The BCS Professional Examination
Certificate

October 2004

Examiners' Report

Software Development

General

At Certificate level, the examination covers the syllabus as a whole, and asks candidates in part to describe what they know, and in part to apply what they know to specific situations. Overall, descriptions were well answered. Applications were not.

A main aim of the examination is to test candidate's ability to create algorithms. Sometimes, candidates offer solutions in code; although these can earn full marks, candidates should be aware that such excessive answers cost them time lost from earning further marks in other parts of the examination.

Candidates should read the question closely and follow its requirements in order to earn full marks. Too many candidates lost marks by not providing what the question asked for.
Important: Candidates should cross-check that they have attempted all parts of a question; often, questions are structured to lead a candidate into the mind-set to tackle a question; if a candidate only answers a first part for few marks and skips a second or development part for many more marks then that candidate has lost significant marks despite starting well.

Candidates seemed better prepared for descriptive answers such as what a web site should be able to do, and achieved fewer marks for more fundamental elements of software development such as data structures and software testing.

SECTION A

Question 1

The file '*studs*' contains details of students enrolled on their courses in alphabetic order of names. It has the following items in each student record:

<identifier>	<student name>	<course code>	<fees paid>
8-digit integer	30 characters	4 characters	1 character (restricted to Y/N)
45362722	HALFORD, Chris	BSC1	Y

Another file '*marks*' has student the identifier then six integer marks, these being the percentages obtained in six subjects:

<course code>	<mark-1>	<mark-2> …			<mark-6>	
8-digit integer	six two-digit integers					
45362722	55	67	44	45	59	38

There are more student records on '*studs*' than on '*marks*' as some students did not take the examinations. But validation has already been carried out so that every course code corresponds to one on the '*studs*' file.

Both files are stored in increasing identifier order.

(a) Under what possible circumstance could the *'marks'* file description prove inadequate? (2 marks)

(b) Write suitable data structures to contain the student name and one student record. hence write a suitable description for the files '*studs'* and '*marks'*.

(8 marks)

(c) Write an initial algorithm to read the entire 'marks' file sequentially, read the 'studs' file and match the identifier on both files. If the identifier on 'studs' file is less than on '*marks*' file, ignore it and read the next one. When a match is obtained a check is made to see of the student has paid his/her fees. If not, the message 'UNPAID FEES' is to be output. Otherwise the arithmetic mean of the six marks is evaluated then printed as output along with the student's name and course identifier. If this mean is greater than 39.9 %, and none of the individual marks is less than 30% the word 'PASS' is output otherwise 'NOT YET PASSED' is output.

You should not deal with error conditions in any of the data items. (20 marks)

Answer Pointers

ALGORITHM {suitable for development into a Pascal program}

```
VARIABLE DECLARATIONS   needed here {not all languages require this}
SET pass-count = fail-count = 0
OPEN ‘results’ file for READING, ‘studs’ file for READING
WHILE NOT end-of-file ‘results’ file
{it is most important to use the ‘results’ file as terminator, as ‘studs’ file has redundant records}

        READ  one record from ‘results’ file
        READ  one record from ‘studs’ file
        WHILE identifier of ‘results’ file IS NOT EQUAL TO identifier of studs’ file
                READ  next  record from ‘studs’ file
                {match of identifiers found}
         ** you are NOT expected to deal with the error situation where a match is not found **
        low = FALSE
        IF  NOT fees-paid from ‘studs’ file THEN
                BEGIN
                PRINT “UNPAID  FEES”
                add 1 to unpaid-count
                END
                ELSE
                BEGIN
                form average mark from six marks of ‘marks’ file        *development  1
                low →TRUE if any mark  LESS THAN 30.0                  *development  2
                IF average-mark > 39.9  AND   NOT low
                        THEN  BEGIN
                        PRINT “PASS”
                        add 1 to pass-count
                        END
                        ELSE
                        BEGIN
                        PRINT “NOT YET PASSED”
                        add 1 to fail-count
                        END
        END   {main loop}
        PRINT  pass-count “students passed”
        PRINT  fail-count  “students not yet passed”
        PRINT   unpaid-fees  “students have not paid fees”
        END PROGRAM
```

Examiner’s Guidance Notes

The question wanted to test candidates for their ability to formulate an algorithm. Many answers were too large, written out in code rather than an algorithm. It is not necessary to write the code of a solution, though the code often reveals the design behind it. However, it costs a candidate too much time to write code compared with writing design-level algorithmic constructs.

