IB DIPLOMA PROGRAMME
PROGRAMME DU DIPLÔME DU BI
PROGRAMA DEL DIPLOMA DEL BI

N04/5/COMSC/HP2/ENG/TZ0/XX/M+

MARKSCHEME

November 2004

COMPUTER SCIENCE

Higher Level

Paper 2

14 pages

If you do not have a copy of the current Computer Science Guide, please request one from IBCA.

General Marking Instructions

After marking a sufficient number of scripts to become familiar with the markscheme and candidates' responses to all or the majority of questions, Assistant Examiners (AEs) will be contacted by their Team Leader (TL) by telephone. The purpose of this contact is to discuss the standard of marking, the interpretation of the markscheme and any difficulties with particular questions. It may be necessary to review your initial marking after contacting your TL. **DO NOT BEGIN THE FINAL MARKING OF YOUR SCRIPTS IN RED INK UNTIL YOU RECEIVE NOTIFICATION THAT THE MARKSCHEME IS FINALIZED.** *You will be informed by e-mail, fax or post of modifications to the markscheme and should receive these about one week after the date of the examination. If you have not received them within 10 days you should contact your Team Leader by telephone. Make an allowance for any difference in time zone before calling.* **AEs WHO DO NOT COMPLY WITH THESE INSTRUCTIONS MAY NOT BE INVITED TO MARK IN FUTURE SESSIONS.**

You should contact the TL whose name appears on your "Allocation of Schools listing" sheet.

Note:
Please use a personal courier service when sending sample materials to TLs unless postal services can be guaranteed. Record the costs on your examiner claim form.

1. Follow the markscheme provided, do **not** use decimals or fractions and mark only in **RED**.

2. Where a mark is awarded, a tick (✓) should be placed in the text at the **precise point** where it becomes clear that the candidate deserves the mark.

3. Sometimes, careful consideration is required to decide whether or not to award a mark. Indeed, another examiner may have arrived at the opposite decision. In these cases write a brief annotation in the **left hand margin** to explain your decision. You are encouraged to write comments where it helps clarity, especially for moderation and re-marking.

4. Unexplained symbols or personal codes/notations on their own are unacceptable.

5. Record subtotals (where applicable) in the right-hand margin against the part of the answer to which they refer. Show a mark for each part question (a), (b), *etc.* Do **not** circle sub-totals. Circle the total mark for the question in the right-hand margin opposite the last line of the answer.

6. Where an answer to a part question is worth no marks, put a zero in the right-hand margin.

7. Record the mark awarded for each of the five questions answered in the Examiner Column on the cover Sheet.
Add up the marks awarded and enter this in the box marked TOTAL in the Examiner Column on the cover sheet.

8. After entering the marks on the cover sheet check your addition of all marks to ensure that you have not made an arithmetical error. Check also that you have transferred the marks correctly to the cover sheet. **We have script checking and a note of all clerical errors may be given in feedback to all examiners.**

9. Every page and every question must have an indication that you have marked it. Do this by **writing your initials** on each page where you have made no other mark.

10. A candidate can be penalized if he/she clearly contradicts him/herself within an answer. Once again make a comment to this effect in the left hand margin.

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 penalized. However, if the incorrect answer is used correctly in subsequent parts then **follow through** marks should be awarded. Indicate this with **"FT"**.

1. (a)

F	C	F*C
1	2	2
2	3	6
6	4	24
24	5	120

*Award **[1 mark]** for each of the last two lines, completely correct.* ***[2 marks]***
Do not award any marks for simply an answer of 120.

(b) A solution is:

```
function FACTORIAL(val X integer) result integer


    if X > 0 then
        return X * FACTORIAL (X-1)
      elsif X = 0 then
            return 1
    endif
endfunction FACTORIAL
```

*Award marks as follows up to **[8 marks]**.*

X correctly declared (as int and value);
correct return type from function;
Correctly return value of 1 for X=0 (allow X=1, return 1 also)
if test for termination in place of iterative loop;
correct if condition (could either be >1 or >0);
a recursive call to the `FACTORIAL` function itself;
parameter to recursive call reduces by 1;
correct calculation (multiplication); ***[8 marks]***

(c) A solution is:

```
function FACTDIFF(val X integer, val Y integer) result integer

     declare P integer
     declare Q integer

     if ( (X < 0) or (Y < 0) ) then
       return -1
     else
        P <-- FACTORIAL(X)
        Q <-- FACTORIAL(Y)
        if (P<Q) then
            return P - Q
          else
            return Q - P
        endif
     endif
endfunction FACTDIFF
```

*Award marks as follows up to **[5 marks]**.*

ensuring X and Y are acceptable values;
suitable return value on error (essentially any negative number is expected here, NOT 0);
calling `FACTORIAL` to calculate P and Q correctly;
ensuring that a positive value is returned ***[2 marks]***, ***[1 mark]*** for a worthy attempt. ***[5 marks]***

(d) *Award **[1 mark]** for each of the following* up to ***[3 marks]***.

