THE BCS PROFESSIONAL EXAMINATION

BCS Level 4 Certificate in IT

September 2018

EXAMINERS' REPORT

SOFTWARE DEVELOPMENT

General comments on candidates' performance

The distribution of results was somewhat bimodal with a number of good submissions offset by a number of poor submissions. The questions requiring candidates to write sections of code were the least popular.

Please note that the answer pointers contained in this report are examples only. Full marks were given for alternative valid answers.

SECTION A

(Candidates were required to answer **TWO** out of the four questions set)

A1

a) Assume two teams A and B compete against each other and each has a score at the end. By comparing the scores, it can be decided if team A won, or team B won, or it was a tie. Write a function winner(A,B) which returns 1 if the score A is larger than score B, return 2 if score B is larger and returns 0 if they are the same.

(8 marks)

b) Write a function sum(V,N) which returns the total of the first N numbers in the array V.

(8 marks)

c) Now suppose that the monthly sales figures (£, GBP) for two shops A and B are available for the past year in arrays salesA and salesB.

	0	1	2	3	4	5	6	7	8	9	10	11
salesA (£)	22	44	55									
salesB (£)	99	39	55									

A simple method of calculating the winner would just be to sum the sales figures for the year and compare the totals for the two shops. How would you code this calculation to announce the winner?

(6 marks)

d) However, the owner of the shops has decided on a more elaborate scoring system. For each month the shop with the most sales is awarded 4 points and the other shop 2 points, but if there is a tie, they are each awarded 3 points. Using the function created in a) write a function called scoresAB to calculate the scores for each shop for each month and store them in arrays scoresA, scoresB. Using the function created in b) total the scores for each shop and announce the winner for the year.

(8 marks)

Answer pointers

a)

```
#include "stdio.h"
int winner(int A, int B){
	if(A>B)
		return 1;
	else if(B>A)
		return 2;
	else
		return 0;
}
int main(){
	printf("winner(22,99) is %d\n",winner(22,99));
}
```

b)

```
#include "stdio.h"
int V[12]={0,1,2,3,4,5,6,7,8,9,10,11};
int sum(int* V,int N){
	int i,t;
	t=0;
	for(i=0;i<N;i++)
		t+=V[i];
	return t;
}
int main(){
	printf("sum(V,5) is %d\n",sum(V,5));
}
```

c)

```
#include "stdio.h"
int winner(int A, int B){
	if(A>B)
		return 1;
	else if(B>A)
		return 2;
	else
		return 0;
}
int sum(int* V,int N){
	int i,t;
	t=0;
	for(i=0;i<N;i++)
		t+=V[i];
```

```
	return t;
}
int salesA[12]={22,44,55,3,4,5,6,7,8,9,10,11};
int salesB[12]={99,39,55,3,4,5,6,7,8,9,10,11};
int main(){
	int totalA,totalB;
	totalA=sum(salesA,12);
	totalB=sum(salesB,12);
	printf("shop A total sales %d\n",totalA);
	printf("shop B total sales %d\n",totalB);
	if(totalA>totalB)
		printf("winner is shop A\n");
	else
		printf("winner is shop B\n");
}
```

d)

```
#include "stdio.h"
int winner(int A, int B){
	if(A>B)
		return 1;
	else if(B>A)
		return 2;
	else
		return 0;
}
int sum(int* V,int N){
	int i,t;
	t=0;
	for(i=0;i<N;i++)
		t+=V[i];
	return t;
}
int salesA[12]={22,44,55,3,4,5,6,7,8,9,10,11};
int salesB[12]={99,39,55,3,4,5,6,7,8,9,10,11};
int scoresA[12];
int scoresB[12];
int main(){
	int i,w;
	for(i=0;i<12;i++){
		w=winner(salesA[i],salesB[i]);
		switch(w){
			case 1:scoresA[i]=4;scoresB[i]=2;break;
			case 2:scoresA[i]=2;scoresB[i]=4;break;
			case 0:scoresA[i]=3;scoresB[i]=3;break;
		}
```

```
        }
        int totalA,totalB;
        totalA=sum(scoresA,12);
        totalB=sum(scoresB,12);
        printf("shop A total score %d\n",totalA);
        printf("shop B total score %d\n",totalB);
        if(totalA>totalB)
                printf("winner is shop A\n");
        else
                printf("winner is shop B\n");
}
```

Examiners' Comments

An unpopular question with a low average mark. There is evidence that most candidates made a good job of part a) but as the requirements became more complicated many of the answers became unstructured.