Specifically, answers to part(b) are expected to be used by reference in part (c); there is no need to copy out the same data structures. Again, there is no explicit penalty for this, but candidates lose opportunity to earn marks elsewhere because of extra time spent to rewrite already-provided answers.

Question 2

The algorithm given below implements the progressing relationship:

line	**left side**	**right side**
1	1*9 + 2	= 11
2	12*9 + 3	= 111
3	123*9 + 4	= 1111
4	1234*9 + 5	= 11111

a) Write the next two lines of the relationship. **(2 marks)**

The algorithm below demonstrate this relationship by separately calculating the left side and right side of this relationship:

Algorithm

```
line no.   instruction
   1          READ lim
   2          n = 0  pl = 0  pr = 1
   3          WHILE n < lim  DO
   4          n = n + 1
   5          le = pl*10  + n
   6          ln = le*9 + n + 1
   7          rn = pr*10 + 1
   8          WRITE “left side” ln “right side” rn
   9          pl = le
  10          pr = rn
  11          ENDWHILE
```

b) Dry-run the algorithm with ‘lim’ input as 4. **(14 marks)**

c) Devise more meaningful names for the memory locations (identifiers) used in the algorithm and place it in a program with appropriate declarations and more meaningful output. **(14 marks)**

Answer Pointers

(a) *Write the next two lines of the relationship.* **(2 marks)**

12345 *9 + 6 = 111111
123456*9 + 7 = 1111111

(b) *Dry-run the algorithm with 'lim' input as 4.* **(14 marks)**

line no.	working/logic	n	le	pl	pr	rhs	lhs	OUTPUT/comments	other
1		0	?	0	1	?	?		
2	READ								lim=4
3	WHILE/true								
4		1							
5			1						
6							11		
7	1*9 + 1 + 1					11			
8	WRITE							left =11 right = 11	
9				1					
10					11				
3	WHILE/true								
4		2							
5	10*1 + 2		12						
6	12*9 + 3						111		
7						111			
8								left =111 right = 111	
9				12					
10					111				
3	WHILE/true								
4		3							
5	10*12 + 3			12 3					
6	9*12 + 4						1111		
7					111 1				
8	WRITE							left=1111 right=1111	
9									
10									
3	WHILE/false							loop terminates	

(c) *Devise more meaningful names for the memory locations (identifiers) used in the algorithm and place it in a program with appropriate declarations and more meaningful output.*

(14 marks)

```
PROGRAM allone(INPUT,OUTPUT);
VAR lhs,rhs,n,largeno,left,prevleft,prevright:INTEGER;
continue:BOOLEAN;
BEGIN
WRITELN('program to test all-the-ones integer sums');
WRITELN('J.B.WILFORD March 2003');
WRITELN;
WRITELN('input how many demonstrations'); READ(largeno);
{large number condition in question/largeno := maxint DIV 10;}
n := 0;  continue := TRUE;
prevleft := 0; prevright := 1;
WHILE (n < largeno) AND continue DO
        BEGIN
        n := n +1;
        left := prevleft * 10 + n;
        lhs := left * 9 + (n + 1);
        WRITE(n:3,' left side = ',left:4,' * 9 ');
        WRITE(' + ',(n+1):2,' = ',lhs:4);
        rhs := prevright * 10 + 1;
        WRITELN('   ','right side = ',rhs:4);
        prevleft := left;
        prevright := rhs;
        IF rhs <> lhs THEN continue := FALSE;
        END;
WRITELN('progression demonstrated for ', (n-1):5, ' integers')
END.
```

Examiner's Guidance Notes

The tabular method of trace is both quick to create and easy to show execution paths. Other, more voluminous answers are acceptable but to earn equivalent marks they always take much longer to write-out.

Some few mistakes of assignment are only penalised once, though they may cascade and create consequential errors.

Part (c) was largely ignored although there were significant marks available. What was wanted was an understanding of the algorithm's purpose so that more meaningful names could be used for the variables. Answers that copied the algorithm as given showed little understanding.

