BCS Higher Education Qualification

Certificate in IT

April 2025

EXAMINERS' REPORT

Software Development

Questions Report:

A1	
	Part a) Candidates produced a range of answers which generally attracted marks despite some apparent misunderstanding of what was required. Therefore, there was some flexibility in marking accepting different ways of file creation, either in code or using an IDE. However, it was important to state the key parameters that are always required to create a file. These include the location (file path) on the storage media (such as SSD, hard disk); the type of file with matching extension; the mode of access (r/w). Similarly storing data on an existing file is usually accomplished by ensuring the data is compatible with the type of data supported whether binary of text to ensure data integrity and security. Retrieving data from a file is again dictated by file operations that identify the location of the file to open for reading followed by recognising the format and organisation that was used when the file was stored. Part b) Most candidates were familiar with the different types of files and the key differences between them. However, marks were lost because many candidates either omitted a scenario or the explanation was unclear. Part c) Most candidates were familiar with text files, but some candidates did not appreciate that semi-formatted files are essentially a specialised form of a text file in that it has specific delimiters (i.e. commas) to make the file better structured. This allows rapid processing and ease of transforming/reading by high level data intensive applications such as databases (tables) or spreadsheets. Part d) This part required some coding knowledge relating the part a) iii). A very poor set of answers overall that generally lacked actual code examples and lacked the stepwise approach that was needed. The few successful attempts mostly used Python as the source code and described the functions/libraries provided by Python.
A2	
	Part a) This part was generally well answered with some lack of understanding of dry running and the manual approach generally used. It was also evident that many candidates could recall mainly by rote the key topics of black box and white box testing, but even so some candidates managed to get these mixed up. Part b) i) and ii) Most candidates understood what was meant by a test case though some answers did not clearly state the objectives of a test case.

	It was also apparent that many candidates lost marks because they missed off many of the parameters; these being title or summary; specifying the objectives; the context; the preconditions; the test data; the steps; the expected (to compare with the actual) result and so on. Subpart iii) as a consequence of failing to adequately answer subpart ii) those candidates were unable to provide a suitable answer. Overall, this part revealed a lack of practical knowledge of actually carrying out unit tests on code which is a key part of the "bread and butter" of software development. Part c) This question had two subparts requiring knowledge of QA techniques. Candidates were given a number of key principles or challenges of QA which were associated with a range of techniques that candidates were expected to identify.
A3	
	Part a) Most candidates were familiar with data structures such as lists, arrays, queues and stacks and most candidates could identify these as linear data structures. However, it was important to stress that although that linear data structures are organised sequentially but also the way they operate in code is different depending on the requirements of the program/application. So, it was important to explain how they are accessed; stored and manipulated. Non-linear data structures were not generally familiar to candidates with examples given that were linear data structures rather than hierarchical data structures such as trees graphs and networks. Part b) This part followed on from part a) directed at specific examples in particular arrays; lists; tuples. Most candidates had quite good knowledge of the different strengths and weaknesses of the first two data structures. Tuples were unfamiliar to many candidates. Tuples support mixed data types; size and values cannot be changed making them widely used as keys and dictionaries. Part c) Stacks and Queues were very familiar data structures, largely memorised from textbooks. Needless to say, this part mostly scored the highest marks in this question. A mark was often lost in not explaining the practical use of these data structures in actual programming tasks.
A4	
	A very unpopular question but there was a broad range of marks from the small number of candidates. Most candidates however scored poorly in part b) in particular. Given the small number of candidates it is hard to comment on specific details other than this was a more challenging question that required more applied knowledge particularly related to object-oriented programming attracting those candidates who were fluent in say either Java C# C++ or Python to draw examples upon.
B5	
	This question was the least popular choice amongst candidates. Part a) most candidates gave good and accurate representations of the hash table although some answers extended the number of entries (not penalised) that correctly answered the question. Part b) this part required a program snippet that provided a function to return a hash key for an integer. Many candidates embedded the function within a wider

