THE BCS PROFESSIONAL EXAMINATIONS
BCS Level 4 Certificate in IT

October 2007

EXAMINERS' REPORT

Software Development

General Comments

There are many weak candidates who possess nowhere near the appropriate skills to pass this paper. They generally get marks below 15%.

Candidates should learn to answer questions without extensive rough work.

Algorithmic development in problem-solving continues to be weak. Flowcharts have been superseded and should not be taught as a design method. They are not penalized per se.
Declaration of variables in programming questions should not be given unless specifically asked for or distinction of (say) *integer* and *real* is important in the answer.
System directives (For example *uses crt ,clscr*, *#include maths*, *# include <iostream.h>* or *void main()*) are **never** required.

Record descriptions requested to answer one part of a question should not be duplicated in a later part where it is used. It is not penalized but it merely wastes candidates' valuable time. Section 'A' questions should not be attempted as time runs out

A few candidates numbered the questions incorrectly; many more left out parts of a question they had attempted. About 20 % of candidates did not fill in the box at all.

Section A

Question 1

a) A programmer is encouraged to write explanatory comments into a program. Explain why the comment below is not a good comment.

```
x=(b+a)/2;              // x assigned (b+a)/2
```

(5 marks)

b) Choose either version of the code below and then write comments appropriate for
   i) line 1
   ii) line 3
   iii) line 4
   iv) line 5
   v) line 7
   vi) line 9

(18 marks)

	Version 1	Version 2
1	int f(int p, q)	INTEGER FUNCTION f(p, q AS INTEGER);
2	{	BEGIN
3	int x, y;	INTEGER x, y;
4	y = 0;	y ← 0;
5	for(x=p; x<=q; x++)	FOR x FROM p STEP 1 TO q DO
6	{	BEGIN
7	y = y + v[x];	y ← y + v[x]
8	}	END;
9	return(y);	f ← y
10	}	ENDFUNC

c) Based on your answers to (b), decide on better names for the identifiers (the function name, its parameters and the variables).

(5 marks)

d) A subprogram is independent if all the values it requires are provided inside it or come via parameters. State, with reasons, whether the function f is independent in this sense.

(2 marks)

Answer Pointers

a) A comment should explain, not repeat the code. Here it might be more appropriate to state "Calculate the average of a and b"

b)
i) function f to return the sum the values in array v from index p to q
ii) declare local variables x for loop counter, y for total
iii) initialise total
iv) loop on x starting at index p, to index q (inclusive)
v) add the value from array v at subscript x to the current total
vi) exit from function with the final value from the total y

c)
f --> sum / total / sumArray / ...
p --> lower / low / start / ...
q --> upper / high / stop / ...
x --> i / index / ...
y --> total / sum / ...

d) f is not independent because it still relies on the array v being available as a global

Examiner's Guidance Notes

a) Some candidates wrote that the variables a & b were not declared and that their type was not known. This was true but it would not have affected the comment on the line in question if the extra information had been given. Some candidates seemed to be looking for syntax errors in the line given.

b) Many candidates who may have had the right idea did not obtain full marks for parts of this question because they used words incorrectly. It seemed at times that candidates were using all the words: 'declared', 'initialised', 'defined', 'assigned' as though they were interchangeable. Some were even using words like 'initiated' and 'decoded' (presumably intending initialised, declared). There was again confusion between the meaning of 'while' and 'until' in explaining the loop behaviour. Using the phrase 'is-equal-to' is **not** acceptable when 'assignment' is meant. v[x] is **not** v multiplied by x. Line 1 is **not** a call of a function.

The loop counter is **not** started at 1. Return(y) does **not** mean print(y). There was confusion when using the phrases 'is-assigned' and 'is-assigned-to'. **Correct**: y is assigned zero, zero is assigned to y. **Incorrect**: zero is assigned y, y is assigned to zero.

Only half marks were awarded for transliterating the code e.g. writing 'assign zero to the variable y' for 'y=0;'. Full marks were only obtained by candidates who had understood the code sufficiently to realise that the code was summing some elements of an array and who could therefore speak of 'inititialising the total to zero'.

c) This was 5 marks for writing down 5 new, well-chosen names for the identifiers used in the code. However not all new names were acceptable. Examples of suggestions which were **not** acceptable were f changed to calculate, p,q to num1,num2 and x,y to val1,val2.

d) There were far too many answers stating that f was independent.

