National
Qualifications
2022

2022 Computing Science

Advanced Higher

Finalised Marking Instructions

General marking principles for Advanced Higher Computing Science

Always apply these general principles. Use them in conjunction with the detailed marking instructions, which identify the key features required in candidates' 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.

(e) Award marks as per the detailed marking instructions, regardless of minor syntax errors, if the intention of the coding is clear.

(f) 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.

(g) If a candidate scores through their entire response to a question 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.

(h) 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.

(i) 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.

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

Marking instructions for each question

Section 1 – Software design and development

Question			Expected response	Max mark	Additional guidance
1.			Data structure: Double linked list Operation: Adding a new node	**2**	1 mark for correct data structure 1 mark for correct operation Note: answers must mention double linked list to gain mark for correct data structure
2.	(a)		Line 4: not found and low <= high Line 9: target > list[mid]	**2**	1 mark for each correct condition Also accept: Line 4: not found and high >= low Line 9: list[mid] < target
	(b)		The contents of the array are unsorted.	**1**	
3.	(a)		[4, 7, 12, 5, 13, 6]	**1**	
	(b)		[4, 5, 7, 6, 12, 13]	**1**	
4.	(a)		The `Plane` class will inherit the properties/instance variables and the methods from both the `Aircraft` class and the `FixedWing` class.	**2**	1 mark for properties/instance variables and methods 1 mark for `Aircraft` and `FixedWing` classes

	(b)	(i)	`Aircraft()`
			Or
			`FixedWing()`
				1	Accept `FixedWing()` and `Aircraft()`

	(b)	(ii)	`CLASS Helicopter INHERITS Aircraft WITH { INTEGER numberRotors, STRING engineType }`	**2**	1 mark for indicating inheritance of superclass 1 mark for additional parameters with correct data types Note: solutions written in other languages are acceptable
	(c)		The `OVERRRIDE` statement is used to redefine/modify the inherited method. This means that the calculation performed by the `Glider` class would be different from the calculation inherited from the `Aircraft` superclass. This is an example of polymorphism.	**2**	1 mark each for any two statements

Question			Expected response	Max mark	Additional guidance

	(d)	(i)	A new object of the `Plane` class has been instantiated and populated with a value for each instance variable.
			
			The instance variables of the `plane1` object include its own property in addition to those inherited from the `FixedWing` and `Aircraft` classes.
				2	1 mark for instantiation of an object
					
					1 mark for explaining or exemplifying the inheritance of instance variables from super classes
					
					Note: accept the following as an alternative to an explanation of inherited instance variables:
					`aircraftID = "ABC123"`
					`hoursSinceService = 0.0`
					`hoursToNextService = 100.0`
					`wingSpan = 28.9`
					`engineType = "jet"`

		(ii)	Declaration of array of objects called `fleet` with 76 elements that belongs to the `Plane` class. Assignment of values stored in `plane1` object to element 0 of the `fleet` array.	**2**	1 mark for declaration of an array of objects with 76 elements 1 mark for assignment of `plane1` object to the first element of the `fleet` array

		(iii)	`FUNCTION countServiceDue (ARRAY OF Plane group) RETURNS INTEGER`
			
			  `DECLARE totalPlanes INITIALLY 0`
			
			  `FOR index FROM 0 TO 75 DO`
			    `IF group[index].nextService() <=12 THEN`
			        `SET totalPlanes TO totalPlanes + 1`
			    `END IF`
			
			  `END FOR`
			  `RETURN totalPlanes`
			
			`END FUNCTION`
				3	1 mark for initialising, incrementing and returning a local variable such as `totalPlanes`
					
					1 mark for correct use of `fleet` array of objects (formal parameter `group` in this solution) within a fixed loop
					
					1 mark for correct use of `nextService()` method

Question			Expected response	Max mark	Additional guidance
5.	(a)	(i)	Player and robot movements must not go beyond the dimensions of the game board. The game much check for robot collisions and for capturing the player. The game must check for player win and game over.	2	1 mark each for any two additional functional requirements. Other answers possible. Note: functional requirements must relate to data structure and/or logic of the solution
		(ii)	For example: `DECLARE board AS ARRAY OF ARRAY OF STRING INITIALLY [[""]] <with 10 rows and 10 columns>`	1	1 mark for correct dimensions and string type Accept valid declaration of 2D array with correct dimensions in any programming language

		(iii)	`REPEAT 6 TIMES`
			`DECLARE empty INITIALLY false`
			`REPEAT`
			`   DECLATE randomX INITIALLY RANDOM(10)-1`
			`   DECLARE randomY INITIALLY RANDOM(10)-1`
			`   IF board[randomX, randomY] = "" THEN`
			`      SET board[randomX, randomY] TO "Robot"`
			`      SET empty TO true`
			`   END IF`
			`UNTIL empty = true`
			`END REPEAT`
				3	1 mark for fixed loop to generate 6 robot positions
					
					1 mark for checking randomly selected position is empty
					
					1 mark for assignment to random position within the grid
					
					Note: candidates should not be penalised if they don't subtract 1 to generate 0-9 grid if their subsequent answers indicate use of a 1-10 grid.

