BCS THE CHARTERED INSTITUTE FOR IT

BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 4 Certificate in IT

SOFTWARE DEVELOPMENT

Examiner Report

September 2015

General comments on candidates' performance

The standard for this examination was reasonable, with many candidates gaining high marks.

However, although there were some excellent papers, a large number of candidates lost marks by providing answers that contained minimal content or were unrelated to the question. Candidates are therefore advised to spend more time reading and understanding the requirement of each question before writing their answers. In addition, many candidates answered more than the required number of questions. Future candidates should note that no credit is given for answering additional questions.

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

Answer Pointers

a)

```
#include<stdio.h>
int A[100];
int main(){
      int i;
      int max=A[0];
      for(i=1;i<100;i++){
            if(A[i]>max)
                  max=A[i];
      }
      printf("Maximum value in array A is %d",max);
}
```

b)

```
int A[100];
int max(){
      int i;
      int max=A[0];
      for(i=1;i<100;i++){
            if(A[i]>max)
```

```
                max=A[i];
      }
      return(max);
}
int min(){
      int i;
      int min=A[0];
      for(i=1;i<100;i++){
            if(A[i]<min)
                  min=A[i];
      }
      return(min);
}
```

c)

```
#include<stdio.h>
int A[100];
int max(){
      int i;
      int max=A[0];
      for(i=1;i<100;i++){
            if(A[i]>max)
                  max=A[i];
      }
      return(max);
}
int min(){
      int i;
      int min=A[0];
      for(i=1;i<100;i++){
            if(A[i]<min)
                  min=A[i];
      }
      return(min);
}
int sumAll(){
      int t=0;
      int i;
      for(i=1;i<100;i++){
            t += A[i];
      }
      return t;
}
int sumWithout(int low, int high){
      int t=0;
      int i;
      for(i=1;i<100;i++){
            if(A[i]!=low && A[i]!=high)
                  t += A[i];
      }
      return t;
}
int main(){
      int i;
      int sum=sumAll(), sumNoOut=sumWithout(min(),max());
      float ave1=sum/100;
      printf("Average value in array A is %.2f",ave1);
```

```
	float ave2=sumNoOut/100;
	printf("Average value in array A (without outliers) is %.2f",ave2);
}
```

[Note: In part c) answers which made good use of functions obtained the most marks]

Examiners' Guidance Notes

Although this was not a popular answer in section A, many of the candidates that attempted the question produced a correct or near correct solution for part a) finding a maximum number in an array and for part b) creating functions to find the minimum and maximum number in an array.

In part c) several candidates used functions to create all the elements needed for a correct solution. There is evidence that some candidates lost marks by misunderstanding or omitting the requirement to determine the average temperature without including the occurrence of the maximum and minimum temperatures in the array.

A2

Answer Pointers

```
#include<stdio.h>
char SECRET[6];
char REVEALED[6];
int main(){
	int found=0;
	int guesses=0;
	char guess;
	int i;
	for(i=0;i<6;i++){
		REVEALED[i]='@';
	}
	printf("Player 1: Please input the SCECRET 6 letter word (lower case)");
	scanf("%s",SECRET);
	while(!found && guesses<12){
		printf("Player 2: Please input a 1 letter guess (lower case)");
		scanf("%c",&guess);
		guesses++;
		for(i=0;i<6;i++){
			if(guess==SECRET[i]){
				REVEALED[i]=guess;
			}
		}
		int atCount=0;
		for(i=0;i<6;i++){
			if(REVEALED[i]=='@'){
				atCount++;
			}
		}
		if(atCount==0){
			found=1;
		}
	}
	if(found){
		printf("Player 2 wins in %d guesses",guesses);
	}else{
		printf("Player 1 wins (not found in 12 guesses)");
	}
}
```

Examiners' Guidance Notes

This was the least popular questions in Section A. In a few cases the candidate produced a near correct structured solution based on entering a secret word by player 1 and allowing player 2 to have 12 guesses at the secret word with the letters guessed correctly displayed; finally, the winner was determined
However, few marks were gained by many candidates as the quality of their answers was generally poor; for example, the evidence shows that many candidates got no further than simply entering a secret word or even using the example (ANIMAL) provided in the question followed by player 2 taking one guess at this word.

A3

Answer Pointers
a) The trace of the call of W(5,6) would be

	0	1	2	3	4	5		X	Y	Z		V[Z]==X	V[Z+1]==Y	||
V(initial)	4	4	5	5	6	6		5	6					
										0		false	false	false
										1		false	false	false
										2		true	false	true
										3		true	true	true
										4		false	true	true
	n	n	y	y	y	6								

b)
The change required is to substitute && for ||

```
if(V[Z]==X && V[Z+1]==Y) V[Z]='y'; else V[Z]='n';
```

c)

The change required is to substitute >=, <= for ==

```
if(V[Z]>=X && V[Z+1]<=Y) V[Z]='y'; else V[Z]='n';
```

d)
Better names

```
Function  W    inRange
Parameter X    low
Parameter Y    high
Local var Z    index
```

Examiners' Guidance Notes
This was a very popular question with many candidates achieving high marks in part a) where they attempted to trace through the code provided. In many cases all stages of the trace were completed and the correct answer provided.