Question 2

Consider either version of the recursive function r shown below.

a) Write down the textual output and the numerical result that you would expect from the function r following the call r(1);

(4 marks)

b) Write down the textual output and the numerical result that you would expect from the function r following the call r(2);

(6 marks)

c) Write down the textual output that you would expect from the function r following the call r(4);

(10 marks)

d) What result is obtained by the call r(4) ?

(4 marks)

e) The printing in this function has been put there by the programmer to help during testing. Why is it not good practice to have a function which both creates output and returns a result?

(6 marks)

Version 1	Version 2

int function r(int p){
  int x;
  x = 1;
  printf("p on entry %d \n",p);
  if(p > 1)
  x = p * r(p – 1);
  printf("p on exit %d\n",p);
  return(x);
}
	INTEGER FUNCTION r(p as INTEGER)
	  INTEGER x
	  x ← 1
	  PRINT "p on entry" p
	  IF p GREATER THAN 1
	  THEN x ← p * r(p – 1)
	  PRINT "p on exit" p
	  RETURN x
	ENDFUNC

Answer pointers

a) p on entry 1
p on exit 1
result: 1

b) p on entry 2
p on entry 1
p on exit 1
p on exit 2
result 2 * 1 = 2

c) p on entry 4
p on entry 3
p on entry 2
p on entry 1
p on exit 1
p on exit 2
p on exit 3
p on exit 4

d) result 4 * 3 * 2 * 1 = 24

e) Essence: unwanted side-effects

Functions which return results can be used inside expressions e.g. 4*r(6)+1 and it is not expected in this context that evaluation of the function would cause side-effects (i.e. printing in this case)

Examiner's Guidance Notes

a, b, c) Not a popular question and not very well answered. There are two output statements that happen in every function call, one before and one after the recursive call. Thus if r(4) was to call r(3) then any output that comes from r(3) would be 'sandwiched' between the initial and final output from r(4).

Many candidates wrote down the output from the first output statement, but not the second and thus showed only half the actual output. This was generously awarded half marks.

d) Many answers to this part just contained a single number (e.g. 34) which, if wrong, scored no marks. It would have been better to show some working as well e.g. 4*3*2*1. But even amongst answers that did show working, there were still mistakes, like 4+3+2+1.

e) The useful expression 'side-effect' does not seem to be known. However marks were awarded for describing such functions as 'confusing' (or similar).

Question 4

Each day an Examiner marks some questions from A[1], A[2], B[5],B[6], B[7], B[8] from this Examination paper. He needs to input daily the number of questions marked that day for each of these questions and then calculate the number of A-questions marked, the number of B questions marked and the total number of questions marked. The totals for A and B are written to a file and the table shown below is printed.

Table of results

Question number	A1	A2	B5	B6	B7	B8	Total-A	Total-B	Grand-total
Day 1 totals	12		9	5			12	14	26
Day 2 totals	4	2	50		16		18	80	98

Develop the Initial Algorithm given below to a stage where coding would be straightforward. At least TWO stages must be shown.

Initial Algorithm
Open **questions marked** file and **totals** file
Read in the number of questions marked that day for A1, A2, B5,…, B8
Append the number of questions marked to the **questions marked** file
Read previous totals from file (Total-A, Total-B)
Calculate the new Total-A
Calculate the new Total-B
Output the new totals to the **totals** file
Calculate the new Grand-total

(30 marks)

Answer pointers
Algorithmic Development

```
ASSIGN file <questots> for READING
REPEAT
        PRINT "Section A or Section B questions?"
        INPUT reply  {needs checking as character 'A' or 'B'}
        IF reply = 'A' THEN
                PRINT "input question number"  INPUT sub
                PRINT "input mark awarded"     INPUT A[sub]
        ELSE    {Section 'B' question assumed}
                PRINT "input question number"  INPUT sub
                PRINT "input mark awarded"     INPUT B[sub]
        ENDIF
        PRINT "another question?"
        INPUT reply
                                UNTIL reply = 'N'
(* form question totals *)
tot-A = tot-B = 0
ADD all of A[sub] into tot-A
ADD all of B[sub] into tot-B
(* add in previous question totals from file record *)
READ (questots, tot-rec)
ADD tot-rec.A into tot-A, tot-rec.B into tot-B
Grand-total = tot-A + tot-B
PRINT tot-A, tot-B, Grand-total
```