A2 A teacher requires a pass list for a module with some unusual rules. The Coursework is worth 40% of the Overall mark and the Exam is worth 60%. In order to pass the module the student must have one mark (either the Coursework or the Exam) at least 40% and the other mark must be minimum performance (mark at least 30%) [If it helps, it can be assumed that there are 100 students in the class.]

	0	1	2	3	4	...	98	99
Student Id	2017001	2017002	2017003	2017004	2017005	...	…	…
Coursework (%)	20	49	51	70	10	…	…	…
Exam (%)	33	50	69	0	80	…	…	…

Write a program to:

a) produce a list of all the students who have passed. For each student list their Student Id, Coursework mark, Exam mark and Overall mark.

(12 marks)

b) produce a separate list of Student Ids for those students who failed.

(12 marks)

c) report the Student Id of the student who should get the module prize (the student with the highest overall mark).

(6 marks)

[The most marks will be awarded to answers which make good use of functions/subroutines].

Answer pointers

```
#include "stdio.h"
//These 5's should really be 100
int StudentId[5]={2018001,2018002,2018003,2018004,2018005};
int Coursework[5]={20,49,51,70,10};
int Exam[5]={33,50,69,0,80};
```

```
int pass(C,E){
	return (C>=40 && E>=30) || (C>=30 && E>=40);
}
void passlist(){
	int i;
	float mark;
	printf("Pass List\n");
	for(i=0;i<5;i++){
		if(pass(Coursework[i],Exam[i])){
			mark=(Coursework[i]*40 + Exam[i]*60)/100;
			//% cues an insert, %% means print a single %
			printf("Id %d Cwk %d%% Exm %d%% Mark
%5.2f%%\n",StudentId[i],Coursework[i],Exam[i],mark);
		}
	}
}
void faillist(){
	int i;
	printf("Fail List\n");
	for(i=0;i<5;i++){
		if(!pass(Coursework[i],Exam[i])){
			printf("Id %d\n",StudentId[i]);
		}
	}
}
void prize(){
	float mark,max=0;
	int i,prizeId;
	printf("Prize\n");
	for(i=0;i<5;i++){
		mark=(Coursework[i]*40 + Exam[i]*60)/100;
		if(mark>max){
			max=mark;
			prizeId=StudentId[i];
		}
	}
	printf("Prize should be awarded to Id %d\n",prizeId);
}
int main(){
	passlist();
	printf("\n");
	faillist();
	printf("\n");
	prize();
}
```

Examiners' Comments

Another unpopular question. The evidence shows there were some poor answers with some candidates presenting the results that were asked for rather than the algorithm for obtaining them. There were also some good answers but clearly candidates found this a difficult question.

A3 The array **x** has been initialised as follows

index	0	1	2	3	4	5	6	7	8	9	10	11
x	1	9	5	0	8	6	3	5	1	0	2	9

The subroutine ***a*** in the code below is going to be executed with parameter ***b*** set to 10 and parameter ***c*** set to 6.

a) Trace the call of the function ***a(10,6)*** and show clearly the results of the call.

(8 marks)

	function
1	int a(int b, int c){
2	int d,e;
3	e = -1;
4	d = 0;
5	while(d < b){
6	if(x[d] == c)
7	e = d;
8	d++;
9	}
10	return(e);
11	}

b) Write a brief summary of what the subroutine does.

(6 marks)

c) Decide on better names for the identifiers (the subroutine name, its parameters and the variables) and rewrite the code using your new names and including suitable comments.

(10 marks)

d) Rewrite lines 5 to 9 using a for-loop instead of a while-loop.

