M19/4/COMSC/HP1/ENG/TZ0/XX/M

Diploma Programme
Programme du diplôme
Programa del Diploma

Markscheme

May 2019

Computer science

Higher level

Paper 1

19 pages

Subject details: Computer science HL paper 1 markscheme

Mark allocation

Section A: Candidates are required to answer **all** questions. Total 25 marks.
Section B: Candidates are required to answer **all** questions. Total 75 marks.
Maximum total = 100 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 that part of a question.

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

Each statement worth one 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.

If the candidate's answer has the same meaning or can be clearly interpreted as being the same as that in the markscheme then award the mark.

Mark positively. Give candidates credit for what they have achieved and for what they have got correct, rather than penalizing 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. In this subject effective communication is more important than grammatical accuracy.

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**".

General guidance

Issue	Guidance
Answering more than the quantity of responses prescribed in the questions	In the case of an "identify" question, read all answers and mark positively up to the maximum marks. Disregard incorrect answers. In the case of a "describe" question, which asks for a certain number of facts *eg* "describe two kinds", mark the first two correct answers. This could include two descriptions, one description and one identification, or two identifications. In the case of an "explain" question, which asks for a specified number of explanations *eg* "explain two reasons …", mark the first two correct answers. This could include two full explanations, one explanation, one partial explanation *etc*.

Section A

1. A piece of / a hardware device that is externally connected or attached / remotely connected or attached (to the computer system);

Example answer
A peripheral is an external (computer) device that is connected to a computer such as a keyboard; **[1]**

2. *Award **[2 max]**.*
Menus;
Dialogue boxes;
Windows;
Icons;
Pointers;
Buttons;

Note to examiners: allow other correct user interface features **[2]**

3. *Award **[2 max]**.*
The MAR holds the memory location of data/instructions;
…that need to be accessed (read/write) (fetch/store); **[2]**

4. *Award **[2 max]**.*
Single processor is mainly occupied with the OS jobs / might not be able to handle multiple jobs/;
Rendering 3D graphics requires a great deal of processing (which a single processor may not be able to give);

If attempting rendering on a single processor, a (very) high clock speed is required; which may not be available;

3D graphics processing is (inherently) parallel;
A single processor is not able to handle parallel processing; **[2]**

5. (a) *Award **[2 max]**.*
Award [1] for an advantage and [1] mark for an outline.

Allows more applications to run than there is available physical memory;
By the use of page/swap files/part of hard disk as primary memory;

Larger application can run
With less real RAM; **[2]**

(b) *Award **[2 max]**.*
Award [1] for a disadvantage and [1] mark for an outline.

Applications run more slowly;
Uses hard drive memory as primary memory / takes more time to switch between applications;

When a computer's virtual memory resources are overused /Reduced amount of hard drive space available for your use;
programs lock-up/do not run/disk thrashing; **[2]**

6. DF; **[1]**

7.

A	B	C	S	Z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	1

*Award **[3 max]**.*
*Award **[3]** for all 8 correct rows.*
*Award **[2]** for 7 correct rows.*
*Award **[1]** for 5,6 correct rows.* **[3]**

8. *Award **[2 max]**.*
Surveys;
(General) questions distributed to many stakeholders as a written or online document;

Interviews;
(Specific) questions asked of nominated stakeholders in an individual setting;

Direct observations;
Observer watches stakeholders performing their current tasks; **[2]**

9. *Award **[2 max]**.*
Smallest type of network;
Consists of connected devices in close proximity to the individual using them;
Connected via Bluetooth/wireless;
Suitable example: smartphone to car connection; **[2]**

10. *Award **[2 max]**.*
To provide a set of rules/procedures;
To enable two or more different electronic devices/computers/entities to understand each other during data transfer / enable successful communication; **[2]**

11. *Award **[2]** for completely correct answer and **[1]** for any **3** numbers in correct order.*
Postorder traversal: 76 75 79 70 68 72 83 **[2]**