Question 3

a) Write a procedure, in Pseudocode, or Structured English or a programming language with which you are familiar, that implements *sorting* as follows:

The function should *sort* integer data in an array named MATERIAL, of size LENGTH, such that the elements of MATERIAL are arranged in *ascending order.* Both MATERIAL and LENGTH are defined as globals to the function. (15 marks)

b) Dry-run your procedure with the following data:
LENGTH = 6 MATERIAL =

2
10
4
8
6
12

(15 marks)

Answer Pointers

a) Procedure SORT should

- Initialise local variables including some flag or marker that an exchange of position has been made, initially 'null' 3 marks
- set up a double loop outer loop clearly predicated at end on some test of whether an exchange has been made or not 3 marks
- set up a inner loop to pass over the array, comparing pairs of data and exchanging them if first of pair is larger than second of pair. If an exchange is made, the 'exchange flag' is set non-null to cause another inner-loop execution. 3 marks
- Control of inner loop is determined by LENGTH and an appropriate test with the value of the array index (i.e. whether p and p+1, or p-1 and p). 3 marks
- Function terminates when inner loop passes over the data without many any exchanges of position. 3 marks

b) For the dry run, a table of variables with the time-sequence of their changing values was expected. Dry Run should show something like

Ex Flag	1st loop	2nd loop	P	P+1	MATERIAL(P)	MATERIAL(P+)	Swap test and flag setting
Preset non-Null	Yes, + set Flag = null	yes	1	2	2	10	no
Not-Null		yes	2	3	10	4	Yes, Flag=non-Null
		yes	3	4	10	8	Yes
		yes	4	5	10	6	Yes
		yes	5	6	10	12	no
Not null	Yes, + set Flag = null	Yes	1	2	2	4	no
		Yes	2	3	4	8	No
Not null		Yes	3	4	8	6	Yes, Flag=non-Null
		Yes	4	5	8	10	No
		Yes	5	6	10	12	No
	Yes, + set Flag = null	Yes	1	2	2	4	No
		Yes	2	3	4	6	No
		Yes	3	4	6	8	no
		Yes	4	5	8	10	No
		Yes	5	6	10	12	No
	Stops, flag still null.						

Examiner's Guidance Notes
Candidates who attempted part (a) of this question did very well. Common errors were an algorithm with only one loop, and the omission of some test for premature termination when no values were exchanged but the loop counters had not reached their limits.

For part (b), the table was often incomplete or created in a way that the method of creation was not traceable by the examiner, so the candidate's method of dry-run was not apparent. Some candidates tackled the layout differently, and marks were given where this was also traceable and understood.

Question 4

a) Describe a program generator tool (such as a software development environment, SDE, or software development kit, SDK, or CASE program generator) with which you are familiar. Explain the facilities and functions that it has, and the purpose of the software that it produces. (15 marks)

b) What do you think are the benefits of using such program generator tools? How do they change the quality of product, the speed of software development, and the satisfaction of the end-user? Give reasons for your answer. (15 marks)

Answer Pointers
Part (a)

A generator tool has:
a) A repository for storing parts and part-relations
b) Support for a disciplined method, such as diagramming DFDs, ERDs, Use Cases, etc.
c) Some method of compile-and-test, usually visual on-screen comparison of execution state and source-code.

d) Some method of configuration-build using part-relationships from the repository.
e) Some support for semi-automatic testing by test-case generation or test-case tracking (in the repository)

Marking: 3 marks for a genuine tool, not a method.
3 marks each for 4 recognisably distinct features.

Part (b)
Candidates' defensible views are sought on impact of program generator tools on:
- the quality of product, e.g. better since generators auto-generates
- the speed of software development, e.g. faster since a generator tool integrates design and development
- the satisfaction of the end-user e.g. better since a generator tool either delivers faster or supports prototyping for better definition of requirements.

Marking: *5 marks* each area of change: the quality of product, the speed of software development, and the satisfaction of the end-user. With reasons. Or persuasive alternative with 3 distinct effects.

Examiner's Guidance Notes
Part (a) was well answered. Many candidates were familiar with software development toolsets such as Visual Basic, Relational Rose, or Visio. Some candidates asserted that applications such as MS Word or MS Access were software generators; such answers were not acceptable as these applications do not 'develop software' in the sense required by the curriculum definition.

