Week 1

A-Level Computer Science

Homework bundle for this week

本周作业合集

exercises.lol

Contents 目录

1. Information representation

Starter Quiz · 课前小测

A-Level Computer Science Name: _____ Score: _____

1. How many bits does one hexadecimal digit represent?

   ☐ 4

   ☐ 1

   ☐ 8

   ☐ 16

2. Add the binary numbers ‘0101‘ + ‘0011‘. Give the 4-bit result.

   _____

3. How many different code points does 7-bit ASCII have?

   _____

4. The colour depth (bit depth) of a bitmap image is:

   ☐ the number of bits used to store each pixel

   ☐ the number of pixels across the image

   ☐ the physical size of the screen

   ☐ how much the file is compressed

5. The sampling rate of a digital sound is:

   ☐ the number of amplitude samples taken per second

   ☐ the number of bits per sample

   ☐ the loudness of the sound

   ☐ the length of the clip

6. Lossless compression means:

   ☐ the exact original data can be recovered

   ☐ some detail is permanently lost

   ☐ the file gets bigger

   ☐ the data is encrypted

7. How many bits are in one byte?

   _____ bits

8. Overflow in binary addition means:

☐ the result needs more bits than the register can hold

☐ the numbers were too small

☐ a bit was inverted by mistake

☐ the register was empty

9. How many bits of colour depth are needed to store 256 different colours?

   _____ bits

10. A 5-second mono clip is sampled at 8000 Hz with 8-bit resolution. What is its size in bits?

    _____ bits

1. Information representation

Answers · 答案

A-Level Computer Science

1. **4**

   One hex digit (0–F) covers 16 values = 2^4, so it maps to exactly 4 bits (a nibble).

2. **1000**

   $5 + 3 = 8$, which is '1000' in binary.

3. **128**

   7 bits give $2^7 = 128$ code points.

4. **the number of bits used to store each pixel**

   Colour depth is bits per pixel — more bits means more possible colours (8 bits → 256, 24 bits → 16.7 million).

5. **the number of amplitude samples taken per second**

   Sampling rate = samples per second (hertz). The bits per sample is the sample resolution.

6. **the exact original data can be recovered**

   Lossless compression lets you rebuild the original data exactly — essential for text, programs and ZIP/PNG.

7. **8 bits**

   A byte is 8 bits (and a nibble is 4 bits).

8. **the result needs more bits than the register can hold**

   Overflow occurs when the sum is too large to fit in the available bits; the carry out of the leftmost column is lost.

9. **8 bits**

   $2^8 = 256$, so 8 bits per pixel give 256 colours.

10. **320000 bits**

    size = $8000 \times 8 \times 5 \times 1 = 320\,000$ bits.

A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

1.1 Data Representation

Name: _____ Class: _____ Date: _____ **Total: 28 marks**

KEY WORDS binary 二进制 • denary 十进制 • hexadecimal 十六进制 • two's complement 补码 • overflow 溢出

Objective

Build the skills to answer the real exam questions on **data representation** 数据表示 — number bases, binary arithmetic, two's complement, BCD, and **character sets** 字符集.

You must be able to:

- convert between **binary** 二进制, **denary** 十进制 and **hexadecimal** 十六进制, and use **BCD** 二进码十进数
- state the **minimum number of bits** needed to store a value
- add 8-bit binary numbers and **name and describe overflow** 溢出
- use **two's complement** 补码: write a negative denary number in binary, read a two's-complement value, and subtract
- represent character data with **ASCII** and **Unicode**, and explain why Unicode is used

1 Worked examples

Study these first. Each one shows the method for an exam question type used later — follow the steps and you can do the Practice and Exam-style questions yourself.

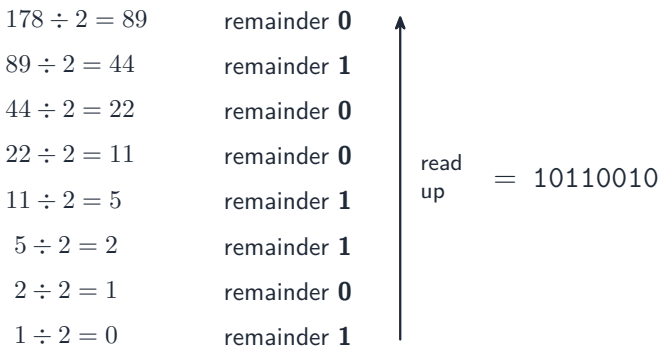
■ Binary ↔ denary

Multiply each bit by **2 raised to its position** (positions count from 0 at the right), then add. This same method works for *any* base — for hexadecimal you use powers of 16.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	1	0	0	1	0

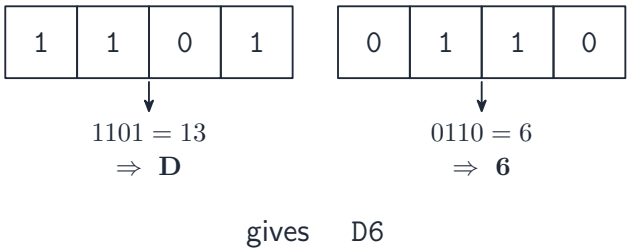
$$(1\times2^7)+(1\times2^5)+(1\times2^4)+(1\times2^1)$$
$$= 128+32+16+2 = \mathbf{178}$$

To go **denary → binary**, divide by 2 again and again; the **remainders** 余数, read from **bottom to top**, are the bits.