OPEN questots for WRITING
WRITE (questots, tot-rec)
CLOSE questots
ENDPROG

Tested PASCAL program

```
PROGRAM writquest(INPUT,OUTPUT,questots);
{Stores last set of totalled questions for A,B sections}

TYPE    questype = RECORD
      tot_A, tot_B:INTEGER;
           END;
VAR  sub, tot_A,tot_B, Grand_tot:INTEGER; reply:CHAR;
    A:ARRAY[1..2] OF INTEGER; B:ARRAY[5..8] OF INTEGER;
   questots: FILE OF questype; totrec:questype;

PROCEDURE Readncheck(VAR achar:CHAR);
BEGIN
REPEAT READ(achar) UNTIL achar IN ['A','B','Y','y','N','n']
END;

BEGIN  {main block}
WRITELN('reads and writes file of question totals called <questots>');
ASSIGN(questots, 'C:\PROPAS\SOURCES\questots');
RESET(questots);
{INPUT section}
REPEAT
   WRITELN('input<A> for section-A or <B> for section-B');
   Readncheck(reply);
   IF reply = 'A' THEN
      BEGIN
      WRITELN('Input question number');READ(sub);
      WRITELN('Input how many MARKED');READ(A[sub])
      END
      ELSE
      BEGIN
      WRITELN('Input question number');READ(sub);
      WRITELN('Input how many MARKED');READ(B[sub])
      END;
   WRITELN('another question?');
   Readncheck(reply);
                        UNTIL reply = 'N';
{form question totals}
tot_A := 0;   tot_B := 0;
FOR sub := 1 TO 2 DO tot_A := tot_A + A[sub];
FOR sub := 5 TO 8 DO tot_B := tot_B + B[sub];
{add in previous totals from file}
READ(questots,totrec);
tot_A := tot_A + totrec.tot_A;
tot_B := tot_B + totrec.tot_B;
Grand_tot := tot_A + tot_B;
WRITELN('total - A    total - B   Grand Total');
WRITELN(tot_A:5,' ':7,tot_B:5,' ':10,Grand_tot:5);
REWRITE(questots);
totrec.tot_A :=tot_A;totrec.tot_B := tot_B;
```

```
WRITE(questots,totrec);
WRITELN('file totals written');
CLOSE(questots,False)
END.
```

Examiner's Guidance Notes

Here the kind of record used for the file (or files) was left to the student's choice. Answers were expected to use the given algorithm, though! Most answers used code immediately from the given algorithm without attempting any development; hence the coded answers were wrong!
An unpopular question and ***very*** few answers had an algorithmic development as required. This attracted most answers from the weaker candidates who did not understand what was expected, hence the low marks.
Copying out the given algorithm without development gained no extra marks. One answer had 11 separate files, but would have worked eventually. Other answers had two or more files when only one was specified. Most answers were also poor on file usage; these needs attention in future teaching.

Question 4

A bookshop enquiry system contains details of books held in a warehouse. Customers make enquiries about particular books. Each book has a record structure as shown below:

BOOK		
	NAME	
		Title 30 characters.
		Author 30 characters.
		Publisher 30 characters.
		ISBN-Number 10 digits.
		Catalog-ref 10 characters.
		Hardback True/False
	LOCATION	
		Row-number 1 character, 1 digit.
		Aisle-number 1 character, 1 digit.
		Shelf-number 1 digit.
	DETAILS.	
		Price per copy (decimal currency)
		In-stock Indicator (yes/no)
		Number of copies available (whole number)

a) Specify this record structure in an appropriate programming language. State clearly which language you are using.

(6 marks)

b) Set up a file or table to hold up to a thousand such records. You MUST be quite clear about what type of file you are using.

(4 marks)

c) Develop an algorithm or pseudocode for a series of enquiries for particular books. An enquiry may use any of the entries under 'NAME'. For those whose entries are found, the data stored under 'LOCATION' is to be printed. If it is not found print 'NOT AVAILABLE'. If the 'Hardback' value is true, print 'HARDBACK BOOK'. The price is then printed and the customer is asked if he/she wishes to purchase it. If the answer is 'YES' the file is updated and the in-stock indicator adjusted accordingly. After each enquiry the user is asked whether he/she wishes to terminate or continue with enquiries. Counts must be made of enquiries; when the last enquiry has been processed how many successful and unsuccessful enquiries are displayed.