12. *Award **[2 max]**.*
Time efficient searching / traversing for a contact in an address book;
Each iteration / comparison allows the size of the search to be reduced (by skipping about half of the remaining contacts);

Fast/easy addition / removal of contacts in an address book;
Quick search for the leaf node / empty node where a new contact can be placed / for the node containing the contact to be deleted;

Contacts can be listed / output in alphabetical order/ fast sorting;
using inorder traversal; **[2]**

Section B

13. (a) *Award **[4 max]**.*
*Mark as **[2]** and **[2]**.*

Malicious activities;
An unauthorized user gaining access to data and deleting/altering it;

Natural disasters / earthquake / storm / power loss;
Causing the system to crash and destroy data;

Malware/viruses/spyware/worms;
Which infiltrate and damage the data;

Human error;
Accidental deletion/overwriting of files; **[4]**

(b) *Award **[4 max]**.*
*Mark as **[2]** and **[2]**.*

Copies of backup could be kept off-site/cloud backup;
Unlikely that the other site would be affected by the natural disaster/can be reloaded/reinstalled if needed;

Incremental backup only backs up data that has changed;
Therefore, requiring less storage capacity / can be completed more quickly than a complete backup;

Failover system/mirrored system/disk mirroring;
A duplicate copy to be used in the event the main system fails; **[4]**

(c) *Award **[3 max]**.*
Testers outside the organization use the operating system in a “real world” setting;
Enables feedback to be given to the developers;
So that the software can be improved/corrected/debugged;
Before it is finally released;
“Real world” testers may find more bugs as the system is used in ways not originally intended / tested; **[3]**

(d) *Award **[2 max]**.*
The software may not work as expected / may not be better than the existing software / may not meet user requirements / expectations;
The software may be missing some key features;
The software may not be user friendly; **[2]**

(e) *Award **[2 max]**.*
Touch screens;
Voice recognition;
Text-to-speech;
Braille keyboard
A colour-blind option
Large font option;

Note to examiners: allow other correct accessibility features **[2]**

14. (a) *Award **[2 max]**.*
RAM is volatile/ contents erased when power is switched off;
Access speed is fast / faster than hard drive;
Data / instructions can be read from and written to it//RAM can be overwritten;
Size is limited; **[2]**

(b) To store programs / files / data in a non-volatile device so it isn't lost;
Stores more data as it has a larger capacity; **[1]**

(c) *Award **[2 max]**.*
Word processor;
Spreadsheet;
Database management system;
Email;
Web browser;

Allow any general purpose application that is appropriate for a 'training room computer' **[2]**

(d) (i) *Award **[2 max]**.*
May save money;
Due to not having to supply all the training computers;

May be able to increase the size of the training group;
Which may generate more income;

Trainees / teachers likely to be more familiar with software on own machine (and how new training software interacts with OS / user interface);
Making training sessions more efficient / allowing trainer to concentrate on the training rather than using generic applications; **[2]**

(ii) *Award **[2 max]**.*
May cause security issues;
Due to multiple users having network access from their "unsecured" devices;

May interfere with running of training sessions;
As some machines may not be compatible; **[2]**

(e) *Award **[2 max]**.*
Encryption;
Scrambles the contents of the network transmissions so that if they are intercepted they can't be understood (without the decryption key);

User ID (and password);
Only allows authorized users to access the network;

Media Access Control (MAC) addresses;
Unique identification codes embedded in networkable equipment so that only authorized equipment may access the network;

Firewall;
Checks traffic coming into the network and leaving the network, and can block suspicious data; **[2]**

(f) *Award **[4 max]**.*
*Mark as **[2]** and **[2]**.*

The speed of data transmission (on a wireless network) slows down;
The further the receiver is from the transmitter;

Passing through obstructions such as solid walls;
Can slow down transmissions (on a wireless network);

The bandwidth available for transmission on a wireless network is finite;
So, transmission speeds can be affected if the number of users on the network increases;

