INTERNATIONAL BACCALAUREATE
BACCALAURÉAT INTERNATIONAL
BACHILLERATO INTERNACIONAL

N01/650/H(1)M A

MARKSCHEME

November 2001

COMPUTER SCIENCE

Higher Level

Paper 1

12 pages

SECTION A

1. *Award marks as allocated up to a maximum of **[4 marks]**.*

*Award **[2 marks]** for any **two** of the following user documentations.*
instructions on how to load program;
how to input data;
functions that the program can perform;
output to expect from program;
help files;

*Award **[2 marks]** for any **two** of the following system documentations.*
system flowchart;
variable listing / record and tables listing;
annotated listing of code;
details of algorithms used;
requirements definition;
software specifications;
test plan *etc.*;

2. *Award marks as allocated up to a maximum of **[4 marks]**.*

*Award **[2 marks]** for any **two** of the following local variables.*
defined within a procedure or subroutine;
no effect outside that procedure;
any changes do not affect the rest of the program;
stored on stack;

*Award **[2 marks]** for any **two** of the following global variables.*
declared in the main body of the program;
can be used and changed in any part of the program;
any changes made anywhere are carried through to the rest of the program;
stored in global memory space;

3. (a) *Award **[1 mark]** maximum for any suitable input device:*
most likely is a voice recognition device but accept device that can be touched (such a large push button, chord) provided it is made clear that there are many and are available from all parts of the apartment;

(b) *Award **[1 mark]** maximum for any suitable output device:*
most likely is again sound but could be flashing light;

4. *Award **[1 mark]** for any of the following, up to a maximum of **[3 marks]**.*
syntax is the grammar of a programming language;
or set of rules that have to be followed;
for example every **`begin`** must have an **`end`**;
a translator checks the syntax by applying the rules;
if rule broken the program stops (in the case of an interpreter) or is reported;

5. *Award **[1 mark]** for each of the following steps, up to a maximum of **[4 marks]***
instruction loaded into IR;
address found from IP;
instruction decoded;
instruction executed;
IP incremented;
or
fetch
decode
execute
store

6. *Award **[1 mark]** for each of the following, up to a maximum of **[6 marks]**.*
*Award **[1 mark]** for each activity and **[1 mark]** for the explanation.*
point for feature;
point for explanation;

periodic reviews, to give state of system;
evaluate, way in which system works;
modify, according to needs;
document changes;

7. *Award marks as allocated, up to a maximum of **[4 marks]**.*
fragmentation when files split over disk after many deletes and saves ***[2 marks]***;
utility software can defragment the disk ***[1 mark]***;
by reorganising and adjusting index of addresses ***[2 marks]***;

8. *Award marks as allocated, up to a maximum of **[4 marks]**.*
validation means checking input against reasonable values ***[1 mark]*** by software ***[1 mark]***.
verification means repeating data entry and checking the two are the same ***[2 marks]***.

9. (a) a stack is a First In Last Out (FILO) structure; ***[1 mark]***

(b) *Award **[1 mark]** for each of the following, up to a maximum of **[4 marks]**.*
when procedure called;
return address, push on stack;
local variables stored;
when completed address, poped from stack;
repeated calls means successive address put on and taken off in reverse order;

10. *Award **[1 mark]** for each of the following, up to a maximum of **[4 marks]**.*
protocol is a set of rules and procedures;
followed when transmitting packets of data;
part of this is to send information about the packet;
such as destination;
with the packet;
so that the same protocol can be interpreted at the other end when unpacking;

SECTION B

11. (a) Bubble Sort or Exchange Sort; ***[1 mark]***

(b) array of string or array of 5 characters; ***[1 mark]***

(c)
```
procedure ALPHA(val N integer, ref LETTER string array (1..26))
   declare TEMP, COUNT1,COUNT2 integer
      for COUNT1<-- 1 upto N-1 do
      for COUNT2<-- COUNT1+1 upto N do
      if LETTER(COUNT1)>LETTER(COUNT2)
      then TEMP<--LETTER(COUNT1)
         LETTER(COUNT1)<--LETTER(COUNT2)
         LETTER(COUNT2)<--TEMP
      endif
      endfor
   endfor
```

*Candidates do not need to write out all the original statements. Allocate marks as follows, up to a maximum of **[4 marks]**.*

correct declaration of parameters ***[2 marks]***;
[1 mark] if at least one is of correct type;
correct declaration of variables within procedure ***[1 mark]***;
correct change of loop terminators ***[1 mark]***;

(d) *Award marks as allocated, up to a maximum of **[4 marks]**.*
add `SWAPS` as Boolean type variable ***[1 mark]***;
set `SWAPS` to **`false`** between the two **`for`** statements and **`if`** `SWAPS=`**`true or`** COUNT1=1 **`then`** ***[1 mark]***;

add `SWAPS=true` between **`then`** and **`endif`** ***[1 mark]***;
add **`if`** `not SWAPS` **`then`**

and an extra **`endif`** at the **`end`** ***[1 mark]***;

One example is:
```
for COUNT1 <-- 1 upto N-1 do
  swaps <-- false
  for COUNT2 <-- COUNT1+1 upto N do
    if LETTER(COUNT1) > LETTER(COUNT2) then
      swaps <-- true
      ....
    endif
  endfor
  if swaps then return
endfor
```

12. (a) A, B,C represent obstacle to left, right or in front respectively.

