National
Qualifications
2025

2025 Computing Science

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.

SQA ©

General marking principles for 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 puts a score 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 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, AND COMPUTER SYSTEMS

Question			Expected response	Max mark	Additional guidance
1.	(a)		511	**1**	Accept 2^9 -1
	(b)		10 (bits)	**1**	
2.			• authenticates sender • confirms that the document has not been altered	**2**	
3.			• initialising and increment both counters • loop for length of string/each letter • two if statements or if/else if to check each single character • correct conditions for lower case and uppercase	**4**	Set lower to 0 Set upper to 0 For each character in sentence If current character>='a' and <='z' Set lower to lower +1 Else if current character>='A' and <='Z' Set upper to upper +1 End if End loop Candidate can also use ASCII codes for upper (65-90) and lowercase (97-122) Award fourth bullet if candidate has used a correct condition and an Else statement.
4.			• agile would have regular feedback (identifying changes) • prototypes would provide vehicle for discussion • requirements gathered for each sprint	**1**	Award 1 mark for any one bullet.
5.	(a)		(Control bus) activates the read line	**1**	
	(b)	(i)	• increasing cores • increasing clock speed • increasing or adding cache • increasing width of data bus	**1**	Award 1 mark for any one bullet.
		(ii)	• allows multiple instructions simultaneously • more instructions executed per second • faster access time compared to slower main memory/increased number of cache hits • more bits transferred in a single operation/less fetch executes required	**1**	Award 1 mark for bullet that matches method stated in part (i). If mark for (i) not awarded, candidate can be awarded for suitable description.

Question			Expected response	Max mark	Additional guidance
6.	(a)		Graphic B as it has not pixelated/is resolution independent.	**1**	
	(b)		Individual pixels can be edited.	**1**	
7.	(a)		Step 2: IN: month(), year() OUT: Step 3: IN: year() OUT: Step 4: IN: numStormsYear() OUT:	**3**	Award 1 mark for each step. For each step the mark is awarded for the correct IN with no additional parameters (with exception of year() in Step 4). Brackets must be indicated.
	(b)		• initialise and increment of count • input from user, loop and output • condition for targetYear • condition for autumn (September, October, November) with OR including brackets	**4**	Candidates may use nested if for fourth bullet which would not require brackets.

	(c)		• declaration and use of flag to determine presence
			• loop for correct range and both output messages
			• input of name and if condition for given name
			
			Example answer:
			
			`DECLARE stormName INITIALLY FROM KEYBOARD`
			`DECLARE found INITIALLY False`
			`FOR index = 0 to length(name) - 1`
			`  IF name[index] = stormName THEN`
			`  SET found TO True`
			`  SEND "Name already used" TO DISPLAY`
			`  END IF`
			`END FOR`
			`IF found = False THEN`
			`  SEND "Name not used" TO DISPLAY`
			`END IF`
				3	"Name not used" output must be outside of loop.

Question			Expected response	Max mark	Additional guidance
8.	(a)	(i)	• record structure with candidate selected name • four fields specified Example answer: `RECORD player IS (STRING name, STRING email, STRING attemptID, REAL time)`	**2**	If data types are indicated they must be appropriate.

		(ii)	• array of 10 000 elements
			• use of record structure from (i)
			
			Example answer:
			
			`DECLARE allPlayers AS ARRAY OF player INITIALLY[]*10000`
				2	Award 1 mark for
					`allPlayers =[player()]*10000`
					`allPlayers =[player]*10000`

	(b)	(i)	• initialise fastest to first time or suitable high value and re-assigning fastest
			• loop using `numPlays`
			• if condition to update fastest
			• use of array variable from (ii)
			• use of fields from (ii)
			
			Example answer:
			
			`SET fastest TO allPlayers[0].time`
			`FOR i FROM 1 TO numPlays - 1 DO`
			`  IF allPlayers[i].time < fastest:`
			`  SET fastest TO allPlayers[i].time`
			`  END IF`
			`END FOR`
				5	

		(ii)	Traversing array elements which will not have been populated/0/NULL.	**1**	
	(c)		• sign bit: 0 • mantissa: 100 0000 0000 0000 • exponent: 1111 1110	**3**	

