Cambridge Assessment International Education

# Cambridge International AS & A Level

**COMPUTER SCIENCE** **9618/31**

Paper 3 Advanced Theory **October/November 2023**

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 2023 series for most Cambridge IGCSE, Cambridge International A and AS Level components, and some Cambridge O Level components.



© UCLES 2023

## 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.

© UCLES 2023

| Question | Answer | Marks |
|---|---|---|
| 1(a) | **One** mark for working (**Max 1**)<br>• conversion of 65.25 to binary seen e.g. 1000001.01 = 65.25 // 64 + 1 + 0.25 / ¼<br><br>**One** mark per mark point (**Max 2**)<br>• correct mantissa<br>• correct exponent<br><br>**Mantissa**: 0 1 0 0 0 0 0 1 0 1 0 0<br>**Exponent**: 0 1 1 1 | **3** |
| 1(b) | **One** mark per mark point (**Max 2**)<br>MP1 the decimal fraction 0.20 cannot be represented **exactly** (the closest is 0.25 / 0.1875)<br><br>MP2 therefore, there will be a loss of precision due to a rounding error/truncation | **2** |

| Question | Answer | Marks |
|---|---|---|
| 2(a) | **One** mark for each correct line connecting a protocol to its most appropriate description (**Max 4**).<br><br>**Protocol**: HTTP, BitTorrent, SMTP, IMAP<br>**Use**: to provide peer-to-peer file sharing; when retrieving email messages from a mail server over a TCP/IP connection; when transmitting hypertext documents; to map MAC addresses onto IP addresses; when sending email messages towards the intended destination | **4** |
| 2(b) | **One** mark per mark point (**Max 2**)<br>MP1 To ensure correct network protocols are followed<br>MP2 To enable the upper layers to access the **physical medium** // enables connection/ communication with the internet / network layer<br>MP3 To be responsible for transporting data within the network/local segments<br>MP4 To format the data into frames for transmission<br>MP5 Maps IP addresses to MAC/Physical addresses. | **2** |

| Question | Answer | Marks |
|---|---|---|
| 3 | **One** mark per mark point – enumerated type (**Max 2**)<br>MP1 A user-defined non-composite (data type) (only award once)<br>MP2 …with a list of all possible values<br>MP3 …that is ordered.<br><br>**One** mark per mark point – pointer type (**Max 2**)<br>MP4 A user-defined non-composite (data type) (only award once)<br>MP5 …that stores addresses/memory locations only<br>MP6 …and indicates the type of data stored in the memory location. | **4** |

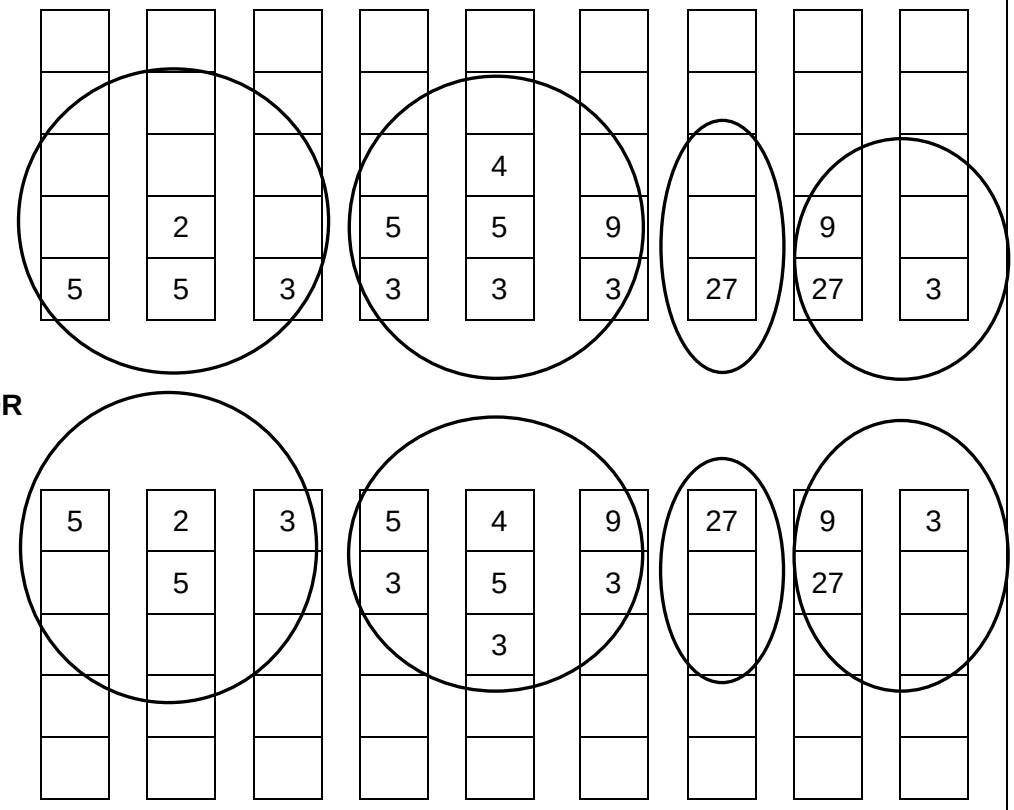
| Question | Answer | Marks |
|---|---|---|
| 4(a) | **One** mark per mark point – sequential (**Max 2**)<br>MP1 **Records** (in the file) are ordered<br>MP2 **…**based on the **key field**<br>MP3 A new version (of the file) has to be created to update the file<br><br>**One** mark per mark point – random (**Max 2**)<br>MP4 **Records** are stored in no particular order within the file // There is no sequencing in the placement of the **records**<br>MP5 There is a relationship between the key of the **record** and its location within the file // a hashing algorithm is used to find the location of the **record**<br>MP6 Updates to the file can be carried out directly. | **4** |
| 4(b) | **One** mark per mark point (**Max 2**)<br>MP1 Start at the beginning of the file<br>MP2 …check records linearly<br>MP3 …until the desired record is found // … processing / updating records as required //… EOF found. | **2** |