The majority of candidates tabulated the trace which made it easy to follow, whilst others copied out and evaluated the code for each run through the loop. In some cases candidates simply wrote down a solution to the trace without showing how they arrived at it, consequently marks could not be awarded for method and technique if they produced an incorrect solution.

Many candidates gained full marks for part d) where they chose meaningful or better names than W, X, Y and Z for the code outlined in part c).

A4

Answer Pointers

a)

```
char* x(char y){
      char* z;
      if('0'<=y&&y<'8')
            z="octal";
      else
            z="error";
      return(z)
}
```

b)

Difference between 10 and "10" - 10 is an integer number and "10" is a string of length 2 characters
Difference between 10 and 10.0 - 10 is an integer number and 10.0 is a floating point or fixed number

c)

identifiers	x y z
constants	'0' '8' "octal" "error"
type of parameter	char
type of result	char* (=string)
conditional expression	'0'<=y
assignment statement	z="octal";

d)

Turned round 'if' using ! (extra parentheses necessary)

```
      if(!('0'<=y&&y<'8')) z="error"; else z="octal";
```

Not using !

```
      if('0'>y || y>='8') z="error"; else z="octal";
```

equivalently…

```
      if(y<'0' || y>='8') z="error"; else z="octal";
```

Examiners' Guidance Notes

This was a very popular question that was attempted by 94% of the candidates; it was generally well answered with approximately 66% of the candidates achieving a satisfactory pass mark.

Most of the candidates were able to gain maximum or near maximum marks for part a) and generally all showed a reasonable understanding of part b).

Marks were mainly lost in part c) where many candidates found it difficult to identify the types of function parameter and results from the example code. Also, there is evidence that many candidates misunderstood "turning around the if-statement" in part d) and the use of the not (!) operator to achieve it.

SECTION B
(Candidates were required to answer FIVE out of the eight questions set)

B5

Answer Pointers

a) Black box testing is where the internals of the system are ignored and the focus of the testing is system functionality or the relationship between system inputs and their corresponding outputs.

b) Advantages include:
- Tester does not need any knowledge of the complexities of the system
- Test plans can be constructed as soon as the function specification has been completed.

Disadvantages include:
- Difficult to identify all possible inputs in a limited testing time
- There is a risk of having unidentified paths through the system during testing.

c) Systems often fail on boundaries so Boundary Value Analysis is a method where the extreme boundary values are chosen for testing. Boundary values include the maximum, the minimum and values that are just inside and outside of these boundaries.

d) Test values that would be used are: 49, 50, 51 on the lower boundary and 119, 120, 121 on the upper boundary

Examiners' Guidance Notes

Although this was a popular question, few candidates gained maximum marks due to issues with the concept of boundary value analysis. Parts a) and b) were answered well, with most demonstrating the required knowledge and understanding.

B6

Answer Pointers

a) F = 2 * C + 30

b) Accept pseudocode or program

```
PROGRAM Celsisu2Fahrenheiut
        Variable Declarations
        REAL LowC, HighC, Interval, C, Fexact, Fapprox, Fdifference)
        BEGIN
                OUTPUT ("Enter Lower Celsius Value");                    {Input data from keyboard}
                INPUT "LowC"
                OUTPUT ("Enter Higher Celsius Value");
                INPUT "HighC"
                OUTPUT ("Enter Temperature Interval");
                INPUT "Interval"

                OUTPUT ("CELSIUS to FAHRENHEIT Conversion Table")
                OUTPUT ("Deg C      Deg F       F approx.      F difference")

                C ← LowC                                                 {Initialise degrees C}

                WHILE (C < HighC) DO
                        Fexact   ← 9 * C/5 + 32;
                        Fapprox ← 2 * C + 30;
                        Fdifference  ← Fapprox – Fexact;

                        {Output data to row of table}
                        OUTPUT ("C     Fexact       Fapprox.      Fdifference")

                        C ← C + Interval;                                {Increment C for next pass}

                ENDWHILE

        END
```

Examiners' Guidance Notes
Few candidates attempted this question and those that did generally failed to achieve high marks. Although part a) was generally answered well, for part b) candidates often failed to provide an input mechanism or failed to output the results or omitted the requirement to calculate temperatures using both formulae.

B7

Answer Pointers
a) Sequential Access means that available addresses are accessed in a predetermined, ordered sequence.

b) Tape device is an example of a sequential access drive, where the drive must move the tape forward or backward until it reaches the destination it wishes to read from or write to.

c) A typical scenario is batch processing [as opposed to online response] - as in a payroll system working through everyone on the payroll in order

Examiners' Guidance Notes

Candidates generally understood the concept of sequential access and gained some marks for part a). Part b) was not answered so well, with many providing non-computer based examples of sequential access or just naming devices that could be used, without any further explanation. For part c), some situations were described, including payroll and the use of magnetic media for security copies.