Since 0!~ is the same as 1!;
The values of 0 can be found by referring to 1;
Which saves space;
`FACTDIFF (0,5)` found by looking up `FACTDIFF (1,5)` ***[3 marks]***

(e) A possible solution is:

```
function FACTDIFF(val X integer, val Y integer) result integer
/* X and Y are pass by value integers between 1 and 9
The function returns the absolute difference between the FACTORIALS  */

  if ( (X < 0) or (Y < 0) )
    return -1
  else
       if X = 0 then X = 1
       if Y = 0 then Y = 1
    if (X > Y) then
       return LOOKUP[X, Y]
      else
       return LOOKUP[Y, X]
    endif
  endif
endfunction FACTDIFF
```

*Award **[1 mark]** for each up to **[5 marks]**.*
correct test that X and Y are non-negative;
return of suitable error value (negative, not 0);
Adjust if X or Y = 0;
Determine which is larger, X or Y;
correct return of value from `LOOKUP` array; ***[5 marks]***

(f)

Make two comparisons.
*Award up to **[2 marks]** for each correctly made comparison; **[1 mark]** for point of comparison, **[1 mark]** for correct statement about (c), **[1 mark]** for correct statement about (e) – related to efficiency of storage or time.*

For example:

Algorithm (c) has a longer running time/running time of O(N) due to the need to calculate the factorials each time whereas algorithm (e) just needs a lookup/O(1) time.
[1 mark] for comparison – running time, ***[1 mark]*** for point about (c), ***[1 mark]*** for point about (e).

Algorithm (c) may be slower in looking up the value but (e) requires the table to be filled first which also takes time/is an $O(N^2)$ algorithm.
Algorithm (c) requires less memory usage, algorithm (e) requires storage for a 2D array of integer values (in RAM). ***[4 marks max]***

(g) $O(n^2)$; ***[1 mark]***
Accept $O(n^3)$.

(h) The condition is incorrect;
It should be, `if(COL > ROW)`; ***[2 marks]***

2. (a) *Award **[1 mark]** for each of the following points, to a maximum of **[2 mark]**.*

The machines have RAM which stores the program/can be programmed;
And the program can be changed;

(b) *Award **[2 marks]** for a valid comparison, **[1 mark]** for a weak answer with some credit (or where no comparison is made), maximum 2 comparisons for **[4 marks]** (ie) 4 weak answers only gets **[2 marks]**.* ***[4 marks]***

For example:

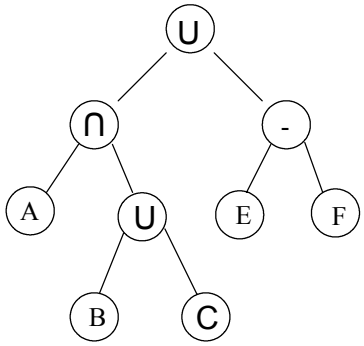
NC tools can control the tools with greater accuracy/precision;
NC tools can be manipulated in more directions simultaneously, humans limited to one or two;
Humans subject to fatigue/need rests/need holidays machines need only periodic maintenance;
etc.

(c) *Award **[1 mark]** for a valid example, **[1 mark]** for explanation * 2 to a maximum of 4 examples*: ***[4 marks]***

LAMA; has reserved words specific to machine movements UNLIKE other languages;
NC/GM/G and M codes; the codes are specific to machine tool operations AND will not be found in other HL languages;

Note that the comparison with other HLL's must be made explicitly for the second mark.

