Cambridge Assessment International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/11**

Paper 11 Theory Fundamentals **May/June 2022**

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 May/June 2022 series for most Cambridge IGCSE, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

This document consists of **8** printed pages.

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)	**1 mark** for: one tebibyte is 1024 gibibytes **and** one terabyte is 1000 gigabytes	**1**
1(b)	1001 1100	**1**
1(c)	**1 mark** for working e.g. • Dividing by 16 // converting to binary (11111011) **1 mark** for answer FB	**2**
1(d)	1000 1110	**1**

Question	Answer	Marks
2(a)(i)	**1 mark** for each correct term. Random Access Memory (RAM) and Read Only Memory (ROM) are both examples of **primary** memory. One item that is stored in RAM is **currently running software/data/part of OS**. One item that is stored in ROM is **the start-up/boot-up instructions/BIOS**. RAM can be either Static RAM (SRAM) or Dynamic RAM (DRAM). SRAM uses transistors arranged as **flip-flops/latches.** DRAM uses transistors and **capacitors**.	**5**
2(a)(ii)	**1 mark** per bullet point to max 3 • PROM can be set once, EPROM and EEPROM can be overwritten multiple times. • EPROM needs to be removed from device EEPROM can be erased in situ. • EPROM and can be erased using UV light, EEPROM can be erased using voltage // is flash storage . • EPROM must be entirely erased before rewriting, EEPROM does not have to be entirely erased before rewriting.	**3**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>2(b)</td><td><b>1 mark</b> per bullet point to <b>max 5</b>
<ul>
<li>The hard disk has (one or more) platter/plate/disk</li>
<li>Each surface of the platter/disk is (ferrous oxide which is) capable of being magnetised</li>
<li>The platters/disks are mounted on a (central) spindle</li>
<li>The entire mechanism is contained inside a sealed (aluminium) box.</li>
<li>The disks are rotated (at high-speed)</li>
<li>(Each surface of the disk) has a read/write head mounted on an arm (positioned just above the surface)</li>
<li>Electronic circuits control the movement of the arm (and hence the heads)</li>
<li>The surface of the platter/disk is divided into <u>concentric</u> tracks / circles</li>
<li>The surface of the platter/disk is divided into sectors</li>
<li>One track in one sector is the basic unit of storage called a block</li>
<li>The data is encoded as a magnetic pattern for each block</li>
<li>When writing to disk, a variation in the current in the head produces a variation in magnetic field on the disk</li>
<li>When reading from disk, a variation in magnetic field produces a variation in current through the head</li>
</ul></td><td>5</td></tr>
<tr><td>2(c)(i)</td><td><b>1 mark</b> for each correct row.
<table>
<tr><th>Gate</th><th>Input 1</th><th>Input 2</th></tr>
<tr><td><b>AND</b></td><td>1</td><td>1</td></tr>
<tr><td><b>NAND</b></td><td>0
0
1</td><td>0
1
0</td></tr>
<tr><td><b>XOR</b></td><td>0
1</td><td>1
0</td></tr>
<tr><td><b>NOR</b></td><td>0</td><td>0</td></tr>
</table></td><td>4</td></tr>
<tr><td>2(c)(ii)</td><td><b>1 mark</b> per bullet point
<ul>
<li><b>A</b> NAND <b>B</b></li>
<li><b>B</b> XOR <b>C</b></li>
<li>OR</li>
</ul>
(<b>A</b> NAND <b>B</b>) OR (<b>B</b> XOR <b>C</b>)</td><td>3</td></tr>
</table>

Question	Answer	Marks
3(a)	Security prevents against **loss** while privacy prevents **unauthorised access**	**1**
3(b)	**1 mark** for identifying threat, **1 mark** for description, **1 mark** for security measure (**times 2**) e.g. • Malware • Malicious software that replicates and can delete/damage the examination papers • Install and run anti-malware • Hacker/unauthorised access • Illegal access in order to delete/damage the examination papers • Use a firewall // strong passwords • Spyware • Software installed on the computer without the teacher's knowledge which records keystrokes and sends the data gathered about the examination papers to a third party • Use a firewall / install and run anti-spyware / use a virtual (onscreen) keyboard	**6**