Question			Expected response	Max mark	Additional guidance

5.	(b)		
			```
			PROCEDURE moveUp()
			  FOR row = 0 TO 9 DO
			    FOR column = 0 TO 9 DO
			      If board[row, column] =
			      "Player" AND row ≠ 0
			      THEN
			          SET board[row - 1,
			          column] TO "Player"
			          SET board[row,
			          column] TO ""
			      END IF
			    END FOR
			  END FOR
			END PROCEDURE

			*Alternative Soln (validator)*
			PROCEDURE moveUp()
			  FOR row = 1 TO 9 DO
			    FOR column = 1 TO 9 DO
			      If board[row, column]) =
			      "Player" THEN
			          SET board[row - 1,
			          column] TO "Player"
			          SET board[row,
			          column] TO ""
			      END IF
			    END FOR
			  END FOR
			END PROCEDURE
			```
				3	1 mark for getting location of player using nested loop
					
					1 mark for checking that player is not already on the top row
					
					1 mark for setting new location by changing row position by -1 and removing from previous position
					
					Note: if candidate is using a different grid layout, `set board[row + 1, column] = "Player"` would be appropriate

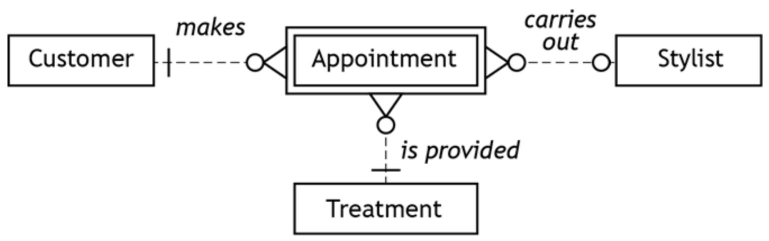
Question			Expected response	Max mark	Additional guidance
5.	(c)		1. set board(robotX, robotY) to ""	**5**	1 mark for clearing current position of the robot
			2. if playerX < robotX then subtract 1 from robotRow 3. if playerX > robotX then add 1 to robotX 4. if playerY > robotY then add 1 to robotY 5. if playerY < robotY then subtract 1 from robotY		1 mark for calculating new robot position closer to the player
			6. if board(robotX, robotY) = "" then 7. set board(robotX, robotY) to "Robot"		1 mark for assignment of robot to the new position
			8. else if board(robotX, robotY) = "Robot" then 9. set board(robotX, robotY) to "Rubble"		1 mark for checking for collision with another robot and creation of pile of rubble
			10. else if board(robotX, robotY) = "Rubble" then 11. set board(robotX, robotY) to "Rubble"		
			12. else if board(robotRow, robotY) = "Player" then 13. display game over message 14. end if		1 mark for checking for player capture Note: steps 10 & 11 not required in solution

Section 2 – Database design and development

Question			Expected response	Max mark	Additional guidance

6.			Corrections required to Lines 3, 4 and 5. For example:
			
			Line 3: `firstName VARCHAR (25)`
			Line 4: `lastName VARCHAR (40)`
			Line 5: `contractType VARCHAR(10)`
			
			Missing Lines:
			`PRIMARY KEY (doctorID)`
			`CHECK contractType IN ('Consultant', 'Junior', 'Locum')`
				3	1 mark each of the following:
					
					• Stating a valid size of `VARCHAR` in Lines 3, 4 and 5
					• Stating primary key of the table
					• Restricting values stored in `contractType`
					
					Notes:
					Accept any reasonable size for `VARCHAR` in Lines 3, 4 and 5
					Don't penalise for use of `NULL`

7.			A surrogate key could be used to replace the current primary key which is a compound key. Use of a surrogate key would make it easier to reference historical data. It would also improve the performance of queries that involve a join with the `Walk` table. In this situation, it would allow a walker to record completion of a route more then once on the same day.	**2**	1 mark for purpose of the surrogate key 1 mark for one appropriate benefit

Question			Expected response	Max mark	Additional guidance
8.	(a)	(i)	Legal feasibility The description makes refers to personal data that would be stored by the system and this has an implication for GDPR.	**1**	1 mark for correct type of feasibility with reason Note: mark should not be awarded if no justification is provided
		(ii)	This shows that the Registered User actor is a form of the User actor and will inherit all of its characteristics.	**1**	1 mark for the correct description
	(b)		ER Diagram – see below for solution  Customer makes Appointment carries out Stylist is provided Treatment	**4**	1 mark for strong entities correct 1 mark for weak entities correct 1 mark for two mandatory relationships correct (Appointment → Customer and Appointment → Treatment) 1 mark for four optional relationships correct (Customer → Appointment, Appointment → Stylist, Stylist → Appointment and Treatment → Appointment)
	(c)		A: `BETWEEN` B: `HAVING COUNT(*)>=3;`	**2**	1 mark for `BETWEEN` operator 1 mark for `HAVING` with `COUNT`
	(d)	(i)	customerID = ANY	**1**	1 mark for correct condition
		(ii)	`IN` would be used in the search criteria to specify the ID of the stylists. For example: `stylistID IN (2, 5, 7)`	**1**	1 mark for correct description of use of `IN` Accept correct code as alternative to description
	(e)		Query Output: John Smith Explanation: For example: The query will select the only record from the `Appointment` table that has no entry in the `stylistID` field. This record has a `customerID` of 1 which matches the customer name in the expected output.	**2**	1 mark for query output 1 mark for explanation of how this output would be generated.

