INTERNATIONAL BACCALAURÉAT BACHILLERATO IBO BACCALAUREATE INTERNATIONAL INTERNACIONAL

N02/650/H(2)M+

MARKSCHEME

November 2002

COMPUTER SCIENCE

Higher Level

Paper 2

14 pages

Subject Details: Computer Science HL Paper 2 Markscheme

Mark Allocation

Candidates are required to answer ALL questions (***[30 marks]*** for question 1, ***[30 marks]*** for question 2 and ***[15 marks]*** for the remaining three questions). Maximum total = ***[105 marks]***.

General

A markscheme often has more specific points worthy of a mark than the total allows. This is intentional. Do not award more than the maximum marks allowed for part of a question.

When deciding upon alternative answers by candidates to those given in the markscheme, consider the following points:

- Each marking point has a separate line and the end is signified by means of a semi-colon (;).
- An alternative answer or wording is indicated in the markscheme by a "/"; either wording can be accepted.
- Words in (…) in the markscheme are not necessary to gain the mark.
- The order of points does not have to be as written (unless stated otherwise).
- If the candidate's answer has the same "meaning" or can be clearly interpreted as being the same as that in the mark scheme then award the mark.
- Mark positively. Give candidates credit for what they have achieved, and for what they have got correct, rather than penalising them for what they have not achieved or what they have got wrong.
- Remember that many candidates are writing in a second language; be forgiving of minor linguistic slips. Effective communication is more important than grammatical niceties.
- Occasionally, a part of a question may require a calculation whose answer is required for subsequent parts. If an error is made in the first part then it should be penalised. However, if the incorrect answer is used correctly in subsequent parts then **follow through** marks should be awarded. Indicate this with **"FT"**.

1. (a) *A good description **[2 marks]**, **[1 mark]** for an answer with some credit, examples:*
The altered values can be passed back to the calling procedure ***[2 marks]***;
Reference values can be altered ***[1 mark]***; ***[2 marks]***

(b) *A good comparison **[2 marks]**, **[1 mark]** for an answer with some credit, examples:*
A function returns a value via the function name whereas a procedure does not ***[2 marks]***;
A function appears on the right hand side of an assignment statement ***[2 marks]***;
A function has a return statement ***[1 mark]***; ***[2 marks]***

(c) *A good comparison **[2 marks]**, **[1 mark]** for an answer with some credit, examples:*
An iterative function uses a loop (OK by example) whereas a recursive routine (loops by) calling itself ***[2 marks]***;
An iterative algorithm needs an ending condition, unlike a recursive one ***[2 marks]***;
A recursive algorithm calls itself ***[1 mark]***; ***[2 marks]***

(d) One solution is:

```
function BINARYSEARCH ( val WANTED integer,
                        val LOW integer,
                        val HIGH integer )      result integer

/ * Returns file position if found or - 1 if not * /

declare MIDPOINT integer
declare CURRENT is ITEM

if (HIGH < LOW) then
      return -1

else

         MIDPOINT <-- (LOW + HIGH) div 2
         moveto(DATA,MIDPOINT)
         input (DATA)CURRENT

     if CURRENT.CODE = WANTED then

           return MIDPOINT

     else

       if CURRENT.CODE > WANTED then
               BINARYSEARCH(WANTED, LOW, MIDPOINT - 1)
       else
         BINARYSEARCH(WANTED, MIDPOINT + 1, HIGH)

       endif

     endif
endif

endprocedure BINARYSEARCH
```

[1 mark] for a correct set of parameters;
[1 mark] for declaring the function **return** type correctly;
[3 marks] for a correct base test:
 [1 mark] for any test, ***[1 mark]*** for correct, *e.g.* `HIGH < LOW` and ***[1 mark]*** for a return of `-1`;
[1 mark] for correct return of required position, *e.g.* `MIDPOINT` when found;
[4 marks] for recursive section:
 [1 mark] for a correct comparison and ***[2 marks]*** for a correct recursive call to the lower part of the array. ***[1 mark]*** for any attempt. ***[1 mark]*** for a correct call to the upper part.

[10 marks]

(e) `(CODE > 9999) and (CODE < 100000)` ***[2 marks]***

[2 marks] *for each correct condition, other solutions are possible.*

(f) A solution is:

```
STOCKCHECK()

declare TOTALVALUE real
declare SHORT integer
declare CURRENT is ITEM
declare C integer
C <-- 0
TOTALVALUE <-- 0

while not eof(DATA)

      moveto(DATA, C)
      input(DATA)CURRENT
      SHORT = CURRENT.REORDER – CURRENT.STOCK

      if (SHORT >= 0) then
         output(CURRENT.CODE, CURRENT.DESCRIPTION, SHORT)
      endif

      TOTALVALUE = TOTALVALUE + CURRENT.STOCK * CURRENT.PRICE
      C <-- C + 1

   enddo

output(TOTALVALUE)

end STOCKCHECK
```

Note: that explanatory text is not required in the output statements.

