BCS Higher Education Qualification

Certificate in IT

October 2022

EXAMINERS' REPORT TEMPLATE

Software Development

General comments[1]

Questions Report:

A1	A1 Syllabus 6.1
	Common issues were misunderstanding what the compiler did, mixing it up with the interpreter and not being aware of many of the pros and cons of one translator compared to the other. Answers here tended to give good accounts of the role of the compiler as translating high level language into a low-level object or executable code. Answers on the role of an interpreter were generally less complete. Most candidates were able to describe the process of source code scanning directly into an execution but tended to avoid explanations of the interpreter as an intermediary in situations such as executing precompiled bytecode. The subsection on strengths and weaknesses of developing code using either compilers or interpreters was generally not well answered with many not focusing on the particulars such as compiler having better secured executables or the interpreter being essentially cross platform. Sub sections d) -e) of the question concerned the integrated development environment. An explanation of the term that focused on the IDE as a set of tools used to create software would have been the best approach to gain maximum marks, however many candidates gave descriptions of the graphical interface and easy editing as the main purpose of an IDE. The final subsection e) of this question required the identification and description of five main features expected of an IDE. Many candidates approached this question from the point of view of the IDE being an interface for database administration or package management and not a programming development environment, those answers gained few, if any marks. Those answers which focused on the IDE features such as stack content examination, setting breakpoints and similar functions tended to gain higher marks
A2	A2 Syllabus 2.1
	This question had four sub sections i) – iv). Covering programming paradigms and required an explanation for each of the strengths and weaknesses associated with each approach. Answers from structured and object-oriented perspectives were generally well answered. The functional and concurrent programming perspective was not well answered by most. Many answers tended to confuse concurrent programming with parallel programming. Many answers on the functional approach simply restated the answers given for the structured programming approach. Very few answers acknowledged immutability, lack of side effects and similar properties of functional programming.
A3	A3 Syllabus 3.3

Phil Barker	This question had subsections a) – e) and concerned aspects of stacks and queues. Most candidates could explain the stack and queue data structures. The stack operations such as push, pop, and peek were generally well described with appropriate pseudocode. However, attempts at pseudocode for an array implementation of a queue with the enqueue, dequeue and peek operations was generally poorly done. Many answers confused insertion and removal of items from the queue with examination of the queue. Many candidates omitted this part of the question.
A4	A4 Syllabus 3.3,3.4,7.1
Phil Barker	a) This question was not a popular choice amongst candidates This subsection concerned the bubble sort algorithm. The requirement to write a program that implements the technique was not accomplished by the majority of candidates, many candidates gave discursive versions with pseudocode. The question required code that implemented the algorithm the most efficient bubble sort is accomplished using two loops. Many attempts omitted this approach. Other attempts used while loops which went beyond the required iterations and although this approach gave a correct answer the solution was not efficient. b) Very few candidates fully attempted this part of the question which was to amend the answer given in part a). For those candidates who found part a) challenging this part tended to be ignored. The question asked for the code to implement a change so that a user could enter a search integer and this would be used to sort the array until that number was reached. Only those candidates who did not go down the while loop route in part a) were able to offer a reasonable program. Subsequently very few marks were gained from this sub section of the question
B5	B5 Syllabus 1.2,1.3,1.4,3.3
	For part a) the requirement was for a description of a suitable data structure for storing the example array of contiguous data of the same type. Many candidates chose a simple array, only a few candidates correctly suggested a 2D array as the most suited. Part b) Required the candidate to write code or pseudocode to show how a user chosen input integer which is between -50 and +50 or default of 0 can be added to the appropriate location in the array structure. Many answers correctly wrote code to check for user input range. However subsequent checking against array values and insertion of updated value proved difficult for many. Many candidates found difficulty in articulating pseudocode or program code to implement their solution. In some cases, this could be seen as a result of not choosing a sensible data structure (from part a). Those candidates who chose the 2D array or nested list data structure tended to gain high marks.
B6	B6 Syllabus 3.4
	Most candidates made a good job of describing a linear search. The descriptions of binary searching were more varied – one or two candidates were wrong-footed by the word “binary” and started to discuss binary representations of data rather than search algorithms. Many candidates were clear about the general principal of a binary search without being clear about the way in which the algorithm works with sorted data by eliminating sections of the structure from consideration. Very few candidates were able to discuss the time complexity of the algorithm in any detail.

