Cambridge Assessment International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **October/November 2021**

MARK SCHEME

Maximum Mark: 75

Published

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the October/November 2021 series for most Cambridge IGCSE™, Cambridge International A and AS Level components and some Cambridge O Level components.

Generic Marking Principles

These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptors for a question. Each question paper and mark scheme will also comply with these marking principles.

GENERIC MARKING PRINCIPLE 1:

Marks must be awarded in line with:

- the specific content of the mark scheme or the generic level descriptors for the question
- the specific skills defined in the mark scheme or in the generic level descriptors for the question
- the standard of response required by a candidate as exemplified by the standardisation scripts.

GENERIC MARKING PRINCIPLE 2:

Marks awarded are always **whole marks** (not half marks, or other fractions).

GENERIC MARKING PRINCIPLE 3:

Marks must be awarded **positively**:

- marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate
- marks are awarded when candidates clearly demonstrate what they know and can do
- marks are not deducted for errors
- marks are not deducted for omissions
- answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous.

GENERIC MARKING PRINCIPLE 4:

Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.

GENERIC MARKING PRINCIPLE 5:

Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen).

GENERIC MARKING PRINCIPLE 6:

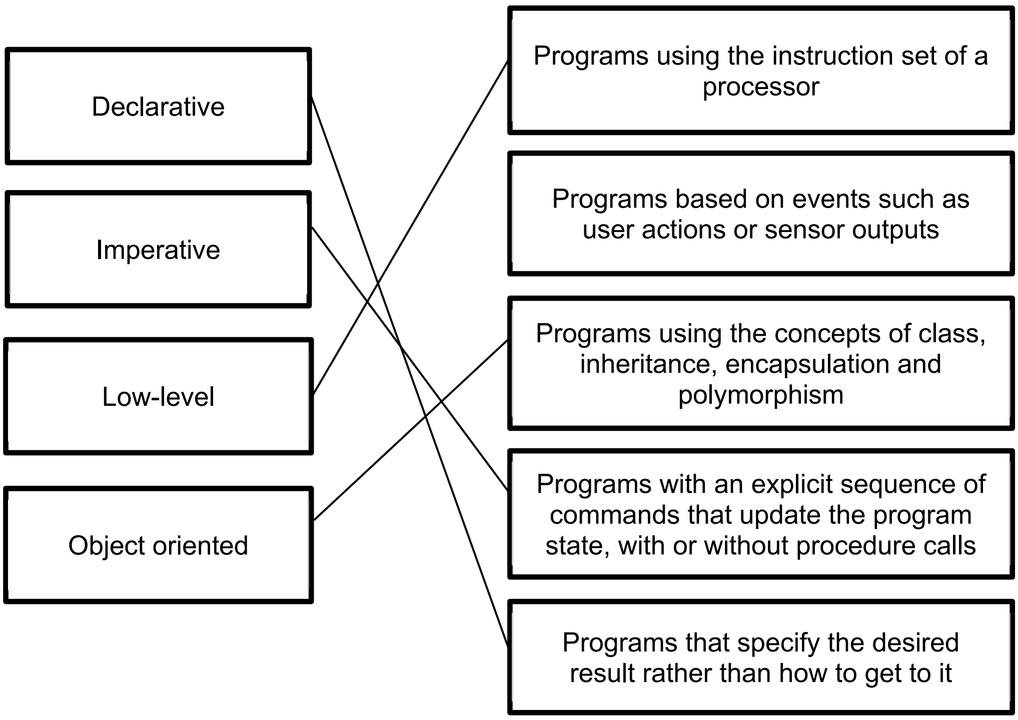
Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind.

Question	Answer	Marks
1(a)(i)	**One** mark for each correct marking point (**Max 2)** • 010111000110 (correct mantissa) • 0111 (correct exponent)	**2**
1(a)(ii)	**One** mark for each correct consequence **One** mark for each correct justification **Consequence** • The precision/accuracy of the number would be reduced **Justification** • … because the least significant bits of the original number have been truncated/lost // the original number had 13 bits / 14 bits with sign but the mantissa can only store 12 bits	**2**
1(b)	**One** mark for each correct marking point (**Max 3)** • To store the maximum range of numbers in the minimum number of bytes / bits • Normalisation minimises the number of leading zeros/ones represented • Maximising the number of significant bits // maximising the (potential) precision / accuracy of the number for the given number of bits • … enables very large / small numbers to be stored with accuracy. • Avoids the possibility of many numbers having multiple representations.	**3**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>2</td><td>