Two stages of algorithmic development must be given in your answer

Initial Algorithm

(7 marks)

Developed Algorithm

(13 marks)

Answer Pointers

(a) Pascal declaration of record structure

```
TYPE NAME = RECORD   {other structures are possible with all the items in ONE record)
            Title:      string[30];
            Author:     string[30];
            Publisher:  string[30];
            ISBN_Number:longint;
            Catalog_ref:string[1];
            Adult-book: BOOLEAN
            END;

      LOCATION = RECORD
            Row_number:       string[2];
            Aisle_number:     string[2];
            Shelf_number:     shortint
            END;

      DETAILS = RECORD
            Price_per_copy:   REAL;
            In_stock_Indicator: BOOLEAN;
            Copies_available: shortint
            END;

      BOOK = RECORD
            Name_rec:         NAME;
            Location_rec:     LOCATION;
            Detail_rec:       DETAILS
            END;
```

b) Set up a file or table to hold up to a thousand such records. You MUST be quite clear about what type of file you are using.

```
VAR bookbuy:FILE OF BOOK; Abook:BOOK;
```

This declares <BOOK> as a simple sequential file. Structured files can be declared for use here such as an indexed sequential file in COBOL or a structured file in Pascal.

c) Write an initial algorithm or pseudo-code for a series of enquiries for particular books. An enquiry may use any of the entries under 'NAME'. For those whose entries are found, the data stored under 'LOCATION' is to be printed. If it is not found print 'NOT AVAILABLE'. If the 'Adult-Book' value is true, print 'RESTRICTED TO OVER 18'. The price is then printed; the customer is asked if he/she wishes to purchase it. If the answer is 'YES' the file is updated and the sale recorded.

After each enquiry the user is asked whether he/she wishes to terminate or continue with enquiries. Counts must be made of enquiries; when the last enquiry has been processed how many successful and unsuccessful enquiries are displayed.
One development stage must be given for your initial algorithm.

Initial algorithm

(

Developed algorithm

Initial Algorithm

```
OPEN booklist file for READING
INPUT details of book enquiry
WHILE NOT finished DO
        SEARCH booklist file for match of book data with enquiry data    note #1
        IF found THEN PRINT location data
                IF adult-book THEN PRINT "restricted to over 18"
                PRINT book price
                PRINT "do you wish to purchase it?"
                IF reply = yes    THEN update bookdata file
                                  ELSE PRINT "not available"
                ENDIF
        ENDIF
        PRINT "another enquiry"
        IF no THEN finished = TRUE
 ENDWHILE
```

Note #1 COBOL provides a SEARCH verb. Other languages require a WHILE NOT end-of-file condition to be used combined with an exit if the book is matched with the search item before reaching end-of-file.

Development

```
ASSIGN booklist file
OPEN booklist for READING
Reply ← yes    found ← false
Set counts to zero                                                  Note *1
WHILE reply <> no DO
   INPUT enquiry                                                    Note *2
   READ booklist, book-data
   WHILE NOT end-of-file(booklist ) AND NOT found DO
                ADD 1 TO enq-ct
                Test enquiry-detail with file data          Note *3
                IF found THEN
                   ADD 1 TO available-ct
                   PRINT location-data
                   IF adult-book THEN PRINT "restricted to over-18"
                                      ELSE PRINT book-price
                                           PRINT "do you wish to purchase it?"
                                             IF buy = yes THEN
```

```
                                                    Update booklist data
                                                    record sale
                                                ENDIF
                    ENDIF
                  ELSE {book data not match enquiry data}
                   READ(booklist, book-data)
                  ENDIF
   IF EOF(booklist) AND NOT found THEN PRINT "not available"
                                                    ADD 1 TO unavailable-ct
         ENDIF
         PRINT "another enquiry?"
         INPUT reply
   ENDWHILE
ENDWHILE
PRINT available-ct  unavailable-ct
CLOSE booklist
ENDPROG
```

Development notes

*1. counters set to zero:

enq-ct ← 0 available-ct ← 0 unavailable-ct ← 0

*2. This must indicate the type of enquiry, which of the book keys is being used in the search. This can be input as <choice> where this is 1,2,3 or 4. A CASE statement can then be used thus:

```
CASE choice OF
1 : Book Title Enquiry: requires title input
2 : Author enquiry: requires author's name input /format could be complex
3 : ISBN enquiry : simplest match criteria
4 : Catalogue reference/format needs example
ENDCASE
```

*3 A similar match test using CASE can be used; the book enquiry item must be tested against the appropriate data from <bookfile>.

End Notes. While this algorithm does not cover every possible combination of circumstances or error conditions it is satisfactory for virtually all enquiries.