(6 marks)

Answer pointers

```
(a)

a(10,6)
line 1:       b=10, c=6
line 3:       e = -1
line 4:       d = 0
line 5:       d<b => 0<10
line 7:       x[d]==c  =>  x[0]==6  =>  1==6  =>  false
line 5:       d<b => 0<10
line 6,8:     x[d]==c  =>  x[0]==6  =>  1==6  =>  false,  d++  =>  d=1
line 5:       d<b => 1<10
line 6,8:     x[d]==c  =>  x[1]==6  =>  9==6  =>  false,  d++  =>  d=2
```

```
line 5:       d<b => 2<10
line 6,8:     x[d]==c  =>  x[2]==6  =>  5==6  =>  false,  d++  =>  d=3
line 5:       d<b => 3<10
line 6,8:     x[d]==c  =>  x[3]==6  =>  0==6  =>  false,  d++  =>  d=4
line 5:       d<b => 4<10
line 6,8:     x[d]==c  =>  x[4]==6  =>  8==6  =>  false,  d++  =>  d=5
line 5:       d<b => 5<10
line 6:             x[d]==c  =>  x[5]==6  =>  6==6  =>  true,
line 7:             e=d  =>  e=5
line 8:             d++  =>  d=6
line 5:       d<b => 6<10
line 6,8:     x[d]==c  =>  x[6]==6  =>  3==6  =>  false,  d++  =>  d=7
line 5:       d<b => 7<10
line 6,8:     x[d]==c  =>  x[7]==6  =>  5==6  =>  false,  d++  =>  d=8
line 5:       d<b => 8<10
line 6,8:     x[d]==c  =>  x[8]==6  =>  1==6  =>  false,  d++  =>  d=9
line 5:       d<b => 9<10
line 6,8:     x[d]==c  =>  x[9]==6  =>  0==6  =>  false,  d++  =>  d=10
line 5:       d<b => 10<10 false
line 10 return (e)  => return (5)
```

(b) function searches array elements

from 0 to b
for the occurrence of c
and returns the index where c was found

3*2=6 marks

(c) new names

a => search or findPos or ...
b => high
c => target
d => index
e => location

```
int search(int low, int target){
    int index, location;
    location = -1;
    index = 0;
    while(index < high){ /* step through array from index 0 to high */
        if( x[index] == target ) /* if target found */
            location = index; /* record index */
        index++;
    }
    return( location )
}
```

```
names 4 marks,      comments 6 marks


(d)


for(index=0; index < high; index++) {
        if( x[index] == target )
            location = index;
}

/* with original identifiers*/
for(d=0; d < b; d++) {
        if( x[d] == c )
            e = d;
}
```

Examiners' Comments

This was a very popular question. The evidence shows that a number of candidates were able to trace the value of variables through execution of the function but were then unable to explain what the function did.

A4 a) Consider the code below and format it in a more familiar human-readable form.

```
char compare(int p1,int p2){if(p1==p2)return('=');
else if(p1>p2)return('>');else return('<');}
```

(6 marks)

b) Referring to the code in part a), find and write out the following

i) all the different identifiers
ii) all the different constants
iii) all the different operators
iv) a conditional (logical, boolean) expression
v) a conditional statement

[*Note that you should copy out exactly what is requested and no more*]

(5 x 2 marks)

c) If, by mistake, the programmer wrote (p1=p2) instead of (p1==p2) how would that change the behaviour of the code?

(8 marks)

d) Consider the following 3 statements

```
v = 2 ;
v = '2' ;
v = two ;
```

They could all be read out loud by saying "v is assigned two", but they are all different. Briefly describe the differences.

(6 marks)

Answer pointers

a)

```
char compare(int p1,int p2){
    if( p1 == p2 )
        return( '=' );
    else if( p1 > p2 )
        return('>');
    else
        return('<');
}
```

b)

i) all the different identifiers: compare, p1, p2

ii) all the different constants: '=', '>', '<'

iii) all the different operators: ==, >

iv) a conditional (logical, boolean) expression: p1==p2

v) a conditional statement: if(p1>p2)return('>');else return('<');

c) Following the mistake p1=p2 is an assignment so p1 would now be given the value of p2.