Accept a solution that loops sequentially from `1 to SIZE`.

Watch out for valid C++ / Java statements such as

```
TOTALVALUE += CURRENT.STOCK * CURRENT.PRICE
```

[6 marks] for the loop:

[1 mark] for initializing `TOTALVALUE`;
[1 mark] for **`while not eof`** or equivalent;
[1 mark] for reading in record (syntax unimportant, do not have to use **`moveto`** `C` or whatever – read next record would do);
[2 marks] for if statement correctly outputting the required info (***[1 mark]*** for a good attempt);
[1 mark] for correct summing of `TOTALVALUE`;
[1 mark] for output of `TOTALVALUE`; ***[7 marks]***

(g) A solution is (check carefully for other correct solutions);

```
declare LOWEST integer
declare P integer
declare M integer
declare CODE integer

for P <-- 1 to 500 do
   CODE <-- SALES(P,1)
   LOWEST <-- SALES(P,2)

   for M <-- 3 to 13 do

      if SALES (P,M) < LOWEST then
         LOWEST = SALES(P,M)
      endif

   endfor

   output("Lowest sales for code", CODE," were", LOWEST)

endfor
```

[2 marks] for correct nested loops, ***[1 mark]*** for any attempt at **nested** loops.
[1 mark] for suitable initializing of `LOWEST` or equivalent;
[1 mark] for correctly changing `LOWEST` to a new low value;
[1 mark] for correct output of `CODE` (or `SALES(P,1)`) and `LOWEST` in a reasonable place; ***[5 marks]***

2. (a) ***[2 marks]*** *for a full answer,* ***[1 mark]*** *for an answer with some credit, examples:*
Because the hardware (give by example: bus, CPU, ALU *etc.*) is designed to work in whole numbers of bytes;
The extra space could be used for error checks, such as parity;
It is more convenient for calculations in software to be done using whole bytes; ***[2 marks]***

(b) *Possible differences (****[2 marks]****,* ***[1 mark]*** *for a weak explanation but with some credit)*
Parity is calculated using 1 byte, check sums over a number of bytes;
Parity has to be stated as odd or even, check sums do not;

Possible similarities ***[2 marks]*** *or* ***[1 mark]*** *as above.*
Both are calculated using the state of the bits in a byte;
Both are transmitted together with data (and decoded at the receiving end); ***[4 marks]***

Do ***NOT*** *award full marks for two similarities or two differences*;

(c) ***[2 marks]*** *for a good description,* ***[1 mark]*** *for one with some credit, examples:*
Handshaking is an exchange of signals to establish protocol before any data exchange takes place;
Exchange of signals;
before data transmission;
establishes protocol; ***[2 marks]***

(d) *Two possible advantages for* ***[2 marks]*** *each up to a maximum of* ***[4 marks]****:*
You can see "inside" a digital model unlike the real thing;
It would cost less to duplicate a computer model on CD than to reproduce a physical model;
To make a physical model requires time and/or expertise;
Less chance of damaging the fossil compared to modelling;

Two possible disadvantages for ***[2 marks]*** *each to a total of* ***[4 marks]****;*
The model is "really" 3 dimensional enabling the user to look from any chosen angle or in any light, gives a better "feel";
A physical model can give a good idea of weight, and balance unlike an image;
You don't need an (enormously expensive) computer system for physical modelling; ***[8 marks]***

(e) ***[1 mark]*** *for identifying an issue and* ***[1 mark]*** *for an elaboration × 2:*
Hacking;
– hackers may break in and do damage to research projects, causing researchers to lose many years of work;
Illegally copying CD ROMs;
– the image files are valuable and copyright to the universities;
Theft of project information;
– by another research group *etc.*; ***[4 marks]***

(f) ***[2 marks]*** *for a good outline,* ***[1 mark]*** *for an answer with some credit × 2:*
The password should not be written down;
and left available;
e.g. on desk or attached to computer with yellow sticky *etc.*;
The password should be changed;
on a regular basis;
The password should be of reasonable length ***[2 marks]*** – allow a figure like 6 or more;
The password should contain some non-alphabetic characters ***[2 marks]***; ***[4 marks]***

(g) ***[1 mark]*** *for identifying a change and* ***[1 mark]*** *for an elaboration, examples:*
Has to learn to work with computers;
since all data is stored / processed on them;
Has to learn to interpret 3D images on screen / CD ROM;
rather than from physically handling specimens;
May not have to travel so much;
as specimens can be viewed, downloaded via computer; ***[2 marks]***