Question			Expected response	Max mark	Additional guidance
8.	(f)	(i)	During final testing, all tasks required to satisfy the test case are carried out by one of the development team who adopts the persona described.	**2**	1 mark for final testing 1 mark for description
		(ii)	The solution shown is not fit for purpose because there is no option to make a follow-up booking so each booking needs to be made individually. The solution shown is fit for purpose since it is possible to book a follow-up appointment by making an entirely new appointment.	**1**	1 mark for indicating that solution is not fit for purpose together with a supporting reason OR 1 mark for indicating that solution is fit for purpose together with a supporting reason

Section 3 – Webs design and development

Question			Expected response	Max mark	Additional guidance

9.			`'<tr>`
			`<td>'.$walker['walkerID'].`
			`'</td>`
			`<td>'.$walker['walkerName']`
			`. '</td>`
			`</tr>';`
				2	1 mark for two `<td>…</td>` pairs within `<tr>` … `</tr>`
					
					1 mark for correct use of array and field names in correct sequence to match heading already displayed by code provided.

10.			Session variables are used to enable a website to retain information from one page to another. `session_start()` used when user logs in to store user login details `session_destroy()` used at logout to remove any stored data	**3**	1 mark for use of session variables to share values across pages of the website 1 mark for accurate description of `session_start()` function 1 mark for accurate description of `session_destroy()` function
11.	(a)	(i)	Legal feasibility The description makes refers to personal data that would be stored by the system and this has an implication for GDPR.	**1**	1 mark for correct type of feasibility with reason Note: mark should not be awarded if no justification is provided
		(ii)	This shows that the Registered User actor is a form of the User actor and will inherit all of its characteristics.	**1**	1 mark for the correct description
	(b)		`$apptID = $_GET["apptID"];`	**2**	1 mark for use of `$_GET` 1 mark for assignment to PHP variable using correct form element names Note: accept alternative PHP variable name

	(c)		`$servername = "db.hbh.com"`
			`$username = "Harvey"`
			`$password = "£dxG67*"`
			`$dbase = "hairbyharvey"`
			
			`$conn =`
			`mysqli_connect($servername,`
			`$username, $password,`
			`$dbase)`
			
			Acceptable alternative:
			`$conn = mysqli_connect`
			`("db.hbh.com", "Harvey",`
			`"£dxG67*", "hairbyharvey")`
				2	1 mark for correct use of `mysqli_connect()` to assign connection to `$conn` variable
					
					1 mark for parameters in correct order (server, user, password, database)
					
					Note: accept alternative PHP variable names
					
					Also: accept alternative solution that doesn't assign connection credentials to PHP variables

Question			Expected response	Max mark	Additional guidance

11.	(d)		Line 3
			`mysqli_query($conn, $sql);`
			
			Line 5
			`mysqli_num_rows($result);`
				4	1 mark for
					`mysqli_query()`
					
					1 mark for
					`($conn, $sql)` arguments with `mysqli_query()`
					
					1 mark for
					`mysqli_num_rows()`
					
					1 mark for `$result` argument with `mysqli_num_rows()`

	(e)		Use @media print instead of @media screen. Change the background colour from light grey to white, remove the second image and reduce the size of the font used to display the heading.
			
			Sample Code:
			`@media print {`
			    `body { background-color: white }`
			    `.mirrorImg { width: 200px; height: 200px }`
			    `.combImg { display: none }`
			    `.heading { font-family: Arabic; font-size: 12px }`
			`}`
				2	1 mark for use of `@media print` rather than `@media screen`
					
					1 mark for additional changes needed to produce the paper output. A minimum of two of the following must be included:
					- changing background colour to white or none
					- removing the second image
					- changing the font size in the heading
					
					Accept marks for correct code as alternative to a description

	(f)	(i)	During final testing, all tasks required to satisfy the test case are carried out by one of the development team who adopts the persona described.	**2**	1 mark for final testing 1 mark for description
		(ii)	The solution shown is not fit for purpose because there is no option to make a follow-up booking so each booking needs to be made individually. The solution shown is fit for purpose since it is possible to book a follow-up appointment by making an entirely new appointment.	**1**	1 mark for indicating that solution is not fit for purpose together with a supporting reason that refers to the evidence provided OR 1 mark for indicating that solution is fit for purpose together with a supporting reason

[END OF MARKING INSTRUCTIONS]