Note to examiners: Answers must relate to wireless networks and not be a comparison between cabled and wireless networks. **[4]**

15. (a)

			Array VALUES[]					
COUNTER1	**MINIMUM**	**COUNTER2**	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**TEMPORARY**
0	0	1	20	6	38	50	40	
	1	2						
		3						
		4						6
				20				
			6					
1	1	2						
		3						
		4						
2	2	3						
		4						
3	3	4						
	4							40
							50	
						40		

*Award **[5 max]**.*
Both `COUNTER1` and `COUNTER2` correct;
`MINIMUM` column correct;
Final `VALUES[]` 0, 1, 2 correct;
Final `VALUES[]` 3, 4 correct;
`TEMPORARY` column correct;

Note to examiners:
Allow follow through (FT).
In case of different representation of values in columns COUNTER1 and COUNTER2, then FT, award marks for the correct values of the variables MINIMUM and TEMPORARY. **[5]**

(b) (i) *Award **[3 max]**.*
Use of correct nested loops;
Correct use of flag;
Inner loop checking adjacent cells;
Values being swapped if necessary;

Example algorithm 1:

```
LIMIT = 4
FLAG = TRUE
loop while FLAG = TRUE
  FLAG = FALSE
  loop COUNTER from 0 to LIMIT - 1
    if VALUES[COUNTER] > VALUES[COUNTER + 1] then
      TEMPORARY = VALUES[COUNTER]
      VALUES[COUNTER] = VALUES[COUNTER + 1]
      VALUES[COUNTER + 1] = TEMPORARY
      FLAG = TRUE
    end if
  end loop
end loop
```

A recursive solution is allowed at HL. SL candidates who submit an above level recursive solution should also receive credit.

Version 1 – basic recursive solution

*Award **[3 max]**.*
BUBBLESORT defined as a procedure with correct pass through parameters and end/return statement;
Correct loop with values swapped if necessary inside procedure;
Recursive call of BUBBLESORT with parameters passed;
Correct condition for recursive call;

Example algorithm 2

```
LIMIT = 4
BUBBLESORT(VALUES, LIMIT)
    loop COUNTER from 0 to LIMIT - 1
      if VALUES[COUNTER] > VALUES[COUNTER + 1] then
        TEMPORARY = VALUES[COUNTER]
        VALUES[COUNTER] = VALUES[COUNTER + 1]
        VALUES[COUNTER + 1] = TEMPORARY
    end if
  end loop
  if LIMIT - 1 > 1 then
    call BUBBLESORT(VALUES, LIMIT - 1)
  end if
end BUBBLESORT
```

Version 2 – more efficient recursive solution

*Award **[3 max]**.*
BUBBLESORT defined as a procedure with correct pass through parameters and end/return statement;
Correct loop with values swapped if necessary inside procedure;
Recursive call of BUBBLESORT with parameters passed;
Correct condition for recursive call;
Correct use of flag;

Example algorithm 2

```
LIMIT = 4
BUBBLESORT(VALUES, LIMIT)
    FLAG = FALSE
    loop COUNTER from 0 to LIMIT - 1
      if VALUES[COUNTER] > VALUES[COUNTER + 1] then
        TEMPORARY = VALUES[COUNTER]
        VALUES[COUNTER] = VALUES[COUNTER + 1]
        VALUES[COUNTER + 1] = TEMPORARY
        FLAG = TRUE
      end if
    end loop
    if LIMIT - 1 > 1 and FLAG = TRUE then
      call BUBBLESORT(VALUES, LIMIT - 1)
    end if
end BUBBLESORT
```

[3]

(ii) Bubblesort; **[1]**

(c) *Award **[2 max]**.*
Use of (any type of) loop;
Correct output statement;

Example algorithm:

```
loop COUNTER from 0 to LIMIT
  output VALUES[COUNTER]
end loop
```

[2]