(h) ***[2 marks]*** *for each outline × 2,* ***[1 mark]*** *for an answer with some credit:*
Advantages:
Data can be displayed on a web site and viewed from anywhere;
over the Internet thus saving;
postage charges;
print costs;
or time to publication;
Data on a web site can be more easily and quickly updated;
compared to physical publications;
Web sites are interactive compared to traditional photos;
so a specimen can be viewed from different angles easily;

Disadvantages:
The web is a poorer medium for the display of images;
compared to traditional printing;
– still true for a few months yet!
Publications on a web site can be put there by anyone (*e.g.* a practical joker);
whereas conventional scientific publications have to be scrutinised carefully;
Publications on a web site can be changed by unauthorised persons;
but there is no physical evidence of the change; ***[4 marks]***

3. (a) ***[2 marks]*** *for each aspect i.e. storage, time, big O:*
Storage requirements: Both algorithms require the same amount of storage space ***[2 marks]***;
Time requirements: The `MARK` algorithm will execute much more quickly ***[1 mark]*** because no search is involved ***[1 mark]***;
Big O: `SHUFFLE` has a time complexity of O(n) ***[2 marks]*** whereas `MARK` is O(1) ***[2 marks]***;

Since big O implies time requirements, award ***[4 marks]*** for correct answers. ***[6 marks]***

(b) Create a new element / node;
New element pointer field is assigned to rear;
Rear is re-assigned to new element; ***[3 marks]***

(c) A single pointer is needed;
The first / front element points to the last / rear element; ***[2 marks]***

(d) Keep a temporary pointer to the rear of the queue;
follow the pointers around until this pointer is encountered again;
use a variable;
to keep track of the number of nodes (passed through); ***[4 marks]***

4. (a) ***[2 marks]*** *for a good outline of the functions of each register × 3 to a total of* ***[6 marks]****:*
The accumulator which is in the ALU;
is used to hold data;
and the temporary results of calculations;
The current instruction register in the CU;
holds the instruction about to be decoded;
and executed;
No marks for "holds the current instruction" only.
The program counter in the CU;
keeps track of the address of next instruction;
to be executed; ***[6 marks]***

Location of the above registers can be shown in a clear diagram for ***[3 marks]****.*

(b) ***[2 marks]*** *for a good outline,* ***[1 mark]*** *for identifying a suitable development without further explanation / description.*
Multiple pipelines;
are used so that instructions are fetched, decoded and executed in parallel, thus if the following instruction is the next one wanted, its already partly processed;

Multiple processor architecture (parallel processing);
is used where several different programs or segments can be executing simultaneously;
Speeding execution time for certain types of problem; ***[2 marks]***

(c) 6D; ***[1 mark]***

(d) The register is bitwise compared with hex 80 using an AND operation;
If the MSB is set the operation yields 80 otherwise 00; ***[2 marks]***
(accept answers in binary)

(e) (i) ***[2 marks]*** *for a good outline,* ***[1 mark]*** *for a weak answer with some credit.*
A buffer is needed to hold data temporarily;
so that the processor can continue with other tasks;
A buffer is used to store data;
because the CPU sends data faster than the peripheral can process it; ***[2 marks]***

(ii) ***[2 marks]*** *for a good outline,* ***[1 mark]*** *for a weak answer with some credit.*
The double buffers. When one is emptied data transfer switches to the other;
The original buffer can then be filled again by the CPU without delaying the transfer to the peripheral; ***[2 marks]***
Not such a full answer is required for ***[2 marks]****.*

Accept alternative reasonable methods.

5. (a) The record (to be added) already exists in the file;
A customer with same ID number already exists; ***[1 mark]***

Do not accept "customer with same name exists".

(b) ***[2 marks]*** *for a good outline* ***[1 mark]*** *for identifying a suitable reason.*
Direct access;
might be needed for updates to the file;
or to quickly locate a specific customer record**;** ***[2 marks]***

(c) ***[2 marks]*** *for a good answer,* ***[1 mark]*** *for a weak one with some credit × 2:*
The files could have different formats (fields / structures);
The files could be from different hardware systems (incompatible);
A different type of key field can exist in each of the files;
The file(s) may be too large to store in memory;
Two customer records could have the same key / customer number; ***[4 marks]***

(d) *Award marks for points as follows, maximum of* ***[8 marks]****:*
Create a new empty output file;
Read a record from each file;
Repeat the following;
`if ID1 > ID2 then` write the record from file 1 to the new file;
and read a new record from file 1;
`if ID1 < ID2 then` write the record from file 2 to the new file;
and read a new record from file 2;
`if ID1 = ID2 then` error report should be generated;
Until the end of file 1 and file 2 is reached;
if end of file 1 is reached then copy all the remaining records from file 2 to the new file;
if end of file 2 is reached then copy all the remaining records from file 1 to the new file; ***[8 marks]***