Cambridge Assessment International Education

# Cambridge International AS & A Level

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CANDIDATE NAME | | | | | | | | | |
| CENTRE NUMBER | | | | | | CANDIDATE NUMBER | | | | |

**COMPUTER SCIENCE** **9618/12**

Paper 1 Theory Fundamentals **October/November 2022**

**1 hour 30 minutes**

You must answer on the question paper.

No additional materials are needed.

**INSTRUCTIONS**

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

**INFORMATION**

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **20** pages. Any blank pages are indicated.

**1** **(a)** Draw **one** line from each utility software to its most appropriate purpose.

| Utility software | Purpose |
|---|---|
| virus checker | to reorganise files so they are contiguous |
| disk formatter | to scan for malicious program code |
| backup | to decrease the file size |
| disk repair | to initialise a disk |
| defragmentation | to create copies of files in case the original is lost |
| | to check for and fix inconsistencies on a disk |

[5]

**(b)** Compilers and interpreters translate programs written in a high-level language into a low-level language.

**(i)** State **two** drawbacks of using a compiler compared to an interpreter during program development.

1 .....

.....

.....

2 .....

.....

.....

[2]

**(ii)** Explain why high-level language programs might be partially compiled and partially interpreted.

.....

.....

.....

..... [2]

**2** **(a)** **(i)** Convert the two's complement binary integer into denary.

10010110

Answer ..... [1]

**(ii)** Convert the unsigned binary integer into hexadecimal.

10010110

Answer ..... [1]

**(iii)** Convert the unsigned binary integer into Binary Coded Decimal (BCD). Show your working.

10010101

Working .....

.....

.....

Answer .....

[2]

**(b)** Perform the following binary addition.

$$\begin{array}{rcccccccc} & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ + & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ \hline \end{array}$$

[1]

**3** **(a)** A greenhouse has an automatic window.

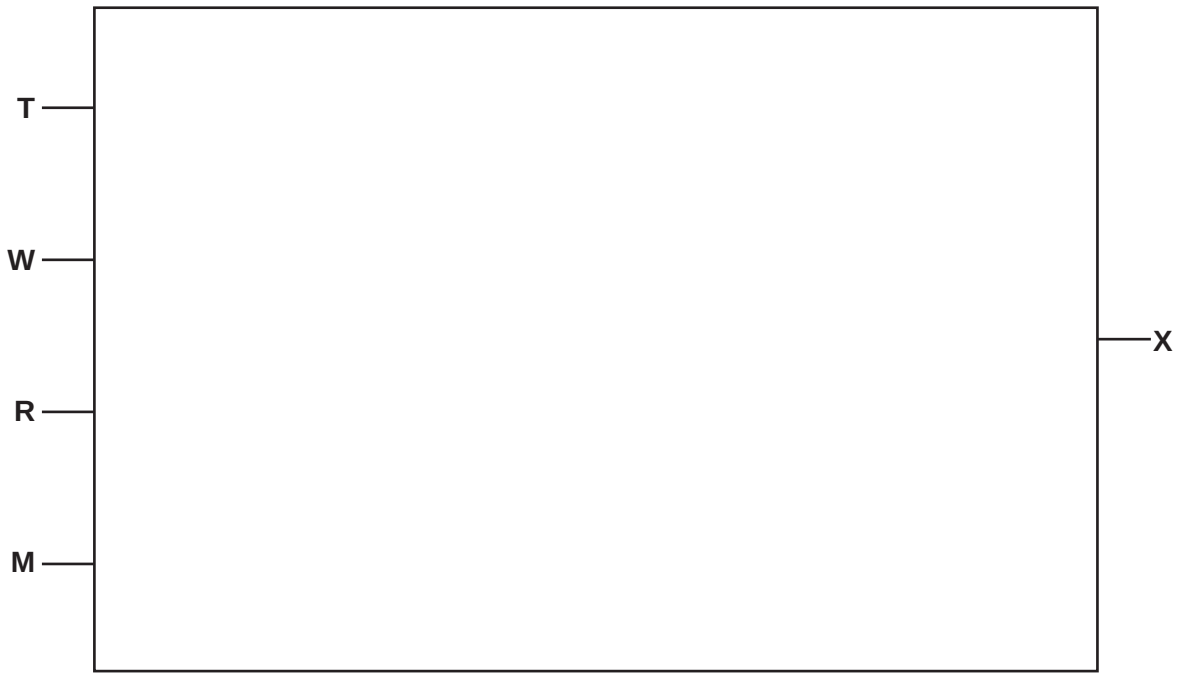
The window (**X**) operates according to the following criteria:

