses of the `Shape` superclass (`Triangle`, `Rectangle` and `Circle`) and replace the code within the method inherited from the superclass. In this way, each of the subclasses can respond to the `calcArea()` method in a way that is unique to that subclass.	**2**	1 mark for use of overriding to replace inherited method. 1 mark for use of overriding to create unique responses for each subclass.
	(b)	(i)	This statement creates an instance of the `Rectangle` class called `r1`. The constructor of the `Rectangle` class then assigns the following values to each of the instance variables of the `r1` object: fillColour = white lineColour = black base = 20 height = 16 area = 0	**2**	1 mark for instantiation of the object `r1` of the `Rectangle` class 1 mark for correct assignment of all instance variables.
		(ii)	The area is 0 The area is 320	**2**	1 mark for area = 0 1 mark for area = 320
	(c)	(i)	The fill colour of the roof is set to red.	**1**	Accepts also: The fill colour of object t1 is set to red.
		(ii)	The object list is an array of objects of type `Shape`. Due to polymorphism, list can contain objects of type `Shape` or of any of its subclass types. Although the `list[8]` references the variable `c2` which is an instance of the `Circle` class, the variable list has been defined as a `Shape` object and acts as `Shape` type container for each object it references. As a result, the only methods that can be accessed by elements of the array are those that are known to the `Shape` superclass. Since the `getCircumference()` method is only associated with the `Circle` subclass, an error will be generated when any of the array elements attempt to invoke it.	**2**	1 mark each for a maximum of 2 marks: • list is a polymorphic array of objects of type `Shape` • `list[8]` references an object belonging to the `Circle` class • the variable list acts as a `Shape` type container for the objects it references • the `getCircumference()` method is not accessible to `Shape` class objects

Question			Expected response	Max mark	Additional guidance
5.	(d)	(i)	`WHILE swapped AND n>=0 DO`	**1**	Also accept: `WHILE swapped = TRUE AND n>=0 DO`
		(ii)	`IF array[i].getArea() < array[i+1].getArea() THEN`	**2**	1 mark for correct use of `getArea()` method 1 mark for correct comparison. Also accept the following: • `IF array[i+1] > array[i} THEN`

		(iii)	`SET temp TO array[i]`
			`SET array[i] TO array[i+1]`
			`SET array [i+1] TO temp`
				1	Also accept:
					`SET temp TO array[i+1]`
					`SET array[i+1] TO array[i]`
					`SET array [i] TO temp`

6	(a)		2.1 open connection to database server 2.2 execute SQL query to retrieve all records from database table MammalData 2.3 loop for number of rows of data 2.4 assign individual data item to an individual field 2.5 end loop 2.6 close connection to database server	**3**	1 mark for open/close connection to database server. 1 mark to import the records using SQL query. 1 mark for writing data to the array of records within a loop. Notes: - do not accept file instead of database or database server - do not penalise for loop of 1000
	(b)	(i)	See sample solution below. Alternative solutions possible.	**3**	1 mark for correct complex condition for inner loop at Line 3.3.7 of the sample solution. Also accept: list(index - 1).recordNumber <= value.recordNumber and index > 0 1 mark for using inner loop to work backwards through the array (Line 3.3.9 of the sample solution). 1 mark for updating contents of array at Line 3.3.11 of the sample solution.

