THE BCS PROFESSIONAL EXAMINATION
Certificate

October 2002

EXAMINERS' REPORT

Software Development

General Comments on Examination Technique

Candidates should always number their answers. Where this does not occur, the examiner has to guess at which question was being attempted and this can only disadvantage the candidate. A number of candidates sandwiched parts of answers between others with no indication that this has been done. All the pages in the answer book are numbered – it is preferable to say 'continued on page xx'.

Writing the answer to a question then writing it again gains no extra marks - it just wastes time. Lengthy crossings-out again wastes time. Candidates must learn how to write an acceptable answer without having to write it out **fully** in rough first – there simply is not enough time. It is often useful to jot down notes/key points as an aide mémoire prior to writing out the full answer but there is not time to write the answer in rough and then in a neater hand.

Writing out the question prior to answering just wastes time and never gains any marks.

Illegible handwriting is assumed to be wrong and gains no marks. The onus is on the candidate to show the examiner that they can answer the question, not for the examiner to guess what the candidate has written.

SECTION A

Answer TWO questions out of FOUR. All question carry equal marks.

The marks given in brackets are indicative of the weight given to each part of the question.

Question 1

1. The file '*studs*' contains details of students enrolled on courses in alphabetical order of student names. Each student record has the following items:

<identifier>	<student name>	<course code>	<options chosen>
8-digit integer	30 characters	4 characters	20 characters
92878112	BABBAGE Charles	PNC1	PRO1 HR1 MATH COMP

Another file, '*descr*', contains full descriptions of the courses and the corresponding course codes:

<course code>	<course description>
4 characters	30 characters
PNC1	National Dip/Computing year 1

a) **Specify a suitable data structure to contain the <student name> and a suitable data structure to contain one student record. Hence write a suitable description for the files '*stud's* and '*descr'.* (8 marks)**

b) **Write an initial algorithm to**

read the entire '*studs*' file
store each different unique course code encountered in the 2-dimensional array '*cscodes*'
count how many unique course codes are found
how many students are on each course and
the total number of student records on the file. (14 marks)

c) **Write code to sort the array '*cscodes*' into descending order of course code.**
(You do NOT need to deal with error conditions in any of the data items).
(8 marks)

This question was generally unpopular, suggesting that either candidates do not feel confident in tackling this sort of question in the time or that they are unprepared in this area of the syllabus. Usually if a candidate attempts this type of question it is done well.

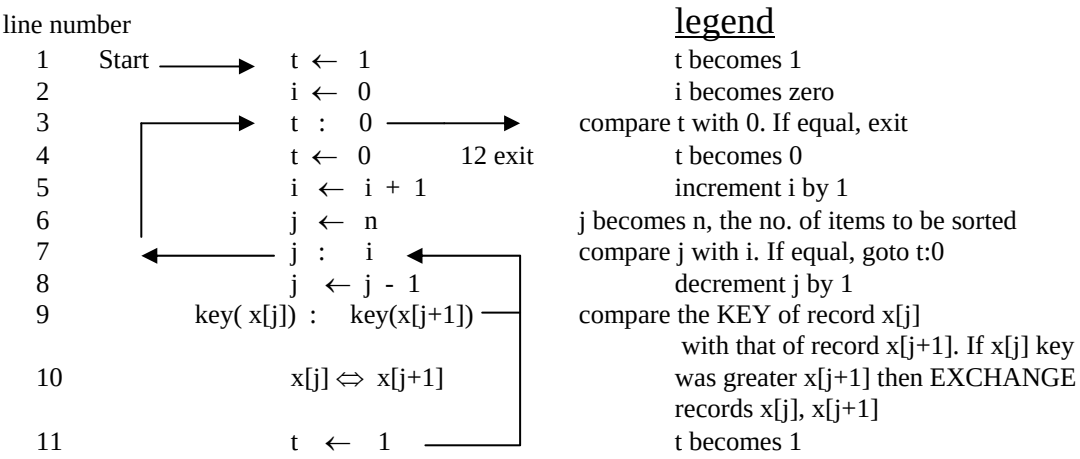
Answer Pointers

This was a straightforward algorithm development question. A simple step wise development approach was perfectly acceptable.