| Parameter | Description of parameter | Binary value | Condition |
|---|---|---|---|
| **T** | Temperature | 1 | Too high |
| | | 0 | Acceptable |
| **W** | Wind speed | 1 | Too high |
| | | 0 | Acceptable |
| **R** | Rain | 1 | Detected |
| | | 0 | Not detected |
| **M** | Manual override | 1 | On |
| | | 0 | Off |

The window opens (**X** = 1) if:

- the temperature is too high **and** the wind speed is acceptable
- **and**
- rain is not detected, **or** the manual override is off.

Draw a logic circuit to represent the operation of the window.




[3]

**(b)** Complete the truth table for the logic expression:

**X** = NOT (**A** OR **B** OR **C**) AND (**B** NOR **C**)

| A | B | C | Working space | X |
|---|---|---|---|---|
| 0 | 0 | 0 | | |
| 0 | 0 | 1 | | |
| 0 | 1 | 0 | | |
| 0 | 1 | 1 | | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

[2]

**(c)** Embedded systems contain Read Only Memory (ROM) and Random Access Memory (RAM).

Explain the reasons why ROM is used in an embedded system.

.....

.....

.....

.....

.....

..... [2]

4 **(a)** State the difference between **data verification** and **data validation**.

.....

.....

..... [1]

**(b)** A checksum can be used to detect errors during data transmission.

Describe how a checksum is used.

.....

.....

.....

.....

.....

..... [3]

**(c)** One validation method is a presence check.

Describe **two other** validation methods that can be used to validate non-numeric data.

1 .....

.....

2 .....

.....

[2]

**5** A relational database, GARDEN, has the following tables:

OWNER(OwnerID, FirstName, TelephoneNo, TreeID, TreePosition)

TREE(TreeID, ScientificName, MaxHeight, FastGrowing)

**(a)** The database is **not** in Third Normal Form (3NF).

Explain how the database can be normalised to 3NF.

.....

.....

.....

.....

.....

..... [3]

**(b)** Write the Structured Query Language (SQL) script to add a new record in the table TREE to store the following data:

| Attribute | Value |
|---|---|
| TreeID | LOW_1276 |
| ScientificName | Salix_Alba |
| MaxHeight | 30.00 |
| FastGrowing | TRUE |

.....

.....

.....

..... [3]

**(c)** State what is meant by a **candidate key** in a relational database.

.....

..... [1]

**(d) (i)** Describe, using an example, what is meant by a **data dictionary**.

.....

.....

.....

..... [2]

**(ii)** Describe what is meant by a **logical schema**.

.....

.....

.....

..... [2]

**6** **(a)** A student uses a networked laptop computer to send an email to a colleague.

**(i)** Explain how a digital signature ensures the email is authentic.

.....

.....

.....

..... [2]

**(ii)** Describe how a firewall protects the data on the computer.

.....

.....

.....

.....

.....

..... [3]

**(b)** The student records a sound file.

**(i)** Explain the effect of increasing the sampling rate on the accuracy of the sound recording.

.....

.....

.....

..... [2]

**(ii)** Explain the effect of decreasing the sampling resolution on the file size of the sound recording.

.....

.....

.....