(d) *Award **[3 max]**.*
Award ***[1]*** for stating a feature of a static/dynamic data structure
*Award **[1]** for stating a difference to dynamic/static data structure*
*Award **[1]** for an explanation*

Memory is allocated to the static data structure in compile time (a pre-set amount of memory/fixed size);
Memory is allocated to the dynamic data structure at run-time;
The size of a static data structure can never change during run-time whilst the size of dynamic data structure can be randomly updated during run time which makes efficient use of RAM;

Memory allocated to the static data structure is always available in order to be used by the data structure and elements can be accessed either directly or sequentially;
Whilst direct access to elements in the dynamic data structure is not allowed and elements should be accessed sequentially;
So in some/most cases algorithms which use dynamic data structures are slower (during execution) than algorithms with static data structures;

[3]

(e) *Award **[1 max]**.*
Linked list
Collection (for example, ArrayList);
Binary tree;

[1]

16. (a) *Award **[2 max]**.*
Managing memory;
Managing peripherals;
Processor management;
Scheduling;

Note to examiners: allow any valid feature of an operating system **[2]**

(b) *Award **[3 max]**.*

Example 1
A dedicated operating system for a mobile phone will take up less storage space than a full-sized operating system;
This will allow the device to function more quickly;
Because it doesn't contain features that aren't needed;

Example 2
A dedicated operating system for a mobile phone can be customized;
Benefits the end users;
As they deal with a familiar interface;

Example 3
The dedicated OS is designed specifically for the mobile phone (hardware equipment);
This avoids compatibility issues;
While a generic operating system is designed for multiple types of hardware (which can lead to compatibility / various issues);

[3]

(c) *Award **[5 max]**.*
A desired temperature is input/pre-set by the user;
Sensors detect temperature;
And regularly/continuously send (temperature) readings to the microprocessor;
The microprocessor compares the actual readings with the pre-set (by the user) / input temperature;
If the temperature is too hot/too cold the microprocessor sends signal to actuator;
To adjust temperature (no need for complex details regarding actions of a heating/cooling system);

*Note to examiners: Award **[1]** for evidence of the use of analogue-to-digital converter (sensors) and/or digital-to-analogue convertor(actuators).* **[5]**

(d) *Award **[3 max]**.*
Award [1] for each contrast/comparison, up to [3].

Centralized system allows control at a central point, which is then transmitted to the various rooms whilst distributed system allows individual control of settings at each unit / room so is more flexible // Centralized system will have same or constant temperature throughout the home whilst distributed system allows individual control of setting at each room;

Easier to control settings centrally whilst distributed system is more complicated to control remotely (due to limited / no connectivity of the control system);

Centralized system is more difficult to install due to the connectivity required between the various components whilst distributed system is easier to install due to fewer connection issues;

If a computer system / or any connection fails in the centralized system, the whole system is not able to function correctly whilst the distributed system as whole would still function correctly;

Centralized system is cheaper / has lower operational cost whilst distributed system is expensive as it requires additional hardware / software;

Centralized systems are difficult to expand whilst distributed systems are easily expandable because self-sufficient systems can be added or removed at any point in time without affecting the overall system; **[3]**

17. (a) *Award **[5 max]**.*
Initialization of TOTAL before loop and output of result after loop;
Use of any loop with correct limits;
Correct row and column subscripts of two-dimensional array;
Correct comparison in if statement;
Totalling within if statement, if statement inside loop;

Note to examiners: Do not accept flowcharts.

Example algorithm 1:

```
SEARCH = "Cardiff"
TOTAL = 0

loop COUNTER from 0 to 511
  if SEARCH = CUSTOMERS[COUNTER][5] then
    TOTAL = TOTAL + 1
  end if
end loop

output "The number of customers is ", TOTAL
```

Example algorithm 2

```
TOTAL = 0
loop COUNTER from 0 to CUSTOMERS.length()-1
  if  CUSTOMERS[COUNTER][5].equals( "Cardiff")
        //accept CUSTOMERS[COUNTER][5]== "Cardiff"
        then TOTAL = TOTAL + 1
  end if
end loop
output  TOTAL
```

Note to examiners:
The array structure is not given in the question.