	(d)	(i)	• assigning to position and calling function
			• correct parameters in correct order
				2	`position=findCharIndex(email,'@')`
					
					Accept
					
					`position= findCharIndex(arrayname[index].email,'@')`

		(ii)	• extracting substring • assignment to `attemptID` using concatenation with `numPlays`	**2**	Candidate may use variable or reference an element of their record

<table>
<tr><th colspan="3">Question</th><th>Expected response</th><th>Max mark</th><th>Additional guidance</th></tr>
<tr><td>9.</td><td>(a)</td><td>(i)</td><td>Area of code in which a variable is usable/accessible.</td><td>1</td><td></td></tr>
<tr><td></td><td></td><td>(ii)</td><td>• prevents errors with variables with the same name in different modules
• more memory efficient as variables will not be in memory when not in use</td><td>1</td><td>Award 1 mark for any one bullet.</td></tr>
<tr><td></td><td>(b)</td><td></td><td>• calls the function <code>removal</code>
• assigns the result to the variable <code>newNums</code>
• passed in the (actual) parameters</td><td>2</td><td>Award 1 mark for each bullet. Maximum 2 marks.

Accept expression of line 4 as removal function.</td></tr>
<tr><td></td><td>(c)</td><td></td><td>• both position updates
• update of index
• array update to [42,12]

<table>
<tr><th>Line Number</th><th>newList</th><th>position</th><th>index</th></tr>
<tr><td>5</td><td>[0,0]</td><td></td><td></td></tr>
<tr><td>6</td><td></td><td>0</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td>0</td></tr>
<tr><td>9</td><td>[4,0]</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td>1</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td>1</td></tr>
<tr><td>9</td><td>[42,0]</td><td></td><td></td></tr>
<tr><td>10</td><td></td><td>2</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td>2</td></tr>
</table></td><td>3</td><td></td></tr>
<tr><td></td><td>(d)</td><td></td><td>• allows execution of the program to be halted
• allows the contents of the variables to be compared</td><td>2</td><td></td></tr>
<tr><td></td><td>(e)</td><td>(i)</td><td>• array to move values into is set up as one less than the original array
• so cannot hold all values if no value was removed</td><td>2</td><td></td></tr>
<tr><td></td><td></td><td>(ii)</td><td>• (add code to) check if the value is present

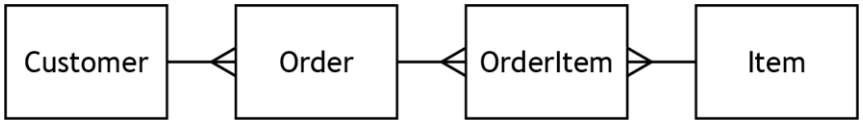
• (add code to) run original code/return original array instead of running code

OR

• (add code to) declare array of the same size as the target array (that is passed in)</td><td>2</td><td>Candidates can count occurrences of the value to be removed for bullet 1 and then may reference adding code to reduce arrays by multiple elements.</td></tr>
</table>

Question			Expected response	Max mark	Additional guidance
9.	(f)		• (it is not robust) it will crash when trying to reference an array element that doesn't exist • (it is not robust) program will crash if the array passed in is empty	**1**	Award 1 mark for any one bullet.

Section 2 - DATABASE 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>10.</td><td></td><td></td><td>• grouping on year and genre to return three rows
• corresponding values for shortest time</td><td>2</td><td><table><tr><th>year</th><th>genre</th><th>shortest time</th></tr><tr><td>2023
2023
2024</td><td>Romance
Sci-Fi
Drama</td><td>40
90
45</td></tr></table>
Results can be in any order.</td></tr>
<tr><td>11.</td><td></td><td></td><td>• customer – Order (1:M)
• order – OrderItem (1:M)
• item – OrderItem (1:M)
  </td><td>3</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Expected response</th><th>Max mark</th><th>Additional guidance</th></tr>
<tr><td>12.</td><td>(a)</td><td></td><td>• COUNT(any field or *)
• Field names, tables and criteria
• GROUP BY fName,sName/staffID</td><td>3</td><td>
<table>
<tr><td>Field(s) and Calculation(s)</td><td>fName,sName, COUNT (*)</td></tr>
<tr><td>Tables(s)</td><td>Photographer, Appointment</td></tr>
<tr><td>Search Criteria</td><td>appDate like %/07/2025</td></tr>
<tr><td>Grouping</td><td>fName,sName</td></tr>
<tr><td>Sort Order</td><td></td></tr>
</table>
Grouping can also be by staffID.