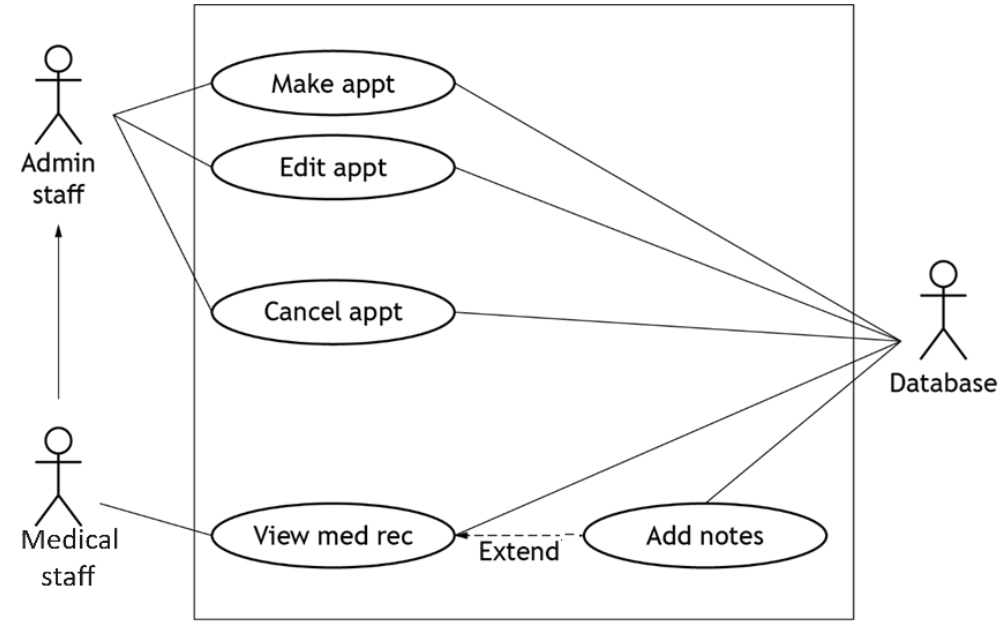
```
3.3.4       loop from n = 1 to length(list) - 1
3.3.5           set value to array(n)
3.3.6           set index to n
3.3.7           loop while index > 0 and value.readingNumber > list(index - 1)
                .readingNumber
3.3.8               set list(index) = list(index - 1)
3.3.9               set index to index - 1
3.3.10          end loop
3.3.11          set list(index) = value
3.3.12      end loop
```

Question			Expected response		Max mark	Additional guidance
6.	(b)	(ii)	See sample solution below.		**4**	1 mark each for any four of the following: • counting rows of new data in temp array • counting number of animals currently in animalList array • backwards loop used to move data in animalList array • assignment of data to correct position in animalList array • creating empty records at the start of the animalList array Note: candidates may used fixed loop of 1000 to traverse animalList array.
			3.4.1	procedure moveData (animalList()) set numberUpdateRecords = 0 set newPosition = 0 loop while field species of temporary array is not empty add 1 to numberUpdateRecords end loop loop while field species of animalList array is not empty add 1 to newPosition end loop loop backwards from (newPosition – 1) to 0 set animalList(counter + numberUpdateRecords) = animalList(counter) end loop loop from 0 to numberUpdateRecords – 1 set animalList(loop) to NULL end loop end procedure		
			3.4.2			
			3.4.3			
			3.4.4			
			3.4.5			
			3.4.6			
			3.4.7			
			3.4.8			
			3.4.9			
			3.4.10			
			3.4.11			
			3.4.12			
			3.4.13			
			3.4.14			
			3.4.15			
			3.4.16			

Question			Expected response		Max mark	Additional guidance
6.	(b)	(iii)	See sample solution below. Alternative solutions possible.		**3**	1 mark for matching individual update record with previous record. 1 mark for calculating percentage difference and comparing with 5% of previous value. 1 mark for displaying details required.

			3.5.1
			
			3.5.2
			
			3.5.3
			
			3.5.4
			
			3.5.5
			
			3.5.6
			3.5.7
			3.5.8
			3.5.9
			3.5.10
			
			3.5.11
				start loop1 from 0 to (numberUpdateRecords - 1)
				
				start loop2 from numberUpdateRecords to (number of records in animalList array – 1)
				if animalList(loop2).mammal = animalList(loop1).mammal and animalList (loop2).readingNumber = animalList(loop1).readingNumber - 1 then
				set difference = animalList(loop2).numberRemaining – animalList (loop1). numberRemaining
				set percentage = difference / animalList(loop2).numberRemaining * 100
				if percentage >= 5 then
				display animalList(loop1).mammal and percentage
				end if
				end if
				end loop
				