Candidates should pay attention to the rubric of the question; if 15 marks are available, it is unlikely that an answer with only two features will access the full range of marks that can be awarded.

Part (b) tried to guide candidates into three distinct areas of benefits delivered by generator tools. Many candidates took the advice, but others made the required three statements but repeated themselves by telling the same result twice. For example, on the point of quality, two equivalent answers are 'the generator method produces many fewer errors' and 'the generator method is easy to test'.

Section B

Question 5

a) A serial file ‘datafile’ has a sequence of records R_1, R_2, …, R_N.
Draw a diagram showing how the records are laid out in the file, including any file pointers, (4 marks)
i) after the file has been opened but before any records are read from the file (4 marks)
ii) after the last record of the file has been read. (4 marks)

b) Write a program fragment which opens the file, countrs how many records are in it, the closed it. State which language you use. (4 marks)

Answer Pointers
(a)
A serial file ‘datafile’ has a sequence of records R_1, R_2, …, R_N.
Draw a diagram showing how the records are laid out in the file, including any file pointers, after

i) *the file has been opened but before any records are read from the file*.

R1	R2	R3	Rest of file	Rn	End File

First ^file pointer last record

File and associated pointer after executing the ‘reset’ instruction. Note that the file pointer indicates the start of the NEXT record on the file that is available.

ii) *the last record of the file has been read.*

After executing one “READ(datafile)” instruction, the file pointer moves down the file by one record.

R1	R2	R3	Rest of file	Rn	End File

^file pointer

End-of-file becomes true when the LAST record has been read. On most systems the actual end-of-file marker is not read over, although it used to be in COBOL programs. An error condition results if another READ operation is now attempted.

R1	R2	R3	Rest of file	Rn	End File

^file pointer

(b)
Write a program fragment which opens the file, counts how many records are in it, then closes it. State which language you use.

In Pascal-style,a solution is

```
OPEN datafile as #1
recordct = 0
WHILE NOT EOF #1 DO
        READ (datafile, onerec)
```

```
                increment recordct
            ENDWHILE
            CLOSE #1
```

Examiner's Guidance Notes
A bookwork question. Algorithms often did not match the data structure and pointer moves that the candidate had already described. This reveals an over-reliance on memory.

Question 6

a) Write a procedure *'convtime'* which takes as an input parameter a decimal time in hours (e.g.5.8765) and which returns the corresponding time as hours (integer), minutes (integer) and seconds (nearest integer). Thus input time 5.8765 IS 5 hours 52 minutes 35 seconds **(8 marks)**

b) Incorporate this into a program fragment which uses *'convtime'*. The program fragment provides appropriate parameters to the procedure *'convtime'* as input and the program fragment then prints out the returned answer. It must also incorporate a 'repeat' facility so that the conversion can be repeated a set number of times during one program run. **(4 marks)**

Answer Pointers
In Part (a), **(a)** the procedure should incorporate a 'repeat' facility so that 'ntimes' test times can be incorporated in a program run .

BASICV Implementation

```
DECLARE SUB convtime (intime, hrs, mins, secs)
PRINT "test harness for time conversion program"
PRINT "Dr Albert Einstein June 2003"
PRINT
PRINT "how many time tests"
INPUT ntimes
FOR H = 1 TO ntimes
PRINT "input time on 12-hour decimal time"
INPUT intime
CALL convtime(intime, hrs, mins, secs)
PRINT TAB(3); H; TAB(10);
PRINT intime; " IS "; TAB(35);
PRINT hrs; " hours "; TAB(45);
PRINT mins; " mins "; TAB(60);
PRINT secs; " seconds"
PRINT
NEXT H
PRINT
PRINT "program ends"
END
```

In Part (b), a solution similar to the following was sought:

```
SUB convtime (intime, hrs, mins, secs)
REM different ways of effecting this may be necessary
REM depending on just what functions are available in
REM your version of BASIC
```

```
PRINT "hours  minutes  seconds"
hrs = FIX(intime)
dec = intime - hrs
res = 60 * dec
 mins = FIX(res)
  res = res - mins
 secs = CINT(60 * res)
 END SUB
```

Examiner's Guidance Notes
An essential feature of a good answer is code that implements truncation and rounding, for example when cobverting raw numbers to structured time values.