If array subscripts begin with 1,
then award 1 mark for
`loop COUNTER from 1 to 512`
and 1 mark for
`CUSTOMERS[COUNTER][6]` **[5]**

(b) *Award **[5 max]**.*
Use of any loop with correct limits;
Correct row and column subscripts of two-dimensional array;
Correct comparison in if statement
Output of all row elements(fields)
Output correctly formatted

Note to examiners: Accept flowcharts.

Example algorithm 1:

```
SEARCH1 = "Jones"
SEARCH2 = "Cardiff"
loop COUNTER from 0 to 511
  if CUSTOMERS[COUNTER][0] = SEARCH1 then
    if CUSTOMERS[COUNTER][5] = SEARCH2 then
      output CUSTOMERS[COUNTER][1] " " CUSTOMERS[COUNTER][0]
      output CUSTOMERS[COUNTER][2]
      output CUSTOMERS[COUNTER][3]
      output CUSTOMERS[COUNTER][4]
      output CUSTOMERS[COUNTER][5]
      output CUSTOMERS[COUNTER][6]
    end if
  end if
end loop
```

Example algorithm 2:

```
R=0
loop while R <= 511
  if CUSTOMERS[R][0]=="Jones" and CUSTOMERS[R][5]== "Cardiff"
     then
       output CUSTOMERS[R][1], CUSTOMERS[R][0]
       loop K from 2 to 6
         output CUSTOMERS[R][K]
       end loop
     end if
  R=R+1
end loop
```

Note to examiners:
Since "Jones" can be either a first or last name, accept

```
CUSTOMERS[COUNT][1] == SEARCH1
```

and the first output line

```
output CUSTOMERS[COUNT][0], CUSTOMERS[COUNT][1]
```

[5]

(c) *Award **[5 max]**.*
Accept correct answers given as diagrams or written in text applying the following marking points:

*Award **[5 max]** if candidate explained the steps to insert "`Jones`" into sorted list at correct place.*
*Award **[1]** for each of the following steps, up to [5]*

Create a new node with data field `Jones` and pointer field;
Start searching from the beginning of the list;
Find the location/position where the new node is to be inserted (`Jones` to be inserted after `Davies`);
Set the pointer in the new node (containing Jones) to the pointer in the node containing `Davies` (to point to the node containing `Pugh`);
Set the pointer in the node containing `Davies` to point to the new node (`Jones`);

*Award **[3 max]** if candidate explained only the steps to insert "`Jones`" at the end of the list.*
*Award **[1]** for each of the following steps*

Create a new node with data field `Jones` and pointer field NIL;
Start searching from the beginning of the list to find the last node (`Williams`);
Set the pointer in the last node (containing `Williams`) to point to the new node (`Jones`);

*Award **[3 max]** if candidate explained only the steps to insert "Jones" at the beginning of the list.*
*Award **[1]** for each of the following steps, up to **[3]***

Create a new node with data field `Jones` and pointer field;
Set the pointer in the new node (containing `Jones`) to the external pointer (which points to the beginning of the list/to the first node in the list (`Bale`));
Set the external pointer to point to the new node (`Jones`); **[5]**

(d) *Award **[2 max]**.*
Award marks for the use of circular lists (rather than linked lists) in an application (accept examples);
in which any node can be a starting point / which traverses the whole list by starting from any point;
to repeatedly go around the list;

Used to allow multiple applications to run in a PC;
The operating system cycles through each one in time giving each a slice of time to execute;

Used for the implementation of a (circular) queue;
The end of the queue points to the beginning, eliminating the need to maintain both front and rear pointers;

Used in multiplayer gaming environment;
The OS cycles through one player at a time using time slicing;

A media playlist that repeats (endlessly);
The last song (node) points to the first song;

[2]