It is essential in this answer to get the IF…THEN…ELSE structures right. Usually this is done by an algorithm which leaves the actions to be filled in later on. In an exam answer the details needed in a runnable program should be omitted such as formatting appropriate output items.

Examiner's Guidance Notes

The marking scheme was varied slightly so that candidates were not penalized for answering (b) as part of (c), where they had actually declared the file as part of the algorithm or final program. Where they had done this (b) 2 marks and (c) 22 marks was used. Over 90 % of the candidates did this. Where they had done what was intended the printed mark scheme was used.

A popular question but many candidates only attempted the RECORD description required for (a). Few made a realistic attempt at an algorithm; those who did usually scored high marks. No marks awarded for copying out the question as printed! Some candidates wrote lengthy and usually incorrect answers to (b) despite only 2 marks being available. The examiner did not penalize those candidates who did only one algorithm without showing development; the planning was probably done as rough work.

Section B

Question 5

a) Write a suitable structure/record to hold data on jobs set down for a group of technicians to perform. The data fields are as follows:

JOB-DESCRIPTION.

Date (format day.month.year)	10 characters
Job number	6-digit integer
Estimated cost	decimal currency
Authorisation	yes/no
Description	30 characters

(3 marks)

b) At the end of each working day a file named newjobs is created containing jobs which came in that day and any not cleared from previous days.

Write an algorithm or program which reads this file and prints out all the data for those jobs which are authorized or the date is more than 2 days old. Count how many such jobs are found and print out this total at the end of the report.

(9 marks)

Answer pointers

Algorithm

```
Jobct ← 0
Get todays_date
Open file for reading
 WHILE NOT EOF(newjobs) DO
        BEGIN
        READ(newjobs,onejob);
        IF onejob.Authorisation  OR  onejob.adate < today_date
                THEN
                        WRITE(onejob in format)
                        ADD1 1 TO Jobct
        ENDIF
        ENDWHILE
PRINT Jobct
ENDPROG
```

Pascal Program

```
PROGRAM joblist(INPUT,OUTPUT,newjobs);
(* BCS Question B5 October 2007 *)
TYPE          Jobs = RECORD
           Adate : INTEGER;
     Estimate_cost : REAL;
     Authorization : BOOLEAN;
     Description   : string[30]
          END;
VAR newjobs:FILE OF jobs; onejob:Jobs;today_date,pos,Jobct:INTEGER;
BEGIN
WRITELN(' AUTHORISED jobs for today');
WRITELN('INPUT todays date in form yyyymmdd  ');
READ(today_date);
Jobct := 0;
```

```
(* Open file <newjobs> for reading *)
ASSIGN(newjobs, 'C:\PROPAS\SOURCES\newjobs');
   RESET(newjobs);
 WHILE NOT EOF(newjobs) DO
      BEGIN
      READ(newjobs,onejob);
      IF (onejob.Authorization)  OR  (onejob.adate < today_date)
            THEN BEGIN
                  WITH onejob DO
                  BEGIN
                  WRITE(adate:10);
                  WRITE(' ':3,' estimated cost ', Estimate_cost:6:2);
                  FOR pos := 1 TO 30 DO WRITE(Description[pos]);
                   WRITELN;
                   Jobct := Jobct + 1
                  END
                 END
      END;
WRITELN(Jobct :4,'Jobs for today');
CLOSE(newjobs,FALSE)
END.
```

Examiner's Guidance Notes

A popular question. Some excellent answers from those who also did the 'A' section programming questions. Bizarre choices for the 'date' part of the record which would prove difficult to use. Those who used a string variable for the date in the declaration were expected to process it as such in part (b); a long integer was a better choice. Some candidates misread the question and expected interactive input; these wasted a lot of time with prompts etc which gained no marks. Others left out the file 'read' instruction from the loop. Some students clearly don't know the difference between interactive input (which they are familiar with from practical work) and file input. They had lengthy interactive input sections for <newjobs> input data before attempting to answer the question as set, if at all. This was not penalized if the candidates then went on to answer the question, but it must have wasted valuable time!
Some algorithmic planning is desirable even in a short question.