One mark for each **single** correct line from **Programming Paradigm** to **Description**

Programming Paradigm **Description**



</td><td>4</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>3(a)</td><td>

One mark for each marking point (**Max 2**)

- `TYPE Parts =`
- `(Monitor, CPU, SSD, HDD, LaserPrinter, Keyboard, Mouse)`

Complete answer

```
TYPE Parts = (Monitor, CPU, SSD, HDD, LaserPrinter,
Keyboard, Mouse)
```

</td><td>2</td></tr>
<tr><td>3(b)</td><td>

One mark for each marking point (**Max 2**)

- `TYPE SelectParts = ^`
- correct data type chosen `Parts`

Complete answer

```
TYPE SelectParts = ^Parts
```

</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>One mark for each marking point (Max 2)

• <character>::=
• $|%|&|*|#

Complete answer
<character>::= $|%|&|*|#</td><td>2</td></tr>
<tr><td>4(b)(i)</td><td>For example: $A9E3</td><td>1</td></tr>
<tr><td>4(b)(ii)</td><td>One mark for each marking point (Max 4)

• <password>::=<character> …
• … <code>

• <code>::= …
• … <digit>|<capital_letter>
• … |<digit><code>|<capital_letter><code>

Complete answer
<password>::=<character><code>

<code>::=<digit>|<capital_letter>|<digit><code>|<capital_
letter><code></td><td>4</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>One mark for each correct marking point (Max 4)

• In both serial and sequential files records are stored one after the other …
• … and need to be accessed one after the other
• Serial files are stored in chronological order
• Sequential files are stored with ordered records
• … and stored in the order of the key field
• In serial files, new records are added in the next available space / records are appended to the file
• In sequential files, new records are inserted in the correct position.</td><td>4</td></tr>
<tr><td>5(b)</td><td>Direct (access)</td><td>1</td></tr>
<tr><td>5(c)</td><td>Sequential (access)</td><td>1</td></tr>
</table>

Question	Answer	Marks
6(a)	**One** mark for each correct marking point **(Max 5)** • A large message is divided up into a group of smaller chunks of the same size called **packets** • The packet has a header and a payload • The header contains a source **IP** address, destination **IP** address (and sequence number) • Each packet is dispatched independently • … and may travel along different routes / paths • The packets may arrive out of order • … and are reassembled into the original message at the destination • If packets are missing / corrupted a re-transmission request is sent.	**5**
6(b)	**One** mark for each correct marking point **(Max 3)** • The router examines the packet's header • It reads the IP address of the destination (from the packet header) • A router has access to a routing table • …containing information about, e.g., available hops / netmask / gateway used • … and the status of the routes along the route • … the router decides on the next hop / best route • … and sends the packet on its next hop.	**3**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>7(a)</td><td>One mark per two correct products (Max 3)

(Z =) $A\ \overline{B}\ \overline{C}\ D + A\ \overline{B}\ C\ D + A\ B\ \overline{C}\ \overline{D} + A\ B\ \overline{C}\ D + A\ B\ C\ \overline{D} + A\ B\ C\ D$</td><td>3</td></tr>
<tr><td>7(b)(i)</td><td>One mark for every two correct rows or columns (Max 2)

<table>
<tr><td></td><td></td><td colspan="4">AB</td></tr>
<tr><td></td><td></td><td>00</td><td>01</td><td>11</td><td>10</td></tr>
<tr><td rowspan="4">CD</td><td>00</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>01</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>11</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>10</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
</table></td><td>2</td></tr>
<tr><td>7(b)(ii)</td><td>One mark for correct loop (Max 2)

<table>
<tr><td></td><td></td><td colspan="4">AB</td></tr>
<tr><td></td><td></td><td>00</td><td>01</td><td>11</td><td>10</td></tr>
<tr><td rowspan="4">CD</td><td>00</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>01</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>11</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>01</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
</table></td><td>2</td></tr>
<tr><td>7(b)(iii)</td><td>One mark per correct marking point (Max 2)