(d) (i)



*Award **[2 marks max]**, **[1 mark]** for correct tree root, **[1 mark]** for correct left subtree, **[1 mark]** right subtree.*

(ii) A B C ∪ ∩ E F - ∪ ***[2 marks]***
*Award **[2 marks]**, **[1 mark]** only if there is 1 error.*

(e) (i) A star topology;

Previous designs can be held on server;
Accessible to all designers;
Outside link to Internet for real world objects;
Direct link to manufacturing; ***[3 marks max]***

(ii) Bus topology; ***[1 mark]***

(iii) Fibre optic cable since fast transmission would give less data loss/more reliable/
Accept other justified answers. ***[2 marks]***

(f) (i) *Award **[1 mark]** for an advantage, **[1 mark]** for a disadvantage, **[1 mark]** for discussion.* ***[3 marks max]***

Reduced cost; and increased quality in the manufacture of cars/jeans/trainers;
Technology; You need a computer and modem to use an FMS to request a personalised product design;

(ii) *Award **[1 mark]** for an advantage, **[1 mark]** for a disadvantage, **[1 mark]** for discussion.* ***[3 marks max]***

Reduced cost; and increased quality in the manufacture of cars/jeans/trainers;
Technology; You need large amounts of money to set up an FMS.

(g) *Award **[1 mark]** for correct identification, **[1 mark]** for explanation*:

(i) online;
the tool moves when each command is given; ***[2 marks]***
Accept batch if justified.

(ii) real time;
the tool has to respond immediately to feedback on its positioning; ***[2 marks]***

3. (a) *Award **[1 mark]** for each of the following points up to **[3 mark max]**.*

A record/object/class structure is used;
there are three fields/data members;
Two of the fields are pointers/references to other records/objects/classes of the same type;
One field is a String field;
One field holds the data item;

(b) *Award **[1 mark]** for an advantage, **[1 mark]** for further explanation, up to **[4 marks max]**.*

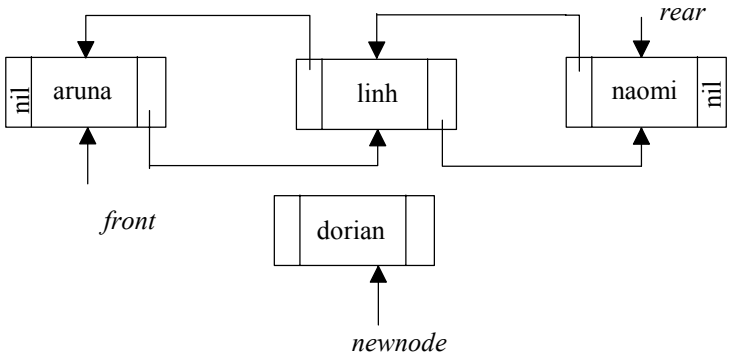
For examples:

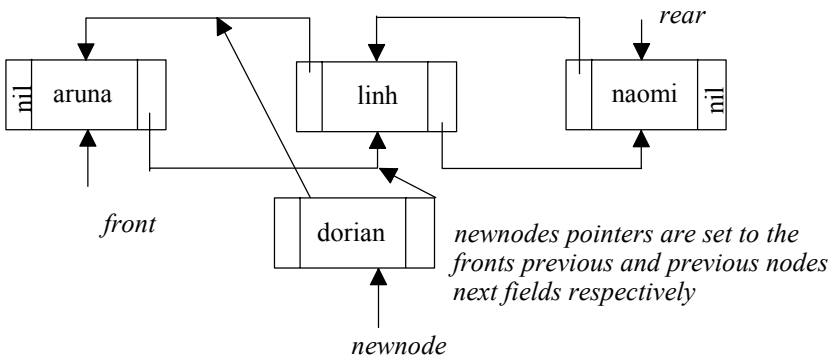
Traversal; with a singly linked list you can traverse the list in only one direction/single list is serial/sequential access;
Additions; It is a simple/O(1) operation to add at the tail unlike with singly-linked lists which take O(n) time;
Output/processing; Easier to process/print the list in reverse order compared with single list/can be done without recursion;
Programming: it is easier to deal with one pointer rather than two;