Question 6

A fortune telling program has as input a birth date in the form ddmmyyyy (where dd = day digits, mm = month digits, yyyy = year digits.) All the digits of the birthday date are then added together to form a two-digit number, as shown in the example below. Finally, these two digits are added together to reveal the primary indicator, whereby the prospects and capabilities are revealed for the person having that birth date. The program output is the indicator in the form shown below:
Example calculation:
A birth date is 11 May 1989.
Input is 11051989 as the full birth date.
Sum of digits = 1 + 1 + 0 + 5 + 1 + 9 + 8 + 9 = 34
Primary indicator = 3 + 4 = 7
Output: the indicator used in fortune telling = 34/7

You may assume that a FUNCTION length( ), exists with suitable parameters, is available which returns the length of a string of decimal digits.

(12 marks)

Answer Pointers

```
PROGRAM digitpwr(INPUT,OUTPUT);
{works out digit sum and indicator digit from long form of birthdate}
VAR year,howlong,date,digit,sum,shortbday,bday,yr,ct:INTEGER;

FUNCTION length(date:INTEGER):INTEGER;
{students are NOT expected to write this function code}
VAR asum:INTEGER;
BEGIN
asum := 0;
WHILE date > 0 DO
      BEGIN
      asum := asum + 1;
      date := date DIV 10
      END;
length := asum
END;
BEGIN  {TOP LEVEL}
WRITELN('WORK OUT DIGIT INDICATOR FROM LONG BIRTHDATE');
WRITELN('input your birth date in the form ddmmyy'); READ(shortbday);
bday := (shortbday DIV 100 )*10000;
yr := shortbday MOD 100;
IF yr > 20 THEN bday := bday + 1900 ELSE bday := bday + 2000;
bday := bday + yr;
WRITELN('full birthdate is ',bday:8);
{GET indicator}
howlong := length(bday);
WRITELN('date length = ',howlong:2);
sum := 0;
FOR ct := 1 TO howlong DO
      BEGIN
      digit := bday MOD 10;
      bday := bday DIV 10;
      sum := sum + digit
      END;
WRITELN('sum of birthday digits = ',sum:2);
digit := sum DIV 10 + sum MOD 10;
WRITELN('LIFE INDICATOR = ', sum:2,'/',digit:1)
END.
```

Examiner's Guidance Notes

Unpopular and poorly done. Most answers assumed that the date would be input as individual digits which could then be added together. More marks were given for those who read in a long integer and then performed appropriate MOD and DIV operations as shown in the answer pointers.

Question 7

The function Fn(x) = LN(1.0 + x + x^2) may be evaluated using the Maclaurin series:
Fn(x) = x + ½ x^2 - 2/3 x^3 + 3/4x^4 – 4/5x^5 + …

a) Write down the next two terms in this series.

(2 marks)

b) Show how to calculate each term in the series.

(4 marks)

c) Write a program which evaluates Fn(x) from input values of x and N using the Maclaurin series where N indicates the number of terms to be used in the series. It prints this value and compares it with the value obtained using the system function for LN(x) in the function definition. Finally the difference between these two values is printed. Information: LN(x) is the logarithmic function using *e* as base.

(6 marks)

Answer Pointers

a) The next two terms are +5/6 x^6 - 6/7 x^7

b) The alternating sign is effected by assigning sign ← (-1) and using
[new] sign = [old] sign * (-1)
The fraction is formed using a variable <ct> initially set to 2 and incrementing it in a loop thus : mult ← (ct – 1)/ct .Each successive x-term is got from its predecessor by
[new] x-term ← [old] x-term * x
Each term in the series is added in by sign*mult*xterm.

c)
```
PROGRAM mclaurin(INPUT,OUTPUT);
(* Evaluates ln[1 + x + x*x] using Maclaurin Series *)
VAR sign,ct,nterms:INTEGER; x,xterm,mult,series,val,diff,fn:REAL;
BEGIN
WRITELN('Maclaurin series for ln[1 + x + x*x]');
WRITELN('input how many terms'); READ(nterms);
WRITELN('input the value for <x>'); READ(x);
sign := -1;
xterm := x;
series := x;
FOR ct := 2 TO nterms DO
      BEGIN
      sign := sign *(-1);
      mult := (ct - 1)/ct;
      xterm := xterm *x;
      series := series + sign*mult*xterm
      END;
fn := 1.0 + x + SQR(x);
WRITELN('function evaluates LN[',fn:5:3,']');
WRITELN('function after ',nterms:3,' terms = ',series:8:6);
val := LN(fn);
WRITELN('actual value = ',val:8:6);
diff := ABS(val - series);
WRITELN('difference = ',diff:8:6)
END.
```

Examiner's Guidance Notes

B7. Surprisingly popular, but weaker candidates gleaned only 2 marks for providing the next 2 terms in the MacLaurin series. One student recognized the series summation was

$$\sum_{n=1}^{\infty} (-1)^{(n+1)} * \frac{n}{n+1} * x^{(n+1)}$$

and had an original method for answering the question! Candidates were not expected to have done mathematics at this level, but those who have done so possess an advantage in that they understand logical thinking.

