National Qualifications 2018

2018 Computing Science

Advanced Higher

Finalised Marking Instructions

SQA ©

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) Marking should always be positive. This means that, for each candidate response, marks are accumulated for the demonstration of relevant skills, knowledge and understanding: they are not deducted from a maximum on the basis of errors or omissions.

(c) If a specific candidate response does not seem to be 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) Marks should be awarded regardless of spelling as long as the meaning is unambiguous.

(e) Candidates may answer programming questions in any appropriate programming language or pseudocode. Marks should be awarded, regardless of minor syntax errors, as long as the intention of the coding is clear.

(f) Where a question asks the candidate to **describe**, the candidate must provide a statement or structure of characteristics and/or features. This should be more than an outline or a list. It may refer to, for instance, a concept, process, experiment, situation or facts in the context of, and appropriate to, the question. The candidates will normally be required to make the same number of factual/appropriate points as there are marks available for the question.

(g) Where a question asks the candidate to **explain**, marks should only be awarded where the candidate goes beyond a description, for example by giving a reason, or relating cause to effect, or providing a relationship between two aspects. These will be related to the context of the question or a specific area within a question.

(h) Credit should be given where a labelled diagram conveys clearly and correctly the response required by the question.

Question			Expected response	Max mark	Additional guidance
1.	(a)		Iterative prototyping/RAD/agile **(1 mark)** It is easier to change layout of user interface if end user feedback is received early on in the development. **(1 mark)**	**2**	1 mark for correct technique. 1 mark for benefit for developers.
	(b)		As the data structure is FIFO, the instructions the team programs will be stored then executed in the correct order.	**1**	1 mark for appropriate explanation that relates to the scenario.

	(c)	(i)	`1. if pointer < stackSize then`
			`2.  set pointer = pointer + 1`
			`3.  set stack[pointer] = instruction`
			`4. else`
			`5.  display stack full message`
			`6. end if`
				3	1 mark each for any 3 of the following:
					
					• for IF to check for overflow
					• for pointer increment
					• for assignment of instruction
					• for stack full message.

	(c)	(ii)	Linked list is a dynamic data structure which allows instruction sets of differing lengths to be stored. **(1 mark)** Only data that needs to be stored will be added to the linked list. **(1 mark)** Use of a linked list provides a better high-level model for the programmer. **(1 mark)** As a consequence of providing a better high level model the solution is likely to be less error-prone. **(1 mark)**	**2**	1 mark each for any 2 of the following: • dynamic nature of linked list structure • ease of insertion and deletion of data • storage of only necessary data • better high level model • less error–prone solution.

Question			Expected response	Max mark	Additional guidance
1.	(d)		Alternative answers are possible. 1. set low = 0 2. set high = 49 3. set found = false 4. enter team name 5. start conditional loop 6. set mid = (low + high)/2 7. if results[mid,0]<teamName then 8. set low = mid + 1 9. end if 10. if results[mid,0]>teamName then 11. set high = mid - 1 12. end if 13. if results[mid,0]=teamName then 14. set found = true 15. end if 16. loop until (found = true) or (low > upper) 17. if found = false then 18. display 'not found' message 19. else 20. display data for team name 21. end if	**5**	1 mark for initialisation of low, high and midpoint. 1 mark for entering team name and displaying data for team name. 1 mark for suitable conditional loop (accept upper<low). 1 mark for correct updating of low and high. 1 mark for correct use of 2D array. NOTE: not found message is not required for full marks. Alternative line 20 `display results[mid,1], results[mid,2], … results[mid,5]`
	(e)		Various answers are possible. For example: **Ethical Issue:** Ethical issue may refer to decisions made by a driverless vehicle – for example, when deciding whether to crash the vehicle into a wall (injuring the only passenger as a result) in order to avoid hitting an on-coming car that has several passengers. **(1 mark)** **Social Issue:** Social issue may relate to the impact of driverless cars on employment, for example, no need for taxi drivers or truck drivers. **(1 mark)**	**2**	1 mark for an appropriate description of an ethical issue. 1 mark for an appropriate description of a social issue.

Question			Expected response	Max mark	Additional guidance

2.	(a)		`CREATE TABLE User`
			`(`
			`userName varchar(20)`
			`email varchar(30)`
			`password varchar(20)`
			`PRIMARY KEY(userName)`
			`);`
				2	1 mark for creating table with correct fields and data types; text, char or varchar acceptable; any valid length acceptable. **Do not accept string**.
					