..... [2]

**7** The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

| Instruction | | Explanation |
|---|---|---|
| **Opcode** | **Operand** | |
| LDM | #n | Immediate addressing. Load the number n to ACC |
| LDD | <address> | Direct addressing. Load the contents of the location at the given address to ACC |
| LDX | <address> | Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC |
| LDR | #n | Immediate addressing. Load the number n to IX |
| STO | <address> | Store the contents of ACC at the given address |
| ADD | <address> | Add the contents of the given address to the ACC |
| ADD | #n | Add the denary number n to the ACC |
| INC | <register> | Add 1 to the contents of the register (ACC or IX) |
| JMP | <address> | Jump to the given address |
| CMP | <address> | Compare the contents of ACC with the contents of <address> |
| CMI | <address> | Indirect addressing. The address to be used is at the given address. Compare the contents of ACC with the contents of this second address |
| JPE | <address> | Following a compare instruction, jump to <address> if the compare was True |
| JPN | <address> | Following a compare instruction, jump to <address> if the compare was False |
| END | | Return control to the operating system |
| <address> can be an absolute or symbolic address<br># denotes a denary number, e.g. #123<br>B denotes a binary number, e.g. B01001101 | | |

**(a)** Trace the program currently in memory using the trace table, stopping when line `90` is executed for a second time.

| Address | Instruction |
|---|---|
| 75 | `LDR #0` |
| 76 | `LDX 110` |
| 77 | `CMI 102` |
| 78 | `JPE 91` |
| 79 | `CMP 103` |
| 80 | `JPN 84` |
| 81 | `ADD 101` |
| 82 | `STO 101` |
| 83 | `JMP 86` |
| 84 | `INC ACC` |
| 85 | `STO 101` |
| 86 | `LDD 100` |
| 87 | `INC ACC` |
| 88 | `STO 100` |
| 89 | `INC IX` |
| 90 | `JMP 76` |
| 91 | `END` |
| … | |
| 100 | `0` |
| 101 | `0` |
| 102 | `112` |
| 103 | `4` |
| … | |
| 110 | `1` |
| 111 | `4` |
| 112 | `0` |

| Instruction address | ACC | IX | Memory address | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | 100 | 101 | 102 | 103 | 110 | 111 | 112 |
| | | | 0 | 0 | 112 | 4 | 1 | 4 | 0 |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

[5]

**(b)** The following table shows another part of the instruction set for the processor.

| Instruction | | Explanation |
|---|---|---|
| **Opcode** | **Operand** | |
| AND | #n | Bitwise AND operation of the contents of ACC with the operand |
| AND | <address> | Bitwise AND operation of the contents of ACC with the contents of <address> |
| XOR | #n | Bitwise XOR operation of the contents of ACC with the operand |
| XOR | Bn | Bitwise XOR operation of the contents of ACC with the binary number n |
| XOR | <address> | Bitwise XOR operation of the contents of ACC with the contents of <address> |
| OR | #n | Bitwise OR operation of the contents of ACC with the operand |
| OR | <address> | Bitwise OR operation of the contents of ACC with the contents of <address> |
| LSL | #n | Bits in ACC are shifted logically n places to the left. Zeros are introduced on the right-hand end |
| LSR | #n | Bits in ACC are shifted logically n places to the right. Zeros are introduced on the left-hand end |
| <address> can be an absolute or symbolic address<br># denotes a denary number, e.g. #123<br>B denotes a binary number, e.g. B01001101 | | |

The contents of memory addresses 50 and 51 are shown:

| Memory address | Data value |
|---|---|
| 50 | 01001101 |
| 51 | 10001111 |

**(i)** The current contents of the ACC are:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

Show the contents of the ACC after the execution of the following instruction.

```
XOR B00011111
```

.....

.....

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

[1]

**(ii)** The current contents of the ACC are:

| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Show the contents of the ACC after the execution of the following instruction.

```
AND 50
```

.....

.....

| | | | | | | | |
|---|---|---|---|---|---|---|---|

[1]

**(iii)** The current contents of the ACC are:

| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Show the contents of the ACC after the execution of the following instruction.

```
LSL #3
```

.....

.....

| | | | | | | | |
|---|---|---|---|---|---|---|---|

[1]

**(iv)** The current contents of the ACC are:

| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

Show the contents of the ACC after the execution of the following instruction.

```
OR 51
```

.....