				end loop		

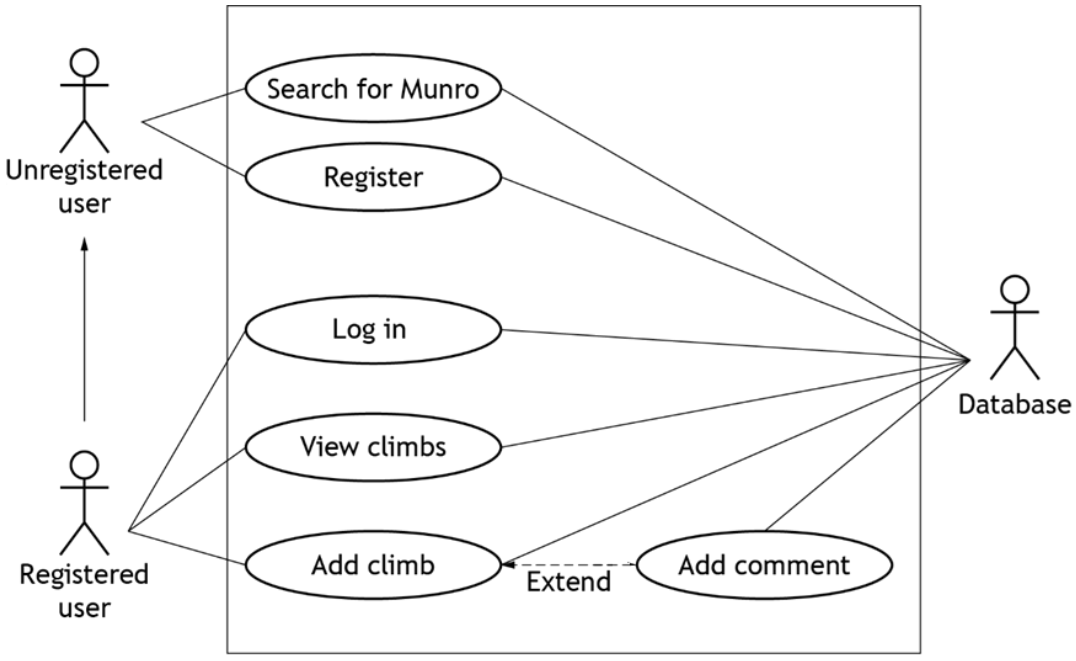
Section 2 - Database design and development

Question			Expected response	Max mark	Additional guidance
7.	(a)		Actor is a strong entity. Role is a weak entity.	**1**	1 mark for both correct.
	(b)		An actor may or may not have a role in one of the movies. This means that the entity Role has optional participation in the relationship. A role must be assigned to one specific actor. This means that the Actor entity has mandatory participation in the relationship.	**2**	1 mark for accurate description of relationship participation. 1 mark for correct use of optional and mandatory terminology.
	(c)		`genre varchar(6) CHECK (genre IN ('Horror', 'Comedy', 'Action'))`	**2**	1 mark for correct use `varchar(6)` with `CHECK` 1 mark for correct use of `IN` Notes: • field size must be between 6 and 20 characters • Do not accept `OR` instead of `IN`
8.	(a)		Technical	**1**	
	(b)		See sample solution below.	**3**	1 mark for 3 actors with generalisation (DB, Admin Staff, Medical Staff). 1 mark for correct use cases (Make appt, Edit appt, Cancel appt and View Patient Medical Record) and appropriate relationships. 1 mark for extend for Add Notes.
					

Question			Expected response	Max mark	Additional guidance
8.	(c)		Search would be improved, for example, reduces the amount of data entered to retrieve details for a particular appointment. Automatically incremented and added at the time the record is created meaning no chance of duplication. It is simpler to create join conditions on complex queries. Surrogate keys are immutable; without a surrogate, changing an appointment would mean changing part of the primary key. Easier to reference historical data.	**2**	1 mark for each of two benefits of a surrogate key relating to scenario. Note: accept 'reduced search time' but not 'faster'.
	(d)		`EXISTS (SELECT patientID FROM Patient WHERE patientID = "GS0901");`	**2**	1 mark for correct use of `EXISTS` 1 mark for correct subquery.
	(e)	(i)	Use of count in the search criteria is incorrect; count should be in the HAVING section of the design. The design cannot use age as part of the search criteria. NOT child or IN(adult, senior) should be used instead.	**2**	1 mark for explaining incorrect use of count with correct solution. 1 mark for explaining incorrect use of age with a possible solution.
		(ii)	`dateApp BETWEEN "2024/05/27" AND "2024/05/31"`	**1**	1 mark for correct use of `BETWEEN` logical operator. Note: do not penalise for date format.
	(f)	(i)	dateApp 09/06/2024 12/06/2024 12/06/2024 14/06/2024	**1**	1 mark for 4 correct results. Note: do not penalise for lack of column headings.
		(ii)	dateApp 09/06/2024 12/06/2024	**1**	1 mark for 2 correct results. Note: do not penalise for lack of column headings.
	(g)		`<form action="search.php" method="post">`	**2**	1 mark for correct `action` 1 mark for correct `method`