Those who have not been taught this kind of programming problem would do well to leave it alone.

Question 8

You are a Manager in a small software house with 10 professional programmer/analysts. You have been invited to amend some numerical analysis software, which is written in FORTRAN. The amendments are necessary due to a regulatory change. The software consists of approximately 30 working programs but not all of the programs need changing. Documentation, in some detail, is available for the larger programs but none exists for the smaller programs beyond comments in the code. The software house does not currently possess expertise in FORTRAN and redevelopment of all of the software in C/C^{++} is not an option as some of the supplied programs are very large.

a) On what basis would you decide whether to undertake this work or not? Give reasons.
(6 marks)

b) State what facilities are required to speed up development of these programs.
(3 marks)

c) Draft a test plan to develop systematically the necessary software.
(3 marks)

Answer Pointers

a) One of your staff must possess (or develop quickly) adequate expertise. You must be certain that you have someone who can do this. Alternatively you could hire a temporary programmer with this expertise.
Adequate documentation of the language must be available. There are numerous texts available on FORTRAN 72 from libraries.
Your knowledge as Manager must then enable you to estimate the time to carry out the work, how many programmers are employed on it, and compare with the payment for the work.

b) A thoroughly tried & tested compiler for FORTRAN-72 must be available. This can probably be obtained from the Internet. It must give helpful error messages or the task could be too difficult. Included with this is documentation so that you can get it installed quickly on your existing hardware.
There must be test data for all of the programs, either with the listings or worked out during testing. This is important for the larger programs; without it the task could take too long.
You must have appropriate practical facilities to install this software on at least 2 PC's.
With the compiler you need several test programs which include the major language features such as file and array processing, subroutines.

c) Test & document the smaller programs; everyone can learn during this period. Gradually explore the larger ones developing test data as you go or from books/internet. Ensure that a specification is obtained from your customer as to just what he/she requires. This should include a priority list if all the programs are not to be completed before hand-over.
Finally write the new FORTRAN programs not included in the available set.
It was also acceptable to decline the contract as the software house does not possess the necessary expertise in FORTRAN; much would depend on what other work is in progress in the software house and whether there are sufficient resources to undertake a project in a different, old programming language.

Examiner's Guidance Notes

Quite popular, but a lot of woffle in the answers. To get more than 6 marks, candidates had to apply their knowledge to the specific problem given, rather than vaguely write about testing. Writing completely new suites of programs in C was inappropriate here. Too much time has been spent in classes on designing completely new software rather than adopting existing programs, work which is often carried out in practice. Few answers had points made succinctly and clearly, but waffled on generally about various forms of testing they had committed to memory. Repetition of the same idea expressed differently does not gain extra marks.
The Examiner recommends that this question be used as homework as:
i) Candidates get opportunity to apply general principles,
ii) Adequate communication skills in English can be shown here.
a) Those who provided cogent reasons for declining the contract were given good marks; this is an acceptable alternative on the information given.
b) Very few realized a user-friendly FORTRAN compiler was essential here!
c) This was answered by 'brain dump' i.e. candidates wrote out what they had rote-learned about general software development with no thought of its application to this problem. they got 1/3marks. Some quite wrong ideas here, such as prototyping, quite unsuitable give the nature of the task and the time constraint.
The examiner wanted to see if candidates could apply their knowledge; mostly he got regurgitated notes without thought as to relevance.

Question 9

a) Name the stage of the traditional software development process in which an algorithm would be introduced and give your reasons?
(4 marks)

b) Name the stages that precede and follow the stage mentioned in (a) and by concentrating on the algorithm describe the links between the three stages.
(8 marks)

Answer Pointers

a) Stage: design
Find a programming description of how a task can be achieved

b) Stages: Specification -> Design -> Implementation
Take the requirement as described in the specification and then design a programming description of how it can be achieved and then implement by coding into a particular programming language

Examiner's Guidance Notes

Not very popular, quite poorly answered.

a) Some students saw the phrase "traditional software development process" and wrote out a description of all the stages - this scored no marks. This was a tragic waste of time. One candidate wrote for SEVEN pages without scoring any marks.