Various implementations (from high-level logic to low-level detail) were acceptable. Implied conversion using type coercions are always suspect because they rely so much on the operation of a particular language system. This question is design to seek ability at design of algorithms that can run on a variety of implementations.

Question 7

7. The equation relating the time in minutes **(x)** from the hour 'H' where 1<=H <= 11 to when the hands overlap on the clock face is given by 30H + x/2 = 6x

a) Arrange this equation into a form suitable for calculating **x** for a given value of H. **(3 marks)**

b) Write code to return a series of these times when the hands overlap as a captioned table thus:
time(hours) minutes(as calculated) minutes seconds **(9marks)**

Answer Pointers
Part (a)

An equation explicit in calculating x from the series of values of H is derived thus:
given $30H + x/2 = 6x$ then $60H + x = 12x$
or $x = 60 * H / 11$

Part (b) An algorithm similar to what follows is sought:

```
Basic Program:
PRINT "clock hands time overlap program"
PRINT "Dr Albert Einstein June 2003"
PRINT "hours     time         minutes  seconds"
FOR H = 1 TO 11
PRINT TAB(3); H; TAB(10);
time = 60 * H / 11
PRINT time; TAB(25);
mins = FIX(time)
PRINT mins; TAB(35);
```

Examiner's Guidance Notes
This is a problem that can be simply expressed in an algebraic expression. The algorithm to implement the expression is straightforward.

Question 8

8. *a)* *i)* Show diagrammatically a linked list with one pointer. (2 marks)

ii) For what is this particular data structure useful? (2 marks)

iii) Why should sorting NOT be attempted with this structure? (2 marks)

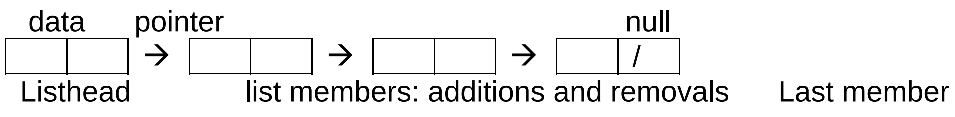
b) *i)* Likewise show diagrammatically an array with one subscript. (2 marks)

ii) What particular operation is expedited by this data structure? Show why it is particularly suitable for sorting operations. (2 marks)

iii) Under what circumstances is an array unsuitable for use? (2 marks)

Answer Pointers

(i) Shown diagrammatically a linked list with one pointer :



(ii) This particular data structure is useful for dynamic data storage. When data is received and deleted during a program run this structure optimises the concomitant use of memory.

(iii) Sorting involves moving the items about within the data structure. It is also convenient if a random access method is available for accessing any individual data item; this is not provided with a simple linked list. Hence a large number of pointer movements would be necessary to get a lengthy list of data into a different order than the one with which it was stored.

(b)

(i)

Item[1]	Item[2]		Item[i]	

Listhead list members: additions and removals Last member

Subscript in square bkts [i]

(ii)
The particular operation expedited by this data structure is Random access to any of the stored data items: an array member item[sub] is immediately available when 'sub' is known or calculable. So the items may be accessed in any programmable order, which is needed in sorting methods.

There are many algorithms for the sorting process, the one used should depend on particular attributes wanted like speed, minimum storage, minimum number of exchanged array members. Hardly ever can all of these be simultaneously maximised.

(iii)
A circumstance when an array unsuitable for sorting is when there is a large amount of data to be received yet the number of expected items is unknown. Some systems cope with this by providing dynamic array storage, when the maximum subscript does not have to be declared explicitly at compilation. With systems like Pascal, or Basic programs a large amount of

storage may have to be declared which is not subsequently used, or the program may fail at run-time if an attempt is made to store more data than was anticipated.

Examiner's Guidance Notes
The bookwork of Part (a) is quite straightforward. However, candidates did not present clear models of dynamic illustrations to show how the list changes when manipulated for insertion or deletion.