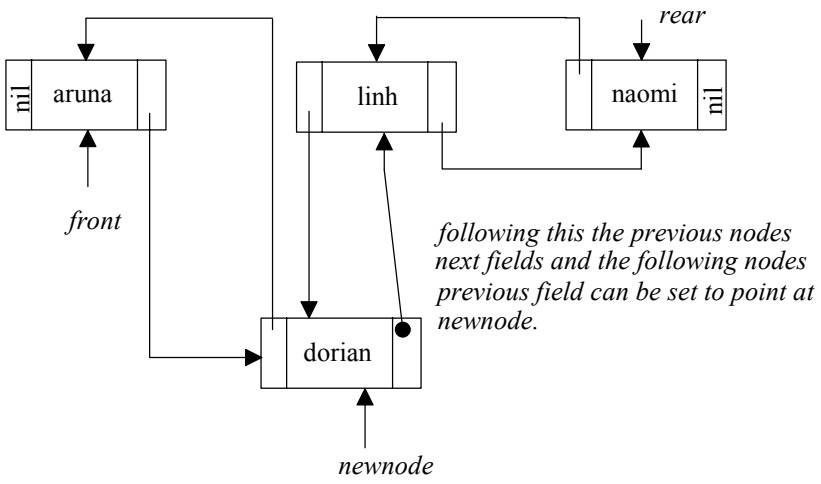
(c) *Award **[1 mark]** for each of the following points in the correct order, up to **[6 marks max]**.*

while not at end/current next = nil;
set current to current next;
if newnode is less than current;
(add here, in front of current, (current prev) points to a node)
set (current prev) next to newnode;
set newnode previous to (current prev) next;
set newnode next to current;
set current prev to newnode;

OR drawing below:

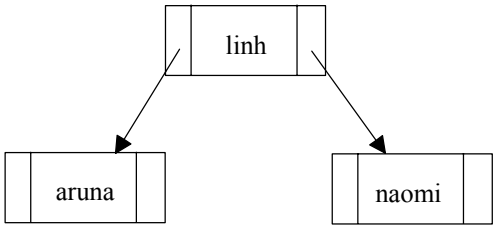






General descriptions are also acceptable if understandable and correctly ordered.

(d)



*Award **[2 mark max]**, **[1 mark]** for error.*
linh as the root pointers;
left to aruna and right to naomi;

4. (a) *Award **[1 mark]** for any of the following up to **[3 marks max]**.*

handles program execution;
monitors/manages i/o;
manages memory;
manages the file system;
manages peripherals;
manages security;
provides a user interface;
handles errors;
etc;

(b) *Award **[1 mark]** for any of the following up to **[3 marks max]**.*

The system allocates time/timeslice;
to each process/user in turn;
appears the same time because process of switching is fast (compared to users processes);

(c) *Award **[1 mark]** for any of the following up to **[2 marks]**.*

virtual memory is the use of disk space as an extension of RAM;
a swap file is kept on disc and pages/memory is temporarily stored there (until required);

(d) *Award **[1 mark]** for an issue and **[1 mark]** for elaboration up to **[4 marks max]**.*

password choice; could have used a long/not dictionary word/frequently changed;
password care; should not write password down anywhere;
physical security; lock machine/keyboard/room while not at workstation;
Do not accept answers relating to viruses unless a Trojan Horse type is explicitly mentioned or described.

For example:

Care on the Internet; do not (ever) type your username/password in to a box that pops up on your screen;

Some other nefarious methods may have surfaced after this mark scheme was completed please check with your senior examiner.

(e) *Award **[1 mark]** for any of the following up to **[3 marks max]**.*

program needs to be compiled;
to machine code;
and linked to library modules;
then loaded into memory;

5. (a) *Award **[1 mark]** for each of the following up to **[4 marks max]**.*

mid point of current search area is calculated;
record at mid point compared to wanted;
if wanted is less than mid, top half is discarded;
else bottom half is discarded;
process repeats until wanted is found;
or no records in search area;

(b) *Award **[1 mark]** for each of the following up to **[3 marks max]**.*

there is a separate index;
the index points to a group of records;
the group is searched;
for the required record;

(c) *Award **[1 mark]** for each of the following up to **[3 marks max]**.*

A combination of sorting and merging could be used;
smaller parts of the file could be sorted;
the sorted parts could be merged together;
the whole file does not need to be in memory during the merge;

(d) *Award **[1 mark]** for each of the following up to **[2 marks max]**.*

A full index could be kept;
on each of the fields;
the index contains the records position in the file;
which can thus be retrieved directly;

(e) *Award **[1 mark]** for each of the following up to **[3 marks max]**.*

The list of ID's is split into blocks;
the id of the record at the start of each block is kept in an index;
the index is therefore much smaller than the total number of records;
the index is searched first;
yielding a position in the file;
where sequential search (of the block) can start from;