The overall value of an assignment is the value that was assigned i.e. p2

If a boolean is required and an integer is found then 0 is treated as false and all other numbers as true

So the first "if" will be false if p2 is 0, true otherwise and does not depend in any way on the initial value of p1

d)

```
v = 2 ;
v = '2' ;
v = two ;
```

The first one assigns the integer value of 2 [one of the integer numbers from -maxint to +maxint]

The second one assigns the character value of 2 [one of the ASCII character set, including upper and lower case letters]

The third one looks up the current value of the variable named two and copies this value (whatever type it is) into v

Examiners' Comments

An extremely popular question. Many candidates made a very good job of this question and many answered it first, suggesting that it stood out as the one with which most candidates felt confident with.

SECTION B

(Candidates were required to answer **FIVE** out of the twelve questions set)

B5

Conversion of a decimal number to its binary equivalent is carried out using the following technique:
The decimal number is successively divided by 2 and the remainders are stored.
The remainders in reverse order form the binary number; the last remainder being the most significant digit. In the example below 19 (decimal) = 10011 (binary)

Process	Quotient	Remainder
÷2	19	1 (least significant bit)
÷2	9	1
÷2	4	0
÷2	2	0
÷2	1	1 (most significant bit)
	0	

Write pseudocode or a program in a language of your choice in which a decimal number is input and its binary equivalent is output. Only positive integer decimal numbers are to be considered.

(12 marks)

Answer pointers

(Pseudocode o program code accepted)

```
PROGRAM Decimal2Binary
        Variable Declarations
        INTEGER DecimalNumber, Quotient, i, j
        INTEGER ARRAY BinaryNumber
        BEGIN
                OUTPUT ("Enter any positive decimal number: ")
                INPUT "DecimalNumber"

                Quotient ← DecimalNumber
                i ← 1

                WHILE (Quotient != 0) DO                    {Loop while Quotient not zero}
                        BinaryNumber[i] ← Quotient MOD 2    {Calculates Remainder}
                        Quotient ← Quotient / 2
                        i++
                ENDWHILE

                {LSB calculated first, so BinaryNumber needs to be printed out in reverse order}
                PRINT ("Equivalent binary value of decimal number:" DecimalNumber)

                FOR (J=i-1, j>0, j--)
                        PRINT BinaryNumber[j]
                ENDFOR
        END
```

Examiners' Comments

This was the least attempted question in Section B. The evidence shows that the overall performance on this question was poor with around a half of the candidates achieving satisfactory pass mark.

A number of candidates produced a correct or near correct solution. An equal balance of pseudocode, C++ code was evident in candidates' answers.

The most common problems seem to stem from coding errors involving iteration (WHILE loop) and inability to translate an algorithm into code particularly the use of the MOD operator to return the remainder of integer division.

B6. Define the following terms:

a) library programs **(3 marks)**
b) truncation **(3 marks)**
c) encryption **(3 marks)**
d) pointer **(3 marks)**

Answer Pointers

a) library programs are pre-written standard programs that are available for immediate use; they are stored in compiled format and can be included by the programmer within one or more programs. Library functions are widely used to assist the programmer in interacting with peripherals such as printers.
b) truncation limits the number of digits to the right of the decimal point. Many programming languages have built-in functions to truncate data.
c) encryption is used to make stored or transmitted data more secure, since the data is unreadable to people who do not have the key to decode it.
d) Pointer is the address or reference of a data element which allows it to be retrieved without further searching.

Examiners' Comments

The evidence shows that overall performance on this question was slightly lower than expected with around 60% achieving a satisfactory pass mark. There were a number of problems that were evidenced mainly by candidates misinterpreting the following terms:

- *Library programs: some candidates assumed this meant a real library with a repository of books.*
- *Truncation: a small number of candidates thought this term related to truncation of code.*
- *Pointer: About a 15-20% of candidates interpreted this as a screen pointer (cursor) on a graphical user interface A few candidates considered issues such as the use of pointers in indexes and in stacks and queues in excessive detail.*

B7. Search and Sort operations are commonly used on data.

a) Explain the difference between searching and sorting in computers **(2 marks)**

b) Describe the operation of the binary chop search in a sorted array, assuming the target value is in the array. The answer can be described in words, or by pseudocode or by actual program code.