• A B // A D
• + A D // + A B

(Z =) A B + A D // A D + A B</td><td>2</td></tr>
<tr><td>7(b)(iv)</td><td>(Z =) A (B + D) // A (D + B)</td><td>1</td></tr>
</table>

Question	Answer	Marks
8(a)	**One** mark for each correct marking point **(Max 2)** • The SSL and TLS protocols provide communications security over the internet / network • … they provide **encryption** • They enable two parties to identify and authenticate each other • … and communicate with confidentiality and integrity.	2
8(b)	**One** mark for each correct marking point **(Max 4)** • An SSL/TLS connection is initiated by an application • … which becomes the client • The application which receives the connection becomes the server • Every new session begins with a handshake (as defined by the (SSL/TLS) protocols) • The client requests the digital certificate from the server // the server sends the digital certificate to the client • The client verifies the server's digital certificate • …and obtains the server's public key • The encryption algorithms are agreed • The symmetric • … session keys are generated / defined	4

Question	Answer	Marks
9(a)(i)	**One** mark for correct statement **(Max 1)** • Enables deep learning to take place • Where the problem you are trying to solve has a higher level of complexity it requires more layers to solve • To enable the neural network to learn and make decisions on its own • To improve the accuracy of the result.	1
9(a)(ii)	**One** mark for each correct marking point **(Max 4)** • Artificial neural networks are intended to replicate the way human brains work • Weights / values are assigned for each connection between nodes • The data are input at the input layer and are passed into the system • They are analysed at each subsequent (hidden) layer where characteristics are extracted / outputs are calculated • … this process of training / learning is repeated many times to achieve optimum outputs // reinforcement learning takes place • Decisions can be made without being specifically programmed • The deep learning net will have created complex feature detectors • The output layer provides the results • Back propagation (of errors) will be used to correct any errors that have been made.	4

Question	Answer	Marks
9(b)	**One** mark for each correct calculation as follows **(Max 4)** • Node B (from Home) (Line 3 in table) • Node C (from Home) (Line 4 in table) • Node B and Node E (from A) (Lines 5 and 6 in table) • Node F and Node School (from E) (Lines 7 and 8 in table) • Node School (from F) (Line 9 in table) **One** mark for correct path **(Max 1)**: • Home ⇨ A ⇨ E ⇨ F ⇨ School	**5**

	Node	Cost from Home Node (g)	Heuristic (h)	Total (f = g + h)
1	*Home*	*0*	*14*	*14*
2	*A*	*1*	*10*	*11*
3	B	5	7	12
4	C	4	9	13
5	B	1 + 3 = 4	7	11
6	E	1 + 6 = 7	3	10
7	F	7 + 1 = 8	3	11
8	School	7 + 5 = 12	0	12
9	School	8 + 3 = 11	0	11

Final Path	Home ⇨ A ⇨ E ⇨ F ⇨ School

Question	Answer	Marks
10(a)	**One** mark for each correct marking point **(Max 3)** • Must have a base case/stopping condition • Must have a general case • … which calls itself (recursively) // Defined in terms of itself • … which changes its state and moves towards the base case Unwinding can occur once the base case is reached.	**3**
10(b)	**One** mark for each correct marking point **(Max 3)** • A stack is a LIFO data structure • Each recursive call is pushed onto the stack • …. and is then popped as the function ends • Enables backtracking/unwinding … to maintain the required order.	**3**
10(c)	**One** mark for each marking point **(Max 2)** • Linked List • Queue Binary Tree	**2**

10(d)	**One** mark for each marking point **(Max 5)**
	
	• Checking if stack is full / empty using `IF … THEN … (ELSE) … ENDIF`
	• … correctly using `StackFull()` function
	• **`RETURN`** suitable message if stack is full
	• **`RETURN`** message if space available on stack
	• Incrementing `TopOfStack` pointer if space available
	• Assigning new data using correct `NewInteger` variable
	• … to correct the array element in `ArrayStack[]` array.
	
	Example algorithm
	`FUNCTION AddInteger(NewInteger : INTEGER) RETURNS STRING`
	`   IF StackFull() THEN`
	`      RETURN "The stack is full"`
	`   ELSE`
	`      TopOfStack ← TopOfStack + 1`
	`      ArrayStack[TopOfStack] ← NewInteger`
	`      RETURN "Item added"`
	`   ENDIF`
	`ENDFUNCTION`
		5