Question 9

a) Describe the operation of a queue data structure. Include a description with suitable diagrams of how a queue data structure is constructed and used. (6 marks)

b) Describe how ONE piece of *system software* uses a queue data structure as part of its functioning. (6 marks)

Answer Pointers
a) A queue is First-In-First-Out. Constructer creates data structure and initialises operations such as **head** and **tail** (null), length (zero). Usage is AddQ checks capacity (not **full**) , adds to **tail** and increments (or allocates dynamic storage for) **tail**. HeadQ either inspects or inspects-and-removes first in queue with appropriate check on capacity (not **empty**) and modification of **head.**

Marking: 2 marks for diagram, 2 marks for constructor description, 2 marks for use description.

b) System Software such as *processor scheduler* or *print scheduler* will keep a queue of tasks waiting service and may process it FIFO. (Other algorithms are possible but not interesting here.) Certain items of network management also employ queue data structures - this and other system software answers are acceptable.

Marking: 2 marks for a valid identification of System Software, 2 marks for brief description of purpose of the system software, 2 marks for the functionality that a queue endows.

Examiner's Guidance Notes
(a)

(i) A common error was putting queue members in the front/rear boxes when it should be the pointers/subscripts.

incorrect:

```
queue                  front      rear
  0   1   2   3   4
+---+---+---+---+---+    +---+      +---+
| 3 | 9 | 2 | 8 | 6 |    | 3 |      | 6 |
+---+---+---+---+---+    +---+      +---+
```

correct:

```
queue                  front      rear
  0   1   2   3   4
+---+---+---+---+---+    +---+      +---+
| 3 | 9 | 2 | 8 | 6 |    | 0 |      | 4 |
+---+---+---+---+---+    +---+      +---+
```

(ii) Also, pointers should be drawn TO the queue to illustrate how the data structure is implemented

```
correct                 incorrect

front           rear    arrive
  |               |       |
  V               V       V
+---+---+---+---+---+    +---+---+---+---+---+
| 3 | 9 | 2 | 8 | 6 |    | 3 | 9 | 2 | 8 | 6 |--> leave
+---+---+---+---+---+    +---+---+---+---+---+
```

(iii) Some answers omitted the two 'pointers' that are crucial to indicate a queue; the absence of these pointers means the diagram represents only a list, i.e. a diagram like the one below did not attract marks because it is a list, not a queue. The diagram uses just ONE pointer to illustrate the working of the data space. One pointer implies all the work (arriving and leaving) is done where the pointer is pointing. The data space is a list. In order to illustrate a queue, two pointers are needed, to show where arrivals occur, and separately where leavings occur.

```
         +---+---+    +---+---+   +---+---+
front -->| 3 | -----> | 6 | ----> | 9 | / |
         +---+---+    +---+---+   +---+---+
```

(iv) The algorithm to delete from the middle of a list is not appropriate as part of an answer.

(v) Answers with no diagrams but some textual description of a queue did not gain full marks because the question specifically asked for diagrams.

(vi) Many candidates suggested that the number of elements in the queue being odd/even is a primitive operation for a queue. Such a primitive operation makes no contribution to any description of a queue.

(vii) It was disappointing that some candidates were unable to spell ‘queue’ correctly in their answer; the word appears in the question.

(b)
(i)The question asked for a "system software" example so "airline reservations" was not acceptable.

(ii) "Computer memory with instruction pointer" does not provide an example of a queue as there is only one pointer.

(iii) Several answers contained 'round robin' scheduling which MAY be implemented as a queue so needs further explanation to earn full marks.

(iv) Among those candidates who gave a printer queue as an example there was some confusion over the pages to be printed in order from a single document (not acceptable - printer's responsibility) and print jobs from separate users which is the operating system responsibility.

(v) In the example of the operating system scheduling using a queue, care is needed with the words used. What is queued are jobs, tasks, not data, instructions

(vi) Subroutine return address store is not an example of a queue because it is a stack.

Question 10

a) Contrast and compare the methods of testing known as 'Black Box' and 'White Box'. Include the stages of development lifecycle when each method of testing is used. (6 marks)

b) State TWO situations when ONLY Black Box testing is employed. Give your reasons. (6 marks)

Answer Pointers

a) Black box is functional, testing inputs and outputs. White Box is procedural, seeking to trigger paths and flows to test internal logic. White box is used principally at code-and-unit-test stage. All other stages – system test, integration test, validation test, maintenance acceptance and regression test) use Black Box.

Marking: 2 for 'what' WB, 2 for 'what' BB, 1 mark for one instance of 'when' for each

b) Black Box is used as functional testing procedure for system test, integration test, validation test, maintenance acceptance and regression test) because the criteria for these tests are functional or behavioural descriptions that focus on data or control transformations – ins and outs.