Accept * or ?? or_ _ for wildcard operator.

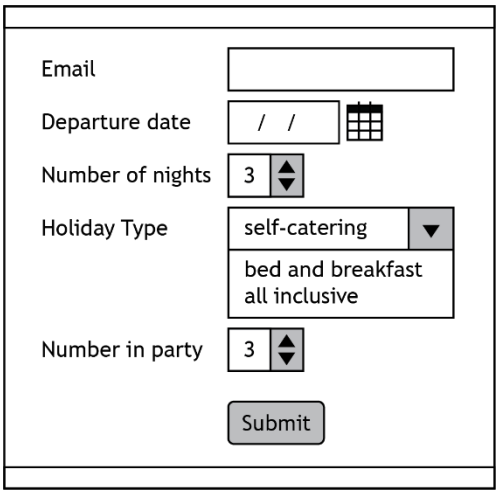
Date criteria can be specified through wildcards or range (1/7/25-31/7/25).</td></tr>
<tr><td></td><td>(b)</td><td></td><td>• UPDATE Invoice
• SET cost = cost + 20
• WHERE Status = "Unpaid" and invoiceDate < 31/03/2025</td><td>3</td><td>Award 0 marks for first bullet if SQL clauses are in wrong order.

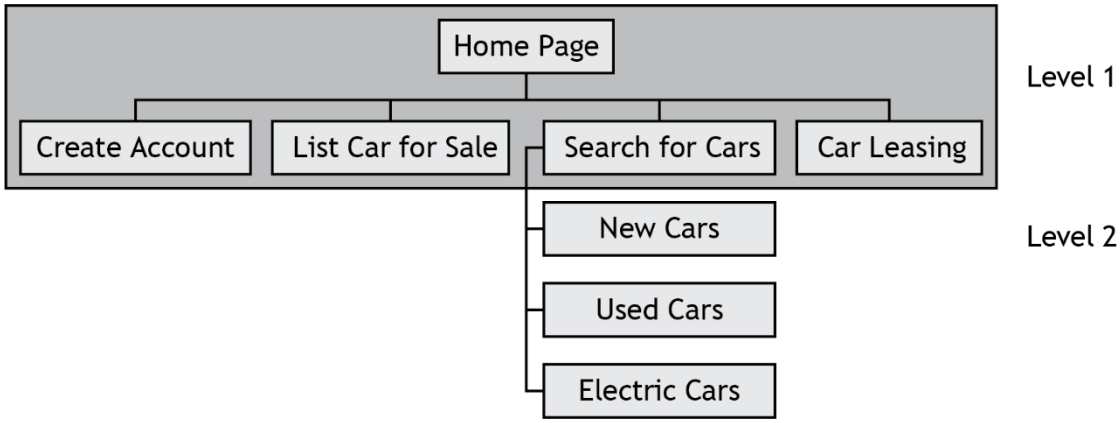
UPDATE Invoice
SET cost = cost + 20
WHERE Status = "Unpaid" and invoiceDate < '31/03/2025'</td></tr>
<tr><td></td><td>(c)</td><td></td><td>• missing Appointment table
• missing join between Appointment and Photographer table
• missing GROUP BY fName, sName (or staffID) is needed</td><td>3</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Expected response</th><th>Max mark</th><th>Additional guidance</th></tr>
<tr><td>13.</td><td>(a)</td><td></td><td>• correct fields and name = ‘L Fletcher’
• all four tables</td><td>2</td><td><table><tr><td>Field(s) and Calculation(s)</td><td>name,regNo, partName</td></tr><tr><td>Table(s)</td><td>Customer, Vehicle, Appointment, Part</td></tr><tr><td>Search Criteria</td><td>name = ‘L Fletcher’ (custID = 0039)</td></tr><tr><td>Grouping</td><td></td></tr><tr><td>Sort Order</td><td></td></tr></table></td></tr>
<tr><td></td><td>(b)</td><td></td><td>• SELECT with ALIAS and Part table
• SUM (quantity * price)</td><td>2</td><td><pre>SELECT SUM (quantity * price) AS [Total value]
FROM Part</pre>Disregard WHERE quantity >0</td></tr>
<tr><td></td><td>(c)</td><td>(i)</td><td>• SELECT AVG(cost) and Alias
• FROM Appointment</td><td>2</td><td><pre>SELECT AVG(cost) as [Average Cost]
FROM Appointment</pre></td></tr>
<tr><td></td><td></td><td>(ii)</td><td>• FROM includes Vehicle, Appointment with Vehicle.regNo = Appointment.regNo in WHERE
• FROM clause includes ‘avgCost’
• WHERE cost > [Average Cost]
• ORDER BY cost DESC;</td><td>4</td><td><pre>SELECT regNo, appID, make, model, cost
FROM Vehicle, Appointment, avgCost
WHERE Vehicle.regNo = Appointment.regNo AND cost > [Average Cost]
ORDER BY cost DESC;</pre>Award second bullet if candidate has used subquery.