					NOTE: minimum acceptable fields sizes are 6, 17, 6.
					
					1 mark for assigning primary key.

	(b)		`<form action="update.php" method="post">` `Username <input type = "text" name = "username">  ` `New Password <input type = "password" name = "password1">` ` ` `Re-enter Password <input type = "password" name = "password2">` ` ` `<input type= "submit" value="Update">` `</form>`	**3**	1 mark for form with correct action and method. 1 mark for 3 labels with 3 text fields (accept password type for both passwords). 1 mark for submit button with correct label. NOTE: Input type for password can be text or password. NOTE: button tag could be used as alternative for submit button `<button type= "submit">Update </button>`

	(c)		`<?php`
			
			`$username = $_POST["userName"];`
			`$pass1 = $_POST["password1"];`
			`$pass2 = $_POST["password2"];`
			
			`if ($pass1 != $pass2) {`
			`  echo ("Error – Passwords do not match");`
			`}`
			`else {`
			`  $sql = "UPDATE User SET password = '$pass1' WHERE username = '$username'";`
			`  mysql_query($sql, $conn);`
			`};`
			
			`?>`
			
			NOTE: accept also
			`mysqli_query($sql, $conn);`
				5	1 mark for assigning POST values to variables (Note: if GET has been used in part (b), accept assigning GET values to variables).
					
					1 mark for conditional statement used to compare pass1 with pass2 With appropriate error message.
					
					1 mark for correct SET clause of UPDATE query.
					
					1 mark for correct WHERE clause.
					
					1 mark for execution of SQL query.
					
					NOTE: POST values should match HTML element names in part (b) above.
					
					NOTE: ignore sequence of SET and WHERE clauses.

Question			Expected response	Max mark	Additional guidance

2.	(d)		`SELECT userName, COUNT(*) FROM Message`
			`GROUP BY userName;`
			
			NOTE: any field can be used with the COUNT function.
				3	1 mark for correct use of aggregate function.
					
					1 mark for correct fields and table.
					
					1 mark for correctly grouping by user name.

	(e)		Various answers are possible. For example: **Legal Issue:** By agreeing to the terms and conditions, users hand over the IPR (including copyright) of any images that they upload to the website. **(1 mark)** **Economic Issue:** Website users will lose potential income from photos used by the website company in publicity or marketing material. **(1 mark)**	**2**	1 mark for any appropriate legal issue. 1 mark for any appropriate economic issue. NOTE: issues must relate to website users and not the business.

Question			Expected response	Max mark	Additional guidance
3.	(a)		`CLASS ForSale INHERITS House WITH {REAL askingPrice, STRING closingDate, BOOLEAN underOffer, BOOLEAN sold}`	**2**	1 mark for an indication of inheritance. 1 mark for 'WITH' or equivalent to indicate additional properties.
	(b)	(i)	Instantiates an object (called saleHouse1) that belongs to the `ForSale` class (with given values/actual parameters).	**2**	1 mark each for any 2 of the following: • instantiates/create object • class or using values/parameters • constructor method invoked.
		(ii)	The procedure call invokes the `updateBedrooms` method of the `saleHouse1` object (with the value/actual parameter 3). The method `updateBedrooms` is inherited from `House` superclass. A value of 3 is assigned to the property `bedrooms` (in the `saleHouse1` object).	**2**	1 mark each for any 2 of the following: • invoking method (or procedure) • use made of inheritance • updating property.
	(c)	(i)	Encapsulation has been used to restrict access to the description variable/property. This property can only be accessed through methods in `ForSale` class (or inherited methods from `House` class).	**2**	1 mark for encapsulation or use of private property. 1 mark for method required to use/change property.
		(ii)	Since the `updateRentStatus` method is not part of `saleHouse1` object. OR The `updateRentStatus` method is only available to objects that belong to the `ForRent` class.	**1**	

Question			Expected response	Max mark	Additional guidance

3.	(d)		`DECLARE total AS INTEGER INITIALLY 0`
			`DECLARE average AS REAL INITIALLY 0.00`
			`DECLARE count AS INTEGER INITIALLY 0`
			
			`FOR EACH house FROM houseList DO`
			`  IF house.getTown() = targetTown THEN`
			`  SET total TO total + house.getValue()`
			`  SET count TO count + 1`
			`  END IF`
			`END FOR EACH`
			
			`SET average TO total / count`
			
			Alternative loop
			`FOR house FROM 0 TO <end of list-1> DO`
			`  IF houseList[house].getTown() = targetTown THEN`
			`  SET total TO total + houseList[house].getValue()`
			`  SET count TO count + 1`
			`  END IF`
			`END FOR`
				4	**Award 1 mark each for any 4 of the following:**
					
					• appropriate loop to traverse the array
					• conditional statement, using houseList array, needed to total correct town
					• correct use of `getTown()` method
					• correct total/average that makes use of a count
					• correct use of `getValue()` method.