Question	Answer	Marks
4(a)	**1 mark** per bullet point to **max 4** e.g. • Reduced data redundancy // less repeated data • …because each item of data is only stored once • Maintains data consistency // improves data integrity • …changes in one table will automatically update in another • ... linked data cannot be entered differently in two tables • Program-data independence • …changes to the data do not require programs to be re-written • Complex queries are easier to run • Can provide different views • ….so users can only see specific aspects of the database	**4**
4(b)	**1 mark** for 3NF **1 mark** per bullet for justification to **max 2** • There are no repeated attributes // it is already in 2NF • Each field is fully dependent on the corresponding primary key // no partial dependencies • No transitive dependencies	**3**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(c)(i)</td><td>

1 mark per bullet point

- Create table, table name, opening and closing brackets
- `StudentID` and `Mark` as integer
- `TestID` as Varchar
- Primary key correctly set up
- Foreign keys correctly set up

e.g.

```
CREATE TABLE STUDENT_TEST (
   StudentId INTEGER,
   TestID VARCHAR,
   Mark INTEGER,
   PRIMARY KEY(StudentID,TestID),
   FOREIGN KEY(TestID) REFERENCES TEST(TestID),
   FOREIGN KEY(StudentID) REFERENCES STUDENT(StudentID)
);
```

</td><td>5</td></tr>
<tr><td>4(c)(ii)</td><td>

1 mark for each point

- AVG(Mark)
- SELECT and FROM STUDENT_TEST
- WHERE clause

e.g.

```
SELECT AVG(Mark)
FROM STUDENT_TEST
WHERE TestID = "A7";
```

</td><td>3</td></tr>
<tr><td>4(d)</td><td>

1 mark per bullet point to **max 3** for validation

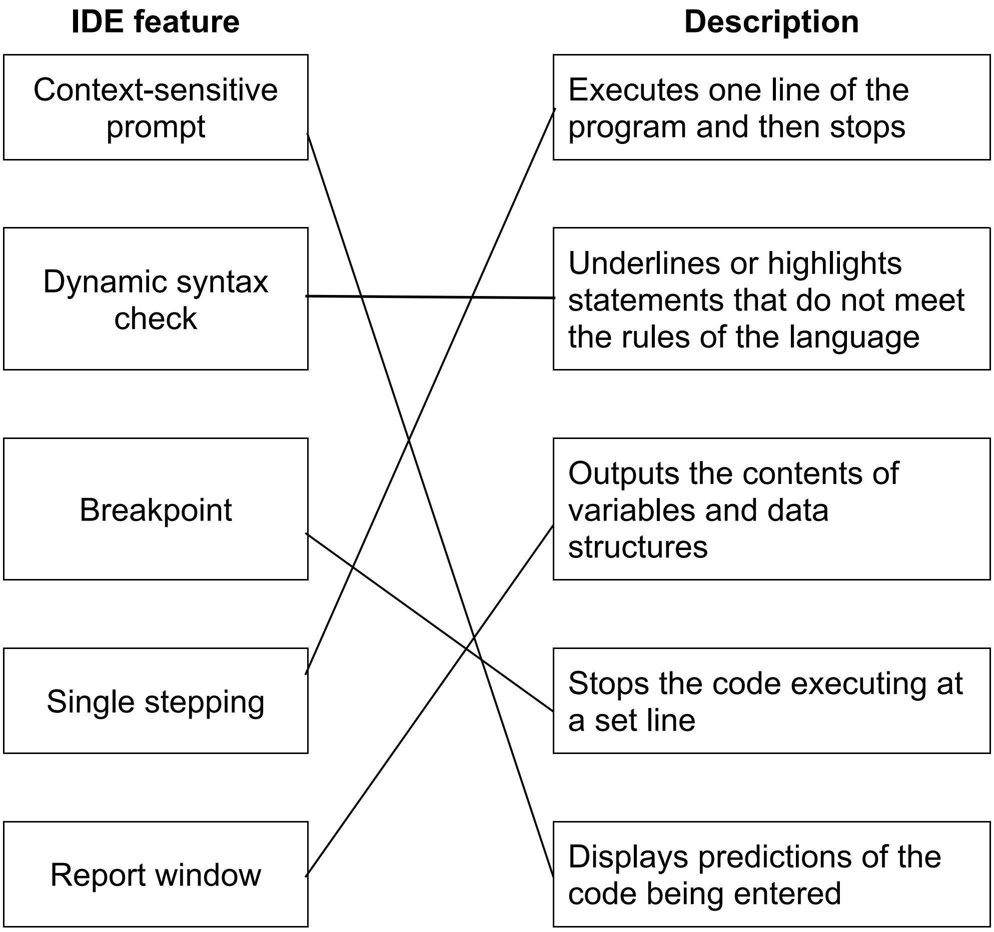
e.g.

