M19/4/COMSC/SP1/ENG/TZ0/XX/M

Diploma Programme
Programme du diplôme
Programa del Diploma

Markscheme

May 2019

Computer science

Standard level

Paper 1

12 pages

Subject details: Computer science SL 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 45 marks.
Maximum total = 70 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 hardware / 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]**.*
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]**

4. *Award **[2 max]**.*
To provide feedback on the efficiency or design of the product;
To give an idea or feel of the final product;
To encourage dialogue between the developers and the client;
Clients can identify errors or omissions in the design; **[2]**

5. DF; **[1]**

6.

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]**

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

8. Arithmetic and Logic Unit/ALU; **[1]**

9. *Award **[2 max]**.*
Pre-processing;
Lexical analysis;
Parsing;
Semantic analysis;
Syntax analysis;
Linking;
Optimisation; **[2]**

10. *Award **[2 max]**.*
On a CD/DVD/Blu-ray disc;
Through embedded help files;
Online/website;
Physical/printed media;

Note to examiners: accept answers relating to methods of supplying or media used to present user documentation. Do not accept answers related to training. **[2]**

11. *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]**

12. *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]**

13. *Award **[3 max].***
A message/the data is broken into a number of parts;
Which are sent independently;
…over the optimum route for each packet;
The individual parts are reassembled at the destination;
Each packet contains the (IP) address of both the sender and recipient; **[3]**

Section B

14. (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; **[2]**

Note to examiners: allow other correct accessibility features

15. (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]**

16. (a) *Award **[1]** for any **two** from:*
LIMIT;
MINIMUM;
COUNTER1;
COUNTER2;
TEMPORARY. **[1]**

(b)

			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]**

(c) (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]**

(d) *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]

(e) *Award **[3 max]**.*
Sub-programs contain reusable code (for use in other programs);
A large programming project can be divided into sub-programs;
Different sub-programs can be constructed by different programmers;
Future maintenance is easier due to the organisation of the large program (into sub-programs); **[3]**