	(e)	(i)	Perfective maintenance since implementing additional functionality.	**1**	1 mark for perfective maintenance with appropriate justification. NOTE: no marks for perfective without justification.
		(ii)	New sub-class of `ForRent` class (or `House` class) with one additional property. OR A new property could be added to the `ForRent` class.	**1**	1 mark for new sub-class. OR 1 mark for new property in the `ForRent` class.

Question			Expected response	Max mark	Additional guidance

4.	(a)	(i)	`RECORD Applicant IS {`
			`  INTEGER refNumber`
			`  STRING lastname`
			`  STRING firstname`
			`  STRING contactNo`
			`  STRING contactEmail`
			`  INTEGER numTickets }`
			
			`DECLARE people AS ARRAY OF Applicant INITIALLY [] * 6129`
				2	1 mark for suitable record structure with variable names and types for each item to be stored.
					
					1 mark for declaration of array of records to store 6129 records (or 6128 depending on language used).
					
					NOTE: contactNo must be STRING

		(ii)	`$server="festival18";`
			`$user="admin";`
			`$password="ticket18";`
			`$database="hopeful.dbase";`
			
			`$conn=mysql_connect($server, $user, $password);`
			`mysql_select_db($conn, $database);`
			
			`$sql="SELECT * FROM Applicants";`
			`$result=mysql_query($conn, $sql);`
			
			`close($conn);`
			
			Note alternative to use of mysql_connect and mysql_select:
			`mysqli_connect($server, $user, $password, $database);`
				3	1 mark for connection to database server and selection of database.
					
					1 mark for appropriate SQL query.
					
					1 mark for allocation of query results to server-side variable.
					
					NOTE: Many alternative answers are possible including those that make use of values rather than variables that have had values assigned to them.

	(b)		One possible answer is shown. Many alternative answers exist.
			
			`1. set swapped to true`
			
			`2. while swapped = true`
			`3.  set swapped to false`
			`4.  start fixed loop from 1 to 6128`
			`5.  if people[loop].lastname and people[loop-1].lastname are in wrong order then`
			`6.  set temp to people[loop]`
			`7.  set people[loop] to people[loop-1]`
			`8.  set people[loop-1] to temp`
			`9.  set swapped to true`
			`10.  end if`
			`11.  end fixed loop`
			`12. end while loop`
				5	1 mark for outer loop that makes use of Boolean variable.
					
					1 mark for inner loop from 1 to 6128 (or 0 to 6127).
					
					1 mark for correct updating of Boolean variable.
					
					1 mark for comparison of last name of 2 adjacent records.
					
					1 mark for swapping adjacent records that indicates use of temporary variable.

Question			Expected response	Max mark	Additional guidance
4.	(c)		One possible answer is shown. Many alternative answers exist. The following solution assumes that a 1D array of integer values called `allocated` is used to store the number of tickets allocated to each applicant. All values in the array are initialised to 0 indicating that no tickets are allocated at the start. 1. set totalAllocated = 0 2. start fixed loop from 0 to 6128 3. set each element of allocated array = 0 4. end loop 5. set betweenLimits to false 6. while totalAllocated <=4000 or betweenLimits = true 7. select random number btw 0 and 6128 8. if (people[random].numTickets + totalAllocated <=4000) and (allocated[random] = 0) then 9. add people[random].numTickets to totalAllocated 10. set allocated[random] to people[random].numTickets 11. If totalAllocated >=3997 then set betweenLimits to true 12. end if 13. end while	**5**	1 mark for randomly selecting an applicant from the full list of applicants. 1 mark for check that an applicant has not been previously selected. 1 mark for ensuring successful applicants receive their full allocation. 1 mark for appropriate use of the array of records defined in part (a) (i) above (numTickets field). 1 mark for allocating at least 3997 and no more than 4000 tickets.

[END OF MARKING INSTRUCTIONS]