- **range check** to make sure it is between 0 and max marks
- **presence check** to make sure a mark is entered
- **type check** to make sure an integer value is entered

1 mark per bullet point to **max 2** for verification
e.g.

- double entry - enter the mark twice and **the computer** compares them
- visual check – **manually** compare the mark entered with the mark on the input document

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

Question	Answer	Marks
5(a)	**1 mark** for 1 correct line, **2 marks** for 2 correct lines, **3 marks** for 3 or 4 correct lines, **4 marks** for all 5 correct lines  **IDE feature** Context-sensitive prompt Dynamic syntax check Breakpoint Single stepping Report window **Description** Executes one line of the program and then stops Underlines or highlights statements that do not meet the rules of the language Outputs the contents of variables and data structures Stops the code executing at a set line Displays predictions of the code being entered	**4**
5(b)	**1 mark** each: • Open Source Initiative • Free Software Foundation	**2**
5(c)	**1 mark** per bullet point to **max 3** • Saves (programming/testing) time as code does not have to be written/re-written from scratch // code does not have to be tested • Code is already tested so it is more robust/likely to work • If there is an improvement in the library routine the program updates automatically • can perform complex calculations that the programmer may be unable to do	**3**

Question	Answer	Marks
6(a)(i)	**1 mark** per bullet point to **max 5** • The Program Counter (PC) holds the **address** of the next instruction … • …and the **contents** are incremented / changed to the next address each cycle • The Memory Address Register (MAR) holds the address to fetch the data (from the PC) • The Memory Data Register (MDR) holds the data at the address in MAR • The instruction is transferred to Current Instruction Register (CIR) for decoding and execution	**5**
6(a)(ii)	**1 mark** for detection • At the start/end of a FE cycle **1 mark** for handling to **max 4** • Priority is checked • If lower priority than current process continue with F-E cycle • If higher priority than current process … • … state of current process is / registers are stored on stack • Location / type of interrupt identified... • …appropriate ISR is called to handle the interrupt • When ISR finished, check for further interrupts (of high priority) / return to step 1 • Otherwise load data from stack and continue with process	**5**
6(b)	**1 mark** for factor **1 mark** for impact e.g. • Clock speed… • …higher clock speed means more FE cycles per second • Number of cores… • …means more instructions can be carried out simultaneously • Bus width … • …allows the transfer of more data each time // allows more memory locations to be directly accessed • Cache … • … the higher capacity the more frequently used instructions it can store for fast access	**2**
6(c)(i)	0000 0000	**1**
6(c)(ii)	0110 1101	**1**
6(c)(iii)	1101 0000	**1**