Nowhere in (a) does it ask for a definition of an algorithm, so writing such a definition in an answer is a waste of time.

The coding stage was occasionally given as a place where the algorithm is *found* but this is not the stage where it is *introduced*.

b) The marking was fairly flexible as regards the actual names of the stages accepted, providing the accompanying explanation made sense.

Do the candidates understand the words precede and follow? Why were there so many answers with two following stages (and no preceding stage)?

Question 10

a) What does the term Sequential Access mean?

(4 marks)

b) Briefly describe a part of a computer system which can be used as a Sequential Access device.

(4 marks)

c) Give brief details of a situation in which a Sequential file would ideally be used.

(4 marks)

Answer Pointers

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

b) Tape device

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

Examiner's Guidance Notes

A more popular question and generally fairly well answered. Some problems are referred to below.

a) Once again explanations used the very term which was supposed to be explained. For example it is unacceptable to write "Sequential access is accessing sequentially".

The primary meaning of sequential is not to do with speed or ascending/descending order.

There is a persistent confusion amongst candidates between the word file and record. For example it was quite common to find a sentence like "To access the file to get to the one you want you have to access all the preceding files". A correct version would be: "To access the file to get to the desired *record* you have to access all the preceding *records*"

b) Examples of unacceptable answers given here were: Hard disc, CPU, database, keyboard, RAM (*random* access!), scheduler.

c) Not enough just to give as an example a 'file of student records'. It has to be made explicit what operation is being performed - e.g. list all students.

Several answers gave a 'telephone directory' as an example of a sequential file - how often is an entry found in a telephone directory by reading every single entry from the beginning?

Several answers mentioned 'backup' (especially to tape) - this was accepted as a good answer

Question 11

A web form is to be used to gather information from students applying to a university computing department to join a course. The information required is the name of the student, their gender, which of the subjects Mathematics, Physics, Computing they have studied at school and which degree course they want to study.

a) State which type of form element is appropriate for each piece of information giving a reason.

(8 marks)

b) Sketch how the final form might look as displayed by a web browser.

(4 marks)

Answer pointers

a) name of student - text box - textual input, not known in advance
gender - radio button - 1 out of n choice (with n small)
subjects - check boxes - multiple choice of n
course - menu - 1 out of n choice (with n large)

b) Simple sketch showing what the form elements look like on a browser screen, with some labelling

Examiner's Guidance Notes

A very popular and well answered question.

a) 'State ... giving a reason' - Candidates who omitted any reasons were limiting themselves to half marks

There is evidence that there is confusion between a label and the form element that it labels. And very few can spell label (lable).

b) It was common to find unwanted elaboration/extras beyond what was asked for in the question.

A worrying number of candidates apparently changed their mind between parts (a) & (b) so that their sketch in (b) did not match their answer to (a).

Question 12

a) There are many kinds of errors that can occur in a program. Describe the following kinds of errors:
   i) Syntax error
   ii) Type error (type check error)
   iii) Overflow error

(6 marks)

b) For each of the kinds of errors mentioned in (a) give simple program extracts in a language of your choice that demonstrate the error.

(6 marks)

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 try to calculate a number which cannot be represented in the m/c

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

c) 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 m/c.

Examiner's Guidance Notes

Quite a popular question - with disappointing answers.

a) As pointed out earlier it is not acceptable to say that "a syntax error is an error in syntax".

Many answers displayed a poor grasp of the necessary vocabulary (e.g. An error is where a syntax is missing).

Many answers stated "Syntax errors are mis-spelt keywords". Although there is an element of truth here, the implication that all syntax errors are mis-spellings or that all syntax errors are to do with keywords is wrong.

Several answers gave mis-spelling of function names as a syntax error e.g. writln("x") instead of writeln("x"). However this is most likely to give a "function not declared" error. In general "identifier not declared" is not a syntax error.

b) There were several instances where a candidate gave an example of an identifier like *name* being declared with type integer and saying that it was a type error. But of course our compilers do not 'understand' our language. It is only later in the program, when we try to perform some operation like name:="Smith", that a type error would show up.

Despite the hint (typecheck error) in the question, candidates still wrote about 'typing' errors as though it meant typing the wrong characters at the keyboard when editing the program.

c) This part of the question demanded that candidates gave actual program extracts - not just descriptions of situations.

An 'infinite loop' is not in itself an example of overflow (consider: while true do x:=x)