(8 marks)

c) Explain why a binary chop search is more efficient than a linear or sequential search.
(2 marks)

Answer pointers

a) A search algorithm is used to find an item with specific properties amongst a group of items, whereas a sort algorithm is used to put elements from a list into a specific order.

b) Binary Chop Algorithm / Pseudocode

```
Binary-Search (A, Target):      //A is sorted array
                                //Target is value being searched for
Variable Declarations
    Left ← 1                    // Set up initial conditions
    Right ← Size[A]             //Right assigned to maximum element in array
    WHILE (Left <= Right) DO
    BEGIN
       Middle ← (Left + Right) / 2
       IF (A[Middle] = Target) THEN
             PRINT "Target Value = " Target "Found at A[" Middle "]"
       ELSE IF (A[Middle] < Target) THEN
             Left ← Middle + 1
       ELSE
             Right ← Middle - 1
       END IF
    END WHILE LOOP
```

Word descriptions also accepted, such as:

1. Array information must be ordered according to some key
2. Envisage the keys in ascending order in an array
3. To find key k look at the value stored halfway down the array, compare with k
4. If equal to k, search completed so stop
5. If k is smaller throw away top half of array, otherwise throw away bottom half
6. Repeat from step 3 until search successful or only one key left

c) In a linear or sequential search, each element in the list is examined until the target value is found; this could take considerable time for a large array. In a binary search the number of elements being examined is halved for each iteration of the program; for example, a maximum of only 6 comparisons are needed to find a target value in a list of 64 elements.

Examiners' Comments

This was the second least popular question attempted in Section B.

There is evidence that the overall performance on this question was good and seems to be a favourite topic amongst a number of candidates with 64% candidates achieving a satisfactory pass mark. Several candidates produced a correct or near correct solution. An equal balance of a written description and code pseudocode/C++ was evident in candidates' answers to part b).

There were not any specific issues to report on any part of this question apart from the occasional delivery of a sorting algorithm instead of a searching algorithm.

a) Explain the term machine code **(2 marks)**

b) Describe the process of translating the assembly language code using an assembler **(6 marks)**

c) Describe the advantages and disadvantages of writing programs in assembly language **(4 marks)**

Answer Pointers

a) Machine code is the set of binary instructions that are used by the CPU to perform a task, the machine code created is processor dependent.

b) The source code is the language instructions that have been written by the computer programmer, which the computer cannot execute directly. In this case, the source code is written in assembly language, which is a series of memorable mnemonics used to represent machine operational codes. The assembler runs through each of the instructions and translates this source code into machine code so that the computer can run the program.

c) An advantage of using assembly language is that it is an efficient low-level language that can be translated quickly as it has a one-to-one relationship with machine code. Assembly code is complex so it is a difficult language to program and needs a high degree of expertise to write and debug it.

Examiners' Comments

64% of candidates attempting this question achieved a satisfactory pass mark.

Breaking the question down to its three parts the evidence shows that: - Part a) Machine Code was overall well understood. Part b) This part caused the most issues with some candidates showing a lack of understanding of the role of an assembler in the context of the question. Part c) was generally well answered despite a lack of knowledge of assembler code as most candidates could associate assembly code mnemonics with machine code.

B9. Documentation is an important part of the process of software development. Write BRIEF notes on documentation to answer the following questions:

a) Why is documentation important?
b) Who is it for?
c) What does it consist of?
d) How is it produced?
e) When is it produced?
f) What problems can be experienced with documentation?

(6 x 2 marks = 12 marks)

Answer pointers

a) Why is documentation important? – The team who wrote software are not necessarily the same team that will maintain or modify the software.

b) Who is it for? – Documentation is for the users and follow-on programmers

c) What does it consist of? - Separate documents, comments in code

d) How is it produced? - By hand sometimes, automatically sometimes

e) When is it produced? - As part of the process it describes/records

f) What are the problems with it? – Documentation is often inadequate or out-of-date

Examiners' Comments

This was question was attempted by 86% of candidates. Overall performance on this question was good with 72% achieving a satisfactory pass mark.

There is evidence that a relatively large number of candidates produced correct or near correct solutions. There was sometimes a need to qualify some of the short answers that were produced with more descriptive comments. Conversely some candidates produced overly long answers that added little value. Candidates should be aware that both these extremes usually result in lost marks.