[END OF SECTION 2]

Section 3 - Web design and development

Question			Expected response	Max mark	Additional guidance
9.			This code is triggered when the width of the screen is less than or equal to 600 pixels to apply styles suitable for a smaller screen.	**1**	1 mark for description of the media query with reference to screen size.
10.	(a)		`<tr><th>Price per month</th><th>Basic</th><th>Standard</th><th>Premium</th></tr>`	**1**	1 mark for correct use of `<th>`, `<tr> </th>` and `</tr>` tags.
	(b)		`<tr><td>Gb Storage</td><td>10</td><td>20</td><td>30</td></tr>`	**1**	1 mark for correct use of `<tr>`, `<td>`, `</tr>` and `</td>` tags.
11.	(a)		Technical	**1**	
	(b)		See sample solution below.	**3**	1 mark for 3 actors with generalisation (DB, unregistered user and registered user). 1 mark for correct use cases (Search for Munro, Register, Login, View climbs, Add climb) together with appropriate relationships. 1 mark for extend for Add comment.
					

Question			Expected response	Max mark	Additional guidance
11.	(c)	(i)	Line 24 number of results returned by the query. Line 26 Display error message saying that email address already registered.	**2**	1 mark each. Note: accept mysqli_num_rows() for line 24.
		(ii)	See sample solution below.	**2**	1 mark for correct use of $_POST function 1 mark for correct use of all name attributes used with HTML form elements. Note: do not penalise for use of alternative PHP variable names.

			`$forename = $_POST['first'];`
			`$surname = $_POST['last'];`
			`$email = $_POST['email'];`
			`$password = $_POST['pass'];`
			`$ability = $_POST['level'];`		

		(iii)	See sample solution below.	**2**	1 mark for `INSERT INTO Climber` 1 mark for correct list of fields and values.. Note: field list must be correct – climberID should not be present; also, values must be in same sequence as field list.

			`INSERT INTO Climber (forename, surname, email, password, climbingAbility) VALUES ('$forename, '$surname, '$email', '$password', '$ability')`
			
			Also accept:
			`INSERT INTO Climber (forename, surname, email, password, climbingAbility) VALUES ("emma", "hinze", "ehinze@freemail.de", "gewinner6@$gold", "intermediate")`		

Question			Expected response	Max mark	Additional guidance
11.	(d)	(i)	Any username entered on the Log in or Register page would be POSTed to the Welcome page but would not be available to any other pages of the website. A session variable is needed to store the POSTed value so that it can persist across all pages of the website.	**1**	1 mark for use of session variable to allow entered email to persist on additional pages of the website.
		(ii)	Since the email of the logged in user would be stored in the session variable, the session variable would be used as the search criteria for the SQL query needed to retrieve details of all climbs completed by the user.	**1**	1 mark for use of username as search criteria in the SQL query.
	(e)	(i)	Username = nevis	**1**	
		(ii)	Line 63 The `$connection` parameter has been omitted; the correct code should be: `mysqli_query($connection, $sql)` Line 64 The parameter `$result` has been omitted; the correct code should be `mysqli_fetch_array($result)`	**2**	1 mark for missing `$connection` parameter at Line 63 1 mark for missing `$result` parameter at Line 64
	(f)	(i)	The script will perform a find maximum algorithm to display the forename and surname of the climber with the highest number of climbs in the database.	**1**	1 mark for find maximum algorithm. 1 mark for displaying details of climber with the highest number of climbs.
		(ii)	At Line 4, the climberID is set to the index of the loop and at Line 5, howMany is set to the number of climbs completed by this climber. However, if the record with this climberID does not exist in the database (for example, the record may have been deleted at some point), then an error may be generated at Line 5.	**1**	1 mark for explanation.

[END OF MARKING INSTRUCTIONS]