Marking: 3 marks for identification of two situations within the development life cycle, further 3 marks for coherent rationale.

Examiner's Guidance Notes

a)

(i) "Black box" testing was sometimes confused with the "black box" flight recorder on aircraft.

(ii) No marks were available simply for giving the alternative names White -> glass, clear; Black -> closed, opaque. The question asked for '..compare the methods…'.

(iii) Many answers had a correct description of Black Box/White Box but not the stages of the lifecycle where they might be used, as asked for in the question.

(iv) It was surprising and disappointing to find answers which described Black Box/White Box fairly well but got them the wrong way round.

(v) Some candidates erroneously asserted that White Box testing is used to find *syntax* errors. The Compiler checks for correct syntax. White Box testing checks for correct implementation of designed intention, that means it check for errors of logic.

(b)

(i) The question asked for situations where ONLY Black Box would be used. Many answers spoke of when Black Box CAN be used , Black Box is OFTEN used, Black Box is MAINLY used...

(ii) The question anticipated an answer based on the stages of the Software Development Life Cycle, illustrating a candidate's knowledge of the activities in each stage of software development. An answer the described an activity that was not based on the development life cycle, such as "when commercial software is procured", is not acceptable.

(iii) Marks were awarded for part of the lifecycle where Black Box used and why, but not for a description of what that part of the lifecycle does.

Question 11
Describe three GUI features that you expect to find in the design of an interactive website for an e-commerce company that sells PCs and parts for PCs. Give your reasons. (12 marks)

Answer Pointers
What's wanted is some discrimination that picks up keywords 'interactive website' and 'e-commerce company that sells books' to decide the user is a B2C client who wants to pick a book. Keeping client 'in control' of the dialog is the aim of the GUI, achieved by such things as:

- drop-down lists - for spec of parts or price variation.
- Search engine – to produce lists of possibles and alternatives.
- Frames approach - to maintain navigation while offering details of PCs or parts
- Radio buttons – for selection of preferences such as disk sizes, CD-ROM speeds, payment option, postage class.
- And similar but distinct GUI features.

Marking: Three distinct features 4 marks each and pro-rata if unclear.

Examiner's Guidance Notes
(i) Some answers discussed windows, icons, menus without relating any of the discussion to the scenario described in the question. For example giving as an example of a dropdown menu File -> Open|Close|Save|Revert is discussing the operating system not an e-commerce website.

(ii) Answers that mentioned user-friendliness, security, help systems did not gain marks unless they actually described how these goals could be met in terms of user interface elements.

(iii) The question asked for THREE GUI features so it was a waste of effort to present 4 or 5.

(iv) Headings, colours, and fonts are common to all web sites and do not play only a minor part of the functionality expected in an INTERACTIVE website; answers that asserting these as GUI features attracted few marks.

Question 12
Briefly describe the operation of file organisations that support the following;

a) Dealing with telephone enquiries from customers about account details (6 marks)
b) Producing a print-out of all customers' invoices to be posted to them. (6 marks)

Use suitable diagrams to illustrate your answer.

Answer Pointers
a) Direct or Index Access Method to select specific records, plus sketches.

Marking: 2 marks for suitable file organisation, 2 marks for rationale, 2 marks for picture.

b) Sequential access, or Index-Sequential Access Method, to support batch processing by sequential read-out. Plus sketches.

Marking: 2 marks for suitable file organisation, 2 marks for rationale, 2 marks for picture

Examiner's Guidance Notes
(i) The question asked for diagrams of file organisations. One third of the marks were lost by candidates who made no attempt at drawing diagrams.

(ii) Candidates lost marks by not answering the question. Some candidates gave diagrams of the telephone operator's screen, some gave an algorithm to produce invoices, and some offered a detail description of databases.

(iii) Too many answers (often covering up to two pages) scored no marks because they did not mention file organisation anywhere.

(iv) Diagrams of the field structure in a database table are not the same as diagrams of file organisation. It is the organisation of whole records and their indexes that sums to an effective file organisation, not the simple layout of data within the records.

(v) Some answers managed to invert their answers and wrongly assign sequential access for (a) and direct access for (b). Such confusion gained few marks.