	scoped program that calculates hash and places in the hash table, subsequently losing a mark. Part c) Some candidates made basic errors in incrementing values in the loop. Those candidates who had embedded the function to return a hash key for the given integer tended to lose track of the loop and lost some marks.
B6	
	Part a) concerning the relationship between debugging and testing was poorly answered by a good number of candidates. Many answers failed to note that debugging determines the cause and applies the fix. The order would correctly note that the fix is then tested. Mentioning that debugging is in early stages of development would have gained higher marks. Part b) This part was generally well answered. Better marks would have been obtained by mentioning some standard series of steps such as find the location with the problem and then identifying the reason for the problem.
B7	
	This question asked for four basic principles of good user interface design. Many candidates simply gave the same principle and examples under different headings. Around half of candidates gave good descriptions for at least two, no answers gained maximum marks.
B8	
	This question asked for an explanation of four software testing techniques. Answers were generally poor. Many confused stress-testing and load testing, with many answers simply repeating the same explanation for both. Functional testing was generally well answered although many incorrectly suggested it was white box testing; few answers acknowledged the role of functional testing in accepting testing and the point at which checking the client requirements have been met.
B9	
	This question asked for four benefits of using functions in programs. Part a) The large majority of candidates failed to achieve a pass mark. Many answers gained a mark for correctly naming a benefit such as avoiding repetition, abstraction, allowing code reuse amongst others. However, many answers failed to offer any further explanation and failed to gain an extra mark for each benefit. Part b) This required candidates to give an explanation of the meaning of byte code. A majority of answers wrongly defined byte code as binary code or machine code. A minority of candidates correctly mentioned byte code as a platform independent language or byte code being an intermediate code.
B10	
	Part a) This question required a flow chart to be drawn on measurements computed from supplied data. Approximately half of all candidates obtained a pass mark. Many answers showed an incorrect use of flowchart symbols. Very few answers were logically correct and although many candidates correctly identified and noted the test results in the correct symbol and provided correct difference calculations (placed in the correct symbol) the majority failed to show all three tests and calculations.

	Part b) required a discussion on pros and cons of storing values in csv files. Almost all of attempts were able to show at least one benefit , such as presented as readable text or easily exported as a flat file or table. Almost all of the candidates failed to give at least one con of CSV files, losing half of the marks available for this discussion. Acknowledging downsides such as lack of support for large datasets, any text file can be stored as .CSV and can have inconsistent row lengths or formatting and can exceed permissible file size for third party tools such as Excel would have gained extra marks.
B11	
	This question was a very popular choice amongst candidates with an overall pass rate over 81%. Part a) asked for reasons to explain why documentation is important in noting changes to software over lifetime of version updates/changes. Most candidates gave a good account of reasons and provided good valid points obtaining maximum or near maximum marks. A minority of candidates who did poorly in this part tended to give one or two terse answers and, in some cases, failed to read the question correctly and ignored the context of documenting changes to an existing program and discussed documenting in the early stages of initial program development. Part b) This part required candidates to contrast differences between internal and external software documentation. Many answers indicated a degree of uncertainty between both types of documentation. A good number of answers tended to repeat the same comparison for both types. Answers on external documentation showed a greater appreciation of what would be expected. However internal documentation proved very difficult for most. Answers here tended to focus on simple and vague statements on ‘for the user’, such as operating instructions and failed to recognise the context of this question. Very few answers to this part achieved half of the marks available.
B12	
	This question which required drawing a flowchart had worst overall performance in the paper. The pass rate is a little over 18%. Many candidates incorrectly used flowchart symbols, and many candidates failed to cover all five of the required Test/Result points. A common issue is the failure to correctly interpret the problem in terms of required testing of various conditions represented by flowchart positions in the process. The average mark for this question was approximately 2.55 with the highest mark of 8 being achieved by a small minority. Very few answers correctly represented correct stage of calling functions. It wasn’t clear to many candidates when the supplied string functions should be positioned in the flowchart. Performance in this question suggests candidates need more exposure and practice in the development and understanding of algorithms.