.....

| | | | | | | | |
|---|---|---|---|---|---|---|---|

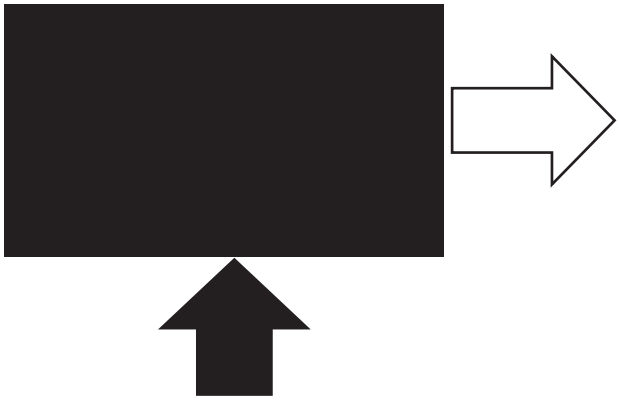
[1]

**(c)** Write the register transfer notation for each of the stages in the fetch-execute cycle described in the table.

| Description | Register transfer notation |
|---|---|
| Copy the address of the next instruction into the Memory Address Register. | |
| Increment the Program Counter. | |
| Copy the contents of the Memory Data Register into the Current Instruction Register. | |

[3]

**8** The following bitmap image has a resolution of 4096 × 4096 pixels and a colour depth of 24 bits per pixel.



The image is displayed on a monitor that has a screen resolution of 1920 × 1080 pixels.

**(a)** Tick (✓) **one** box in each row to identify the effect of each action on the image file size.

| Action | Increases the file size | Decreases the file size | No change to the file size |
|---|---|---|---|
| Change the colour depth of the image file to 16 bits per pixel. | | | |
| Change the screen resolution to 1366 × 768 pixels. | | | |
| Change the colour of the rectangle from black to red. | | | |

[2]

**(b)** State **two** benefits of creating a vector graphic instead of a bitmap image.

1 .....

.....

2 .....

.....

[2]

**(c)** A second bitmap image is stored using a colour depth of 8 bits per pixel.

The file is compressed using run-length encoding (RLE).

**(i)** The table shows the compressed and uncompressed values for parts of the image file.

Each colour of the pixel in the image is represented by a hexadecimal value.

Complete the table. The first row has been completed for you.

| Uncompressed image | RLE compressed image |
|---|---|
| EA F1 F1 F2 F2 F2 EA | 1EA 2F1 3F2 1EA |
| | 2AB 2FF 11D 167 |
| 32 32 80 81 81 | |

[2]

**(ii)** RLE is an example of lossless compression.

Explain why lossless compression is more appropriate than lossy compression for a text file.

.....

.....

.....

.....

..... [2]

**9** One use of Artificial Intelligence (AI) is for facial recognition software.

Describe the social impact of using facial recognition software to identify individuals in an airport.

.....

.....

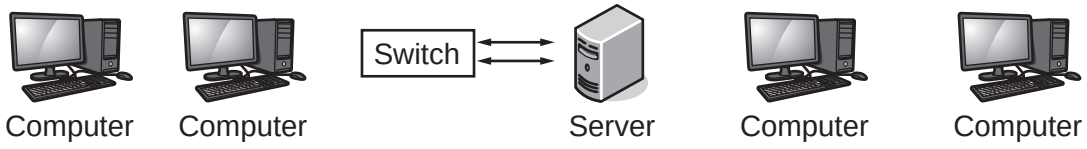
.....

..... [2]

**10** A Local Area Network (LAN) consists of four computers, one server and a switch.

The LAN uses a star topology.

**(a)** Complete the following diagram to show how the hardware is connected.




[1]

**(b)** A router is attached to one of the devices on the LAN shown in **part (a)** to connect the LAN to the internet.

**(i)** Identify the device. Give a reason for your choice.

Device .....

Reason .....

.....

.....

[2]

**(ii)** Describe the role **and** function of the router in the network.

.....

.....

.....

.....

.....

..... [3]

**BLANK PAGE**

**BLANK PAGE**

**BLANK PAGE**