Award marks as allocated:

[1 mark] for all 8 inputs in any order.
[1 mark] for row 1.
[1 mark] for all rows 2, 4, 6, 8 correct.
[1 mark] for both rows 3 and 5 correct.

	A	B	C	L	R		L	R
Row 1	0	0	0	1	1		1	1
Row 2	0	0	1	0	0		0	0
Row 3	0	1	0	0	1		0	1
Row 4	0	1	1	0	0	**or**	0	0
Row 5	1	0	0	1	0		1	0
Row 6	1	0	1	0	0		0	0
Row 7	1	1	0	1	1		0	0
Row 8	1	1	1	0	0		0	0

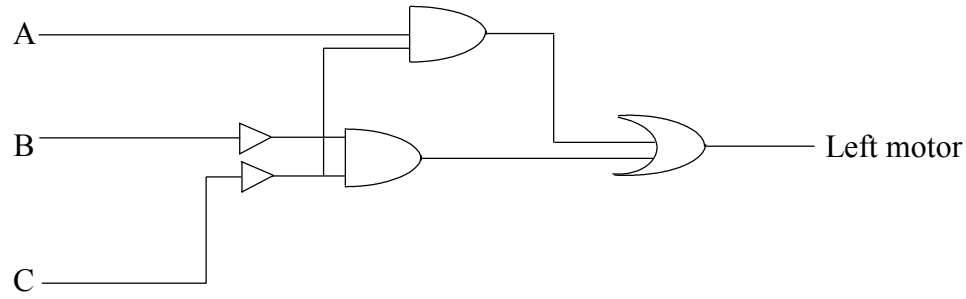
[4 marks]

(b) $\overline{A}\overline{B}C + A\overline{B}\overline{C} + AB\overline{C} = L = \overline{B}\overline{C} + A\overline{C}$;
$\overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + AB\overline{C} = R = \overline{A}\overline{C} + B\overline{C}$; ***[4 marks]***

or

$\overline{B}\overline{C}$ Left ***[2 marks]***, ***[1 mark]*** if not simplified.
$\overline{A}\overline{C}$ Right ***[2 marks]***

(c) Award ***[2 marks]*** for a diagram either for the Left or Right motors.



[2 marks]

13. (a) *Award marks as allocated, up to a maximum of **[4 marks]**.*
real time ***[1 mark]*** as airspeed is the input and the output will be fed back as input ***[1 mark]***, interactive (online) ***[1 mark]*** as user can interact with system and system with user ***[1 mark]***.

(b) *Award marks as allocated, up to a maximum of **[2 marks]**.*
[1 mark] analogue signal ***[1 mark]*** analogue / digital converter;

(c) *Award marks as allocated, up to a maximum of **[4 marks]**.*
sort transaction file ***[1 mark];***
merge translation file ***[1 mark]*** with old master file ***[1 mark]***;
to produce new master file ***[1 mark]***;

14. (a) *Award marks as allocated, up to a maximum of **[2 marks]**.*
bar code identifies item code ***[1 mark]*** related to item code on computer ***[1 mark];***
read by light detection with width of bass giving unique code number ***[1 mark];***

(b) *Award marks as allocated, up to a maximum of **[2 marks]**.*
polling: server "visits" each cash point in turn ***[1 mark]*** to see if processing needed ***[1 mark]***;
small time slices mean this seems always available ***[1 mark]***;

*Award marks as allocated, up to a maximum of **[2 marks]**.*
interrupts: when cash desk needs server ***[1 mark]*** an interrupts sent ***[1 mark]*** sensor stops current process until after cash desk processing ***[1 mark]***;

(c) *Award **[1 mark]** for each of the following, up to a maximum of **[2 marks]**.*
central processing: simple server in supermarket;
one O/S with cash desks accessing;
no processing without server;

*Award marks as allocated, up to a maximum of **[2 marks]**.*
distributed processing: each supermarket has own processing power ***[1 mark]*** linked to other servers and O/S ***[1 mark]***;

15. (a) *Award **[1 mark]** for any of the following **[4 marks]**.*
an object is a combination of data and operations;

Advantages:
encapsulation;
information and data hiding ***[1 mark]***;
object can be used at abstract level ***[1 mark]***;
can be used intuitively ***[1 mark]***;
inheritance means that one object can be derived from another without recoding ***[2 marks];***

(b) *Award **[1 mark]** for advantage and **[1 mark]** for description, up to a maximum of **[4 marks]**.*
easier to debug ***[1 mark]*** as structure evident ***[1 mark]***;
quicker ***[1 mark]*** since separate modules can be coded by separate programmers ***[1 mark]***;
individual programmers may be skilled in specific areas ***[2 marks]***;
general structure better ***[1 mark]*** easier to maintain ***[1 mark]***;

(c) *Award marks as allocated, up to a maximum of **[2 marks]**.*

*Accept any of the following for **[1 mark]**.*
in all software developed,
- original concepts will need continual review and evaluation in the light of how they meet needs;
- new features likely to be added and others modified;

*Accept any of the following for **[1 mark]**.*
- hence system installed now likely to need servicing for a long period of time.
- not all work from new clients.

Accept any other reasonable explanation that refers to continual system update. If in doubt contact the Chief Examiner.