| Question | Answer | Marks |
|---|---|---|
| 5 | **One** mark per mark point – SISD (**Max 2**)<br>MP1 Single Instruction, Single Data (architecture). // Data is taken from a single source and a single instruction is performed on the data.<br>MP2 Contains **one** processor, a control unit and a memory unit.<br>MP3 …that executes instructions sequentially.<br><br>**One** mark per mark point – MIMD (**Max 2**)<br>MP4 Multiple Instruction, Multiple Data (architecture). // At any time, any processor can execute different instructions on different sets of data.<br>MP5 Contains **many** processors<br>MP6 …that operate **asynchronously / independently**. | **4** |

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>6(a)</td><td>One mark for every shaded block of rows for column Z correct (Max 3)<br><br>
<table>
<tr><th>A</th><th>B</th><th>C</th><th>D</th><th>Z</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td><td>3</td></tr>
<tr><td>6(b)</td><td>One mark for correct working from points (Max 2), for example:<br>
(Y =) $\overline{A}.\overline{B}.C.\overline{D} + \overline{A}.\overline{B}.C.D + A.\overline{B}.C.\overline{D} + A.B.C.\overline{D}$<br>
(Y =) $\overline{A}.\overline{D}.(\overline{B}.\overline{C} + \overline{B}.C + B.\overline{C} + B.C)$<br>
(Y =) $\overline{A}.\overline{D}.(\overline{B}.(\overline{C} + C) + B.(\overline{C} + C))$<br>
(Y =) $\overline{A}.\overline{D}.(\overline{B}.(1) + B.(1))$<br>
(Y =) $\overline{A}.\overline{D}.(\overline{B} + B)$<br>
(Y =) $\overline{A}.\overline{D}.(1)$<br><br>
One mark for correct answer<br>
(Y =) $\overline{A}.\overline{D}$</td><td>3</td></tr>
</table>

| Question | Answer | Marks |
|---|---|---|
| 7(a) | **One** mark for a benefit (**Max 1**) e.g.<br>MP1 The user interface hides the complexities of the computer hardware/operating system from the user<br>MP2 It provides appropriate access systems for users with differing needs<br>MP3 Complex commands involving memory locations/buses/computer hardware/ are avoided<br><br>**One** mark for a valid example (**Max 1**) e.g.<br>Clicking on icon rather than writing code<br>Using a graphical user interface / icons for navigation | **2** |
| 7(b) | Blocked (state) | **1** |
| 7(c) | **One** mark per mark point (**Max 2**)<br>MP1 When the time slice of the running process expires (round robin).<br>MP2 …and there is a process with a higher priority in the ready queue, the running process is pre-empted<br>MP3 When an interrupt arrives at the CPU, (the process running on the CPU gets pre-empted). | **2** |

| Question | Answer | Marks |
|---|---|---|
| 8(a) | **One** mark for each correctly completed line (**Max 5**)<br><br>`DECLARE Customer : TAccount`<br>`DECLARE Location : INTEGER`<br>`DECLARE AccountFile : STRING`<br>`AccountFile ← "AccountRecords.dat"`<br>`OPENFILE AccountFile FOR RANDOM`<br>`OUTPUT "Please enter an account number"`<br>`INPUT Customer.AccountNumber`<br>`Location ← Hash(Customer.AccountNumber)`<br>`SEEK AccountFile, Location`<br>`GETRECORD AccountFile, Customer`<br>`OUTPUT Customer`<br>`CLOSEFILE AccountFile` | **5** |
| 8(b) | **One** mark for correct definition<br>(Exception handling is the process of) responding to an unexpected **event** when the **program is running** so it does **not halt unexpectedly** | **1** |
| 8(c) | **One** mark per mark point (**Max 2**), for example:<br>• Programming errors<br>• User errors<br>• Hardware failure<br>• Runtime errors | **2** |