<pre>SELECT regNo, appID, make, model, cost
FROM Vehicle, Appointment
WHERE Vehicle.regNo = Appointment.regNo AND cost > (SELECT AVG(cost)
FROM Appointment)
ORDER BY cost DESC;</pre></td></tr>
<tr><td></td><td>(d)</td><td></td><td>Suitable cardinality between Parts:PartListing and Appointment:PartsListing</td><td>1</td><td>Part/Appointment entities may be shown opposite from below.</td></tr>
<tr><td></td><td></td><td></td><td colspan="3"> 
</td></tr>
</table>

Section 3 - WEB DESIGN AND DEVELOPMENT

Question			Expected response	Max mark	Additional guidance
14.	(a)		• a tester acting as a typical end user of the web site	**1**	
	(b)		• no label for the comments form input • comments would be longer than the text box • fields can be left blank	**2**	Award 1 mark for each bullet. Maximum 2 marks. For bullet 2 candidate may express using a `textarea` as a solution.
15.			• the hover should apply to anchor element	**1**	Candidate may rewrite code `nav ul li a: hover`
16.			• padding: 15(px)	**1**	Can also be padding 15 15 15 15
17.	(a)		• labelled fields for email, date, number of nights, type, number of people • validation of number of nights and people in party. Email being required • drop down or option buttons for holiday type • submit button	**4**	 Accept all fields being required.
	(b)	(i)	Descendant	**1**	
		(ii)	Targets `p` elements that are contained in a `section` element	**1**	Must refer to `<p>` and `<section>`
	(c)	(i)	`block`	**1**	
		(ii)	• onclick event with function • correct parameter supplied to function	**2**	`onclick = "toggleDisplay('info1')"`

Question			Expected response	Max mark	Additional guidance
18.	(a)		• level 1 pages • level 2 pages 	**2**	
	(b)		• required attribute • min and max values	**2**	<input type="number" name="price" **min = "200" max = "75000" required**>
	(c)	(i)	`clear:both`	**1**	
		(ii)	`#silver,#gold`	**1**	Accept use of grouping appropriate descendent selectors, for example: `section h1,section ul/li` Do not award if candidate writes additional CSS rule.
	(d)	(i)	• there is margin added to each side of the image • this makes the width of the div smaller than the two images/ takes the width of the content to 1240 pixels • so the images will be displayed vertically	**2**	Award 1 mark for each bullet. Maximum 2 marks.
		(ii)	Smartphones may not have the screen resolution/number of pixels required.	**1**	Any reference to size must reference resolution/number of pixels.
	(e)		• paragraphs are block elements/take up the whole width of its container • anchors are inline element/do not start on a new line and only take up as much width as necessary	**2**	

[END OF MARKING INSTRUCTIONS]