B10. Write BRIEF notes to compare and contrast the following pairs of terms.

a) data structure and program structure **(4 marks)**
b) black box test and white box test **(4 marks)**
c) data validation and data verification **(4 marks)**

Answer Pointers

a) **Data structure** - the recognition of sequencing (record), selection (variant, union) and repetition (array, file) within data - and recognising primitive data types (i.e. not made up of other types) e.g. integer, character, Boolean, ...). **Program structure** - the recognition of sequencing (;), selection (if) and repetition (for, while) in programs.

b) **Black box test** is testing a subroutine via its interface with no knowledge of its inner workings (code). **White box test** is testing a subroutine with full knowledge of its code.

c) **Data validation** is used to check the input data conforms with the data requirements of the system and to avoid data errors. **Data verification** is used to check that the user enters or inputs the data without making a mistake; can be checked visually or by double entry.

Examiners' Comments

This was the most popular question attempted in Section B, however the evidence shows that the overall performance on this question was relatively poor given its popularity.

Part a) was answered poorly overall with a low understanding of these terms.

Part b) was well understood and this where many candidates gained full marks.

Part c) was poorly answered with quite a few candidates misinterpreting data validation and verification as being associated with whether a product (not the data) was fit for purpose. Many candidates could not adequately distinguish these two related terms.

B11.

a) What do you understand by the term 'debugging'? **(4 marks)**

b) In a simple programming environment where the programmer has only the standard output facilities of the programming language to use, how is debugging approached? **(4 marks)**

c) What extra facilities to assist in debugging might be provided in a more extensive development environment? **(4 marks)**

Answer pointers

a) The program does not respond in the way that was expected and the programmer is looking for more diagnostic information to discover the source of the problem. Debugging includes the identification and correction of errors.

b) The programmer can insert extra output statements in key places in the code. It is usual to output a distinctive label to show that this place has been reached at run-time and often it is useful to output the value of some key variables. Key places in the code are just inside loops (to find out how many times they are repeating) and on the separate branches of conditionals (to understand which branch was taken)

c) In a more sophisticated development environment the programmer might expect to be able to set 'breakpoints' where the program can be temporarily suspended and inspections made of the values of variables for example.

Examiners' Comments

Overall performance on this question was the poorest in Section B, with only a few candidates producing a correct or near correct solution. The evidence shows that:

Part a) was generally well understood although many candidates produced answers that only related to testing.

Part b) was generally poorly answered with very little depth to many answers.

Part c) produced many low mark answers with most candidates not being aware of the major techniques found in open source debugging tools and environments such as Eclipse Netbeans and Visual Studio for example.

Overall there is evidence to show that many candidates do not focus on the practical issues concerned with this question – that of debugging and as a result could only produce superficial sometimes textbook answers.

B12 One particular software development method is named the waterfall method.

a) Write out the names of all the phases in the method. **(3 marks)**

b) Choose THREE phases and write a brief description of each of the three phases chosen. **(9 marks)**

Answer pointers

a) According to some the waterfall method has 5 Phases: Requirements, Specification, Design, Implementation and Testing.
According to others the waterfall method has 7 Phases: Feasibility, Analysis/Requirement, Design, Coding, Implementation Testing and Maintenance/Review.

b) Accept any valid description of a phase of the waterfall method; some examples given below:

Requirements: gathering information about the behaviour of the system.
Specification: writing down the required behaviour and getting agreement of customer.
Design: choosing effective data structures and algorithms.

Implementation: writing final code solution.
Testing: choosing a deliberate testing strategy, for example, by module, white-box, black-box, unit testing, system testing.

Examiners' Comments

Overall performance on this question was the second best in Section B reflecting a good understanding of the Software Development Life Cycle(SDLC).

There is evidence that a relatively large number of candidates produced correct or near correct solutions. It was generally acceptable to mix the listing of phases up though candidates lost marks if the order was incorrectly specified. Some candidates included other techniques in the development process such as prototyping and lost marks as a result. The depth of discussing each phase in turn was quite variable; the best attempts were from candidates who could explain how each phase was delivered using familiar tools and techniques.