| Question | Answer | Marks |
|---|---|---|
| 9(a)(i) | **One** mark per mark point (**Max 2**)<br>• `(5 – 2)`<br>• `* (5 + 4) / 9`<br><br>Final correct expression<br>`(5 – 2) * (5 + 4) / 9` | **2** |
| 9(a)(ii) | **One** mark per ring (**Max 4**)<br><br><br>**OR**<br><br> | **4** |
| 9(b) | **One** mark per mark point (**Max 3**)<br>MP1 Evaluate the RPN expression from left to right<br>MP2 Push each element of the RPN expression onto the stack in order until an operator is reached<br>MP3 Pop the last two elements from the stack and apply the operator<br>MP4 Push the result of the operation onto the stack<br>MP5 Repeat the process until the whole expression is evaluated. | **3** |

| Question | Answer | Marks |
|---|---|---|
| 10(a) | **One** mark per mark point (**Max 3**)<br>MP1 Correct constant declaration<br>MP2 Two correct variable declarations<br>MP3 Correct array declaration<br><br>Example answer:<br>`CONSTANT Capacity = 25`<br>`DECLARE BasePointer : INTEGER`<br>`DECLARE TopPointer : INTEGER`<br>`DECLARE Stack : ARRAY[1:25] OF REAL` | **3** |
| 10(b) | **One** mark for each correctly completed line (**Max 5**)<br><br>`// popping an item from the stack`<br>`FUNCTION Pop() RETURNS REAL`<br>`  DECLARE Item : REAL`<br>`  Item ← 0`<br>`  IF TopPointer >= BasePointer THEN`<br>`  Item ← Stack[TopPointer]`<br>`  TopPointer ← TopPointer – 1`<br>`  ELSE`<br>`  OUTPUT "The stack is empty – error"`<br>`  ENDIF`<br>`  RETURN Item`<br>`ENDFUNCTION` | **5** |
| 10(c) | **One** mark per mark point (**Max 2**)<br>MP1 A queue is a first in first out / FIFO data structure and a stack is a first in last out / FILO / LIFO data structure // Data is removed from a queue in the order it is received and removed from a stack in the reverse order to which it is received<br>MP2 Both ADTs can vary in size / are of indeterminate length<br>MP3 Data is popped and pushed (onto/from a stack) at the same end but it is enqueued and dequeued (to/from a queue) at different/opposite ends // a queue has two accessible ends and a stack has only one<br>MP4 A stack has only one moveable pointer whereas a queue has two. | **2** |

| Question | Answer | Marks |
|---|---|---|
| 11(a) | **One** mark for each correctly completed clause (**Max 3**)<br><br>`(22) student(anthony).`<br>`(23) choice1(anthony, history).`<br>`(24) choice2(anthony, geography).` | **3** |
| 11(b) | `X = tomaz, pietre, nico` | **1** |

| Question | Answer | Marks |
|---|---|---|
| 11(c) | **One** mark per mark point (**Max 4**)<br>• `student(N)`<br>• `subject(S)`<br>• `choice1(N, S)`<br>• **all** logical operators correct with no additional code (see example answers)<br><br>Example answers:<br>`may_choose_subject(N, S)`<br>`IF` **`student(N) AND subject(S) AND NOT choice1(N, S)`**<br><br>`may_choose_subject(N, S)`<br>`IF` **`NOT choice1(N, S), student(N), subject(S)`** | **4** |

| Question | Answer | Marks |
|---|---|---|
| 12 | **One** mark per mark point (**Max 4**)<br>MP1 An artificial neural network is the component of artificial intelligence that is meant to simulate the functioning of a biological brain.<br>MP2 Artificial neural networks are a key component of machine learning.<br>MP3 They can solve problems that would prove impossible or difficult for humans // Artificial neural networks have self-learning capabilities that enable them to produce better results as more data becomes available<br>MP4 Artificial neural networks can be layered (input, hidden and output layers) // Artificial neural networks have many interconnected layers, some / many of which are hidden<br>MP5 Weights are assigned between nodes<br>MP6 Weights are adjusted through training to give a more accurate result<br>MP7 More complex learning capabilities / more accurate results are available with larger numbers of hidden layers | **4** |

© UCLES 2023