Candidates appeared to write the sort code from memory so often important parts were forgotten. If candidates just briefly checked their answers they would pick up the majority of errors and gain higher marks.

Question 2

2. The Iveson Algorithm for a sorting method is given below:



a) **Translate this algorithm <u>literally</u> into a procedural programming language. (Thus the transfers of control implied by → must be translated into 'GO TO label' statements.) State which language you are using. (10 marks)**

b) **Write a detailed criticism of the style of programming implied by the algorithm.**

(6 marks)

c) **Modify the algorithm and translate it into a modern, structured PROCEDURE *ivsort*(…) with meaningful variable names and without using labels. The PROCEDURE *ivsort*(…) requires appropriate parameters; that is, everything *ivsort* uses must be passed as parameters. (14 marks)**

This was a quite popular question but an unwise choice for those who could only answer part a) as they were guaranteed to lose two thirds of the marks.

Answer Pointers

a) A surprising number of candidates who translated the algorithm into Pascal did not use labels even though the question asked for GOTOs to be used. The symbol

⇔, which the question stated meant "exchange", needed translating into code – this was an essential part of the answer.

(b) It would appear that few candidates are taught to evaluate programming styles as many guesses and weird ideas were given. Very few made any comment about the use of GOTOs, for example. Some thought that a description of the algorithm was required and so wasted valuable time – this was already given as the 'legend' in the question.

(c) This part of the question was either omitted or done quite well. Many forgot to include procedure parameters despite this being specifically requested in the question. The examiner suspects that some candidates wrote out memorised code for the sorting method rather than developing the Iveson Algorithm.

Question 3

3. *a)* Describe the three common constructs of programming languages – sequence, selection and iteration. (10 marks)

***b)* In a programming language of your choice, show how code structures are built using these constructs. (10 marks)**

***c)* Show how these constructs can be used to build up data structures. (10 marks)**

A very popular question in which most students did very well in parts a) and b), but very few good answers to part c).

Answer Pointers

a) A basic understanding of sequence, selection and iteration was required. Sequence means 'do in this order'. Selection means 'do this or that'. Iteration means 'do this a certain number of times.

b) Examples in a Pascal-like code could be

begin .; .; . end	for a sequence.
if .. then .. else .. or case x of …..	for a selection
do .. until ..	for an iteration

c) Data structure examples could be

A record definition as a sequential collection of different fields.
A record definition with alternative field definitions as an example of a selection
A file definition as an example of an iteration (of records).

Question 4

4. *a)* Describe a software development method with which you are familiar and identify the type of programming language for which it is most suited. Give reasons for your choice of language type. (15 marks)

b) **For the method you have described in part *a)*, discuss the cost of getting the development wrong at each stage of the method. How does the method contribute to, or reduce, this cost at each stage? (15 marks)**

This was an 'open' question, seeking insight from each candidate about their own reflections of using a programming language under a management discipline (plan, or life cycle). Too many candidates forgot the 'refection' part and offered detail expositions of a life cycle that they knew. Various combinations were acceptable. Notes below are NOT expected to be learned, but to be observations from candidates in particular programming situations. Many candidates gave both example styles shown.

Answer Pointers

a) Example 1 - Waterfall with a procedural (3GL) language such as Pascal or C. Waterfall places great emphasis on completing each stage before continuing. In part, this is because of the low productivity of procedural programming. Design notations that identify modules and interfaces map onto programming languages with facilities for modules and interface definitions. Unit testing and system testing map on to white box 3GL logic tests and module integration black box map on to interface/functional testing.

Example 2 - DSDM (Dynamic Systems Development Method), or Prototyping, or Evolutionary prototyping, places great emphasis on delivering product and defining, often vague, user requirements. In particular, DSDM places more importance on delivery to a fixed time schedule rather than floating the schedule to accept late delivery of functionality. Programming languages suitable for this approach are high-productivity 4GL; MS ACCESS; MS ACCESS with Visual Basic; Visual C or C++ or Basic with CASE support, or O-O systems with CASE support.