	For part a) on linear search most candidates gave a correct description of starting at one end and continuing to the other to look for the search item. Many ignored the fact that the list was already described as sorted and gave a sorting routine or description, which was unnecessary. For part b) on binary search many candidates correctly described the algorithm as recursive and required a sorted list. The divide and conquer methods were successfully described by a majority of candidates although few gained full marks by not completing the explanation with showing (correctly) the rejection of the division of the array data that does not contain the sought value. Part c) proved more problematic for many candidates with a few ignoring this part completely or giving partial answers. Very few answers noted big O descriptions of algorithm efficiency
B7	B7 Syllabus 2.4
	This question was concerned solely with software quality assurance and the importance of quality assurance and how it can be measured. Many candidates successfully answered this question and gave good accounts of quality through testing techniques and equally quality through non-functional aspects. Some candidates neglected the non-functional aspects and solely based their answer on the various code testing techniques, thus failing to explain half of the question. A few candidates deviated to discussions of rating various service providers or off the shelf package solutions and neglected to widen the discussion and focus on the question.
B8	
	Sub section a) is concerned with a flowchart. Many candidates failed to gain more than half marks for this part of the question as they were unable to correctly articulate the flowchart nodes for checking and iteration. Marks were gained by a majority for carrying out test1 checking the correct number of characters and test 2 where a majority were able to state the use of tests for upper/lower case or integer or character input. However, few were able to provide a flow that correctly gave the decision nodes and feedback loops. Sub section b) is about aspects of password checking. This part of the question was largely answered correctly by suggesting the use of special characters or use of password strength measures in feedback
B9	
	A wide range of quality in the answers to this question. Many candidates provided answers that clearly described classes, methods and objects but some were very muddled. Where answers were unclear, this was usually because the candidate did not appreciate the difference between a class and an object. Sub section a) dealt with explanations of common terms used in object-oriented programming. Many candidates were able to give a reasonable explanation of classes, methods and objects. However, a good number of answers failed to fully distinguish between the terms. In some cases, marks were lost by wrongly identifying variables as methods and confusing the terms class and object. Sub section b) proved reasonably difficult for many with many answers failing to give two benefits of using the object-oriented approach. Answers that mentioned modularity or information hiding gained higher marks as did answers suggesting code reuse. In general answers tended to be quite brief and not full enough to gain maximum marks.

	There was also some confusion about methods leading to some poor answers to part (b) where candidates were asked to discuss advantages of OOP – the weaker answers showed that the candidate did not really understand the way in which OOP supports information hiding.
B10	B10 Syllabus 6.1
	This question has four sub sections concerning errors in programming code. Candidates in general were able to give adequate answers for syntax errors and type errors but less successful in describing logic and overflow errors. Quite a few candidates incorrectly referred to type errors as errors in typing in the code on an IDE. Maximum marks were gained by a few candidates who were able to complete the question by indicating the appropriate tool to use in finding the error
B11	B11 Syllabus 2.1
	This question is in two subsections dealing with programming code error detection and tracing Sub section a) required that the candidate provide a trace of the execution of faulty code. Most candidates were able to provide a good trace but, in some cases, lost marks by confusing the expected output with the actual output. In many cases this was a result of some tracing tables where candidates appeared to have correctly identified the calculated output but gave the wrong value as the answer. Sub section b) required the re-writing of the supplied faulty code based on identification of the errors. A substantial number of answers were unable to correctly identify the error, even in cases where maximum marks were gained from a fully correct trace of the code. An inability to identify the errors in the faulty code subsequently resulted in the rewrite code being wrong. This sub part of the question was worth half of the total marks and only minimal marks, if any were gained from the majority of attempts
B12	B12 Syllabus 2.5
	This question concerns the definition of quantum computing and a description of what it is Many candidates were able to give a reasonable basic definition but few mentioned terms such as quantum mechanics, and basic computational units such as cubits, which would have gained more marks. Areas for the application of quantum computing such as cryptography or general AI gained good marks. Many candidates however simply based answers on existing more traditional computing and extended it to a faster or bigger data capacity. In many cases answers were very brief and covered one or two sentences, subsequently gaining few marks .

Additional Examiner comments:

Some students appeared unprepared for the exam, while others had a good grasp of the subject but poor exam skills. E.g. writing nearly a side of A4 for a few marks. Candidates are advised to review the exam guidance and techniques before an exam and also to use the new e-book on this module available from BCS.