B8:

Answer Pointers
a) Linear Search and Binary-Chop

b)

Linear Search
No special requirements of array
Start at one end and compare each element with value sort
Stop when match found or end of array met

Binary Chop
Array must first be sorted (e.g. into ascending order)
i)
a) if no elements remaining - finish - not found
b) else choose the middle element of array and compare with value sought
ii)
a) If match - value found - finish
b) If value sought is smaller, reject upper half of array and repeat from i) with remaining elements
c) If value sought is larger, reject lower half of array and repeat from i) with remaining elements

Examiners' Guidance Notes
A popular question where many candidates gained full marks. Some lost marks by failing to mention that binary searches require the array to be sorted first. There is evidence that a significant proportion of candidates confused "search" with "sort" and answers describing bubble and other sorting algorithms gained no marks.

B9

Answer Pointers

a) The two phases of the software development life cycle that follow analysis are Design and Implementation

b) Two popular methods of sorting are bubble sort and insertion sort

c) Linux OS is an example of Systems software and Microsoft Word is an example of Application software.

d) The hexadecimal system operates in base 16 and uses digits the (0 to 1 and A to F), whereas the octal system operates in base 8 and uses the digits 0 to 7.

e) The acronym GUI stands for Graphic User Interface.

f) The basic (or primitive) data type Boolean can have values of true and false.

g) White space is any program text that is made up of space, tab and newline.

h) Syntax errors are detected at compile time, the compilation cannot be completed until these errors are corrected.

i) An algorithm is a set of instructions to solve a problem; it can be in the form of a flowchart or pseudocode.

j) Two ways to obtain a one-from-many selection from a user on a web-based form are radio buttons and a drop down menu.

k) The alternative names for the data structures called LIFO and FIFO are Stack and Queue.

l) Random Access means reading from or writing to a memory location directly, rather than being accessed in a fixed sequence.

Examiners' Guidance Notes

Another popular question where candidates gained high marks. Note, that to gain a mark, correct words were needed for each of the two omissions in the sentence. Where valid, alternative words were accepted. For example, for Linux, "open source" was accepted.

B10

Answer Pointers

a) Compiler: translates high level code to low level code as a complete task
Interpreter: translate high level code to low level code a section at a time
Difference: compiler equivalent to task of translating a book into a foreign language, interpreter equivalent to task of translating a live speech, sentence by sentence

b) Internal memory: memory directly accessible by the CPU
External memory: memory indirectly accessible, removable (easily)
Difference: internal smaller, faster, direct access – external larger, slower, serial or direct access

c) Design: Choose data structures, choose algorithm, create pseudo code, may involve ER diagrams, dataflow diagrams, database design, flowcharts.
Implementation: Choose the programming language and, using the design ideas, write the final code.
Difference: design is characterized by diagrams on paper, implementation by executable code.

Examiners' Guidance Notes

Generally, this question was answered well, although future candidates answering similar questions would be advised to follow the advice that three sentences should be used. For part a), there is evidence that many candidates misunderstood the difference between compilers and interpreters. Part b) was answered reasonably well, although the examiners were looking for more than just "internal memory is internal and external memory is external". For part c), alternative answers regarding the different textbook definitions of "implementation" were accepted.

B 11

Answer Pointers

a) Syntax error - programmer not obeying the structure rules of the programming language

Type error - programmer not using the correct types for an operator, or not using a variable in a way that is suitable according to its declaration

Overflow error - programmer causing the machine to calculate a number which is too large to fit into the designated field be represented in the machine

b) Syntax error - e.g. omitting a closing bracket in if(x>1 {y=2;z=3;}

Type error - e.g. int x; x[2]=1; is a type error because x is being used as an array but only declared as a simple integer; y=2+"hello" is an error in many languages

Overflow error - e.g. z=0; x=y/z; Dividing by zero leading to an 'infinite' result is a classic overflow error, but overflow can happen in simple addition if the result is larger than the maximum integer for the machine

Examiners' Guidance Notes

Many gained high marks for this question, providing clear and pertinent examples . However, there is evidence that many lost marks by just jotting down code where it was difficult for the examiners to see any syntax errors, read the variables and arrays used.

B12

Answer Pointers

For maximum marks answers were well described and linked to the functionality of a travel agent, including features such as:

- Drop-down lists- for users to choose required dates, or required location
- Search Engine – to produce lists of possible flights / locations and alternatives
- Use of Frames – to maintain navigation within the travel agency site while offering options such as travel, hotel, car hire
- Radio Buttons – for selection of preferences such as airline or hotel.

Examiners' Guidance Notes

Some good answers that demonstrated clear understanding. Marks were given for alternative answers where the candidate described generic features of a GUI interface, such as colour, fonts and the need to provide accessibility. Note that for full marks, the functionality of each feature needed to be described in the context of a travel agency.