Answers that simply described either life cycle or programming language were awarded few points. Any reflection was generously rewarded particularly reflections from application of skills and/or experience.

b) Example 1 - Many good answers described rework as the main reason for extra cost because Waterfall proceeded very slowly and rework proceeded just as slowly, and described how use of design (CASE) and programming tools to speed-up the stages, or use of testing tools to speed-up testing at each stage, were ways of reducing the cost of rework. Poor answers gave great detail on life cycle with cross-referencing to a suitable programming language.

Example 2 - Many good answers identified cost of implementing the wrong functionality but described how iteration using high-productivity programming systems quickly produced software to user satisfaction. Poor answers repeated pieces from a), asserting that 'doing it right' or 'doing all the tests' or 'doing it correctly' was the way to minimise costs.

SECTION B

Answer FIVE questions out of EIGHT. All questions carry equal marks.

The marks given in brackets are indicative of the weight given to each part of the question.

Question 5

5. *a)* What TWO conditions must be met for a recursive solution to a problem to be found? (6 marks)

_b)_ How do recursive solutions operate in practice? (3 marks)

_c)_ When is a recursive solution likely to be unworkable in practice? (3 marks)

A number of candidates clearly tried guesswork and produced remarks such as 'recursion is used to by-pass the real problem'. A number of candidates used memorised code of a recursive example but did not understood how it answered the question.

Answer Pointers

The best answers generally came from those candidates who gave a simple recursive routine and then used that as an example to explain their answer.

Question 6

6. If A is an approximation to the cube root of a real number N then A + C is a better approximation, where C is given by the formula

$C = [N/A^2 - A] / 3$

The initial value of (A) is A/3.

***a)* Develop a process for repeatedly evaluating the cube root of an input number until successive values differ by less than an input limit E.**

***b)* Incorporate it into a user-defined function 'Cubrt(…)' with appropriately-chosen parameters. State your chosen language. (12 marks)**

The inclusion of a simple formula in the question seemed to stop a lot of candidates attempting this question. Some candidates used INTEGER variables throughout – a serious error when values must be REAL numbers.

Answer Pointers

A function along the following lines was required

```
FUNCTION cubrt ( N : REAL;   E : REAL ) : REAL;
    VAR  A, C : REAL;
    BEGIN
       A := N / 3;                          { set first approximation }
       C := E + E;                          { set change of value, i.e. C, greater than E }
       WHILE ABS ( C ) > E  DO
          BEGIN
             C := ( N / (A*A ) - A ) / 3;   { calculate change }
             A := A + C                     { new cube root }
          END;
      cubrt := A
    END;
```

Question 7

7. *a)* **Define a linked list data structure to contain a name (20 characters), an integer telephone number and one pointer to the next list member.**
(2 marks)

b) **Two such linked lists have been set up (list1 and list2) each having their members in increasing telephone number order. Develop an algorithm to merge the lists, creating one new list which retains the members in increasing telephone number order. State your target language.**
(10 marks)

Quite popular though the majority only did part a) which was worth ONLY two marks. A number of candidates spent a significant amount of time on part a) even though it was worth just two marks - a classic case of poor examination technique.

Answer Pointers

Attempts fell into two distinct groups: namely, those who knew the work and scored [10,11,12] marks and those who knew only the linked list definition ad scored [0,1,2]. This was a simple merge algorithm.

Question 8

8. A publisher offers an author a fixed contract price A, to publish a novel by the author. The publisher expects to recoup this outlay from book sales. The publisher will retain a royalty R (typically being between 10% and 30%), being a percentage of the bookshop sale price P. The book selling price is typically between £3 and £15.

Write an interactive program which prints a table showing how many books have to be sold between these limits for the range of selling prices for the publisher to recover his investment. The necessary data (A) is requested interactively before printing the table. (12 marks)

Candidates appeared to be put off by the description of the problem even though it is a simple problem

Answer Pointers

If the book selling price is P, and the publisher retains a royalty R (expressed as a fraction), the publisher's return per book is (P * R). If the author's contract price is A, then the number of copies, C, needed to be sold for the publisher to recover their costs is C = A / (P * R)

If this formula is included within a loop of the range of selling prices, P, then the number of copies required for each price can be output. If this loop is included within a loop of the range of royalty payments, R, then the number of books for different royalty/prices can be output.

Question 9

9. *a)* Describe the operation of a stack. (6 marks)
b) Describe an IT application that needs to use a stack. Show how the application uses the stack. (6 marks)

Answer Pointers

a) Many good answers described the operations of push and pop, the general operation of Last In First Out (LIFO) and tests for underflow or overflow. Poor answers omitted one or more of these descriptors.

b) Good answers showed understanding of what a stack is for, e.g. parsing or translating – conversion to infix, postfix or other orders of expression or used by the Program Counter to store/push a subroutine address at a procedure call and later return/pop, or assembling a set of structured data by pushing each until some terminator then popping the lot somewhere safe. Poor answers that described stacks of plates in a restaurant, or other physical descriptions, but did not relate it to a computing environment did not score highly.

Question 10

10. What types of documentation should you give to a client on completion and hand over of a software project? Give reasons for the type of documentation specified.

(12 marks)

Good answers showed discrimination that 'documentation' does not always means 'everything from the life cycle'.

Answer Pointers

In the question, 'client' and 'handover' guided good answers towards a **User Guide** that described features and functions of the product, and **Maintenance documentation**, including sample test data and results, code listings and design documents. Poor answers stepped through a lifecycle and insisted that all aspects of design and software be handed over. Some answers focussed on ownership and legal guarantees which is important but not what this question was about.

Question 11

11. A graphical user-interface (GUI) is designed to meet the needs of the person using it. Describe THREE features that you expect to find in the GUI of an interactive website for a travel company. Give your reasons.

(12 marks)

Answer Pointers

Good answers linked the functionality of a travel agent with descriptions of how the GUI should be programmed; for example,

- Drop-down lists - for user control to choose required dates, or required location.
- Search engine – to produce lists of possible flights/locations and alternatives.
- Frames approach - to maintain navigation while offering travel, hotels, car hire
- Radio buttons – for selection of preferences such as airline or hotel.

Poor answers neglected the programming nature of this question and described what a good GUI **looked** like, or described the **aims** of a good GUI (to put the user in control) but without describing **how** this might be achieved.

Question 12

12. Describe the advantages and disadvantages of the following file organisations:

a) sequential only **(4 marks)**
b) index only **(4 marks)**
c) index-sequential hybrid **(4 marks)**

This question was poorly answered in the main. Many students did not know about batch processing that can employ simple sequential files very efficiently. Or that index

organisation is best suited to direct-access such as handling telephone or other one-off queries. Or that the hybrid index-sequential can perform equally well under batch and direct access conditions. In each case, increasing demands for index storage and index maintenance are not really 'negatives' but part of structuring data for more efficient processing. Some answers considered unlikely cases where an index might be wrong or absent

Answer Pointers

a) Sequential only – good answers described bulk runs; e.g. invoicing, payroll, mailshots. Update by making a fresh master copy, and the need to create the file initially in some order. Deletion is by omission at next master-create. Disadvantage is slow and unsuitable for individual record access because of need to search sequentially.

b) index only – very fast retrieval; e.g. real-time application such as customer telephone query, or billpoint (ATM) interaction. Needs a lot of maintenance if data population changes often. Must be kept in order. Must be kept disciplined, so that index-lookup is not confounded by variations in the data. For the overhead work of creating an indexed data structure we can get much increased interactive performance. Not much good at bulk runs, since the only access is by index look-up that would have to be repeated for every access, event 'sequential' ones. .

c) Index-sequential – hybrid organisation combining best of both above. Data records are laid out sequentially, for ease of bulk processing, but also there is an index to each record to speed up interactive processing. File updates require index updates as well, so file updating is more costly. Again, the apparent disadvantage of structuring the data delivers rich performance rewards for both bulk runs and interactive queries.