Hexadecimal

Each hex digit is exactly 4 bits (a **nibble** 半字节): 0–9, then A = 10 up to F = 15. **Hex → denary** uses the powers method with base 16: `2F` $= (2 \times 16^1) + (15 \times 16^0) = 32 + 15 =$ **47**. **Shortcut binary ↔ hex:** group the bits into nibbles of 4 from the right and convert each.



Number of bits to store a value

To store the values 0 to N, you need the smallest number of bits b with $2^b > N$. Example: to store 0–200, $2^7 = 128$ is too small but $2^8 = 256 > 200$, so **8 bits**. (An 8-bit byte holds 0–255.)

Binary addition and overflow

Add one column at a time from the right; when a column makes 2, write 0 and **carry** a 1.

```
  0101 1100   (92)
+ 0011 0110   (54)
------------
  1001 0010   (146)
```

Overflow is when the answer is too big for the register (over 255 for 8 bits): a 1 carries out of the leftmost (bit-7) column, so the stored 8-bit result is wrong.

```
  1100 1000   (200)
+ 0101 0000   (80)
------------
1 0001 1000   (carry out of bit 7 → overflow)
```

In words: **overflow** is when a calculation gives a result with more bits than the register can hold, so the carry out of the most-significant bit is lost and the stored answer is incorrect.

Two's complement: negative numbers

In two's complement the **most significant bit** is the **sign bit** (0 = positive, 1 = negative). For 8 bits the range is -128 to $+127$.

- **Write** -18**:** take $+18$ = `00010010`, **invert** every bit → `11101101`, **add 1** → `11101110`.
- **Read `11101110`:** the sign bit is 1, so it is negative. Invert → `00010001`, add 1 → `00010010` = 18, so the value is -18.
- **Subtract** by adding a negative. $45-18$: add `00101101` (45) and `11101110` (-18), then **discard** the carry out of bit 7:

```
  0010 1101   (45)
+ 1110 1110   (-18)
------------
1 0001 1011   (discard carry → 27)
```

Binary Coded Decimal (BCD)

BCD stores each denary digit on its own as a 4-bit code — *not* the plain binary of the whole number. For 59: 5 → `0101` and 9 → `1001`, so 59 is `0101 1001`. Each nibble uses only 0–9; patterns `1010`–`1111` are invalid.

Character sets: ASCII and Unicode

A computer stores text as numbers: each character has a **code point** 码点 fixed by a character set.

- **ASCII** uses **7 bits** → 128 characters (basic English letters, digits, punctuation, control codes). Example: `'A'` = 65 = `1000001`.

- **Unicode** is a universal set covering (almost) every script plus symbols and emoji, so it needs **more bits per character**.

A character's stored code is just a binary number, so you convert it like any other: the value `0010 0111` $= 32 + 4 + 2 + 1 = 39$.

Why Unicode? It represents **far more characters** than ASCII (every language, not just English), at the cost of **larger files** for plain English text.

2 Practice

Now apply the methods above.

2.1 Convert binary `01101010` to denary. [1]

2.2 Convert denary 200 to 8-bit binary. [2]

2.3 Convert binary `11110001` to (a) denary and (b) hexadecimal. [2]

2.4 State the **minimum number of bits** needed to store the denary value 500. Show how you decided. [2]

2.5 Represent the denary number 59 in **BCD**. [2]

2.6 Add the 8-bit numbers `01011100` and `00110110`. State whether overflow occurs. [2]

3 Exam-style questions

3.1 Add the 8-bit binary numbers `10110100` and `01101101`. Show your working, then **name and describe** the error that occurs. [4]

3.2 (a) Convert the hexadecimal number `3D` to denary. Show your working. [2]

(b) Convert the denary number 158 to hexadecimal. Show your working. [2]

3.3 (a) Write the denary number −45 as an 8-bit two's complement binary number. Show your working. [2]

(b) The 8-bit two's complement number `11101110` is stored. Calculate the denary value

it represents. [2]

3.4 A computer stores text using a character set.

(a) The text is stored using the **Unicode** character set instead of **ASCII**. Give **two** advantages of using Unicode. [2]

(b) A character has the binary code `0010 0111`. Convert this code to denary. [1]

3.5 A program stores currency amounts using **BCD** rather than ordinary binary.

(a) State what **BCD** stands for. [1]

(b) Give one **advantage** of using BCD for this purpose. [1]

Solutions

2.1 `01101010` $= 64 + 32 + 8 + 2 = 106$.

2.2 Repeated division by 2 gives remainders, read bottom-up $\rightarrow$ `11001000`.

2.3 (a) `11110001` $= 128 + 64 + 32 + 16 + 1 = 241$.
(b) split into `1111 | 0001` = `F1`.

2.4 $2^8 = 256$ is too small for 500, but $2^9 = 512 > 500$, so **9 bits**.

2.5 $5 \rightarrow$ `0101`, $9 \rightarrow$ `1001`, so `0101 1001`.

2.6 `01011100` (92) + `00110110` (54) = 146 = `10010010`; $146 \leq 255$, so **no overflow**.

3.1 Adding the columns: \

```
  1011 0100   (180)
+ 0110 1101   (109)
------------
  0010 0001   (carry out of bit 7)
```

$180 + 109 = 289 > 255$. The error is **overflow**: the result needs 9 bits but only 8 are stored, so the carry out of the most-significant bit is lost and the stored answer is wrong.

3.2 (a) `3D` $= (3 \times 16) + 13 = 48 + 13 = 61$.
(b) $158 = (9 \times 16) + 14$, and 14 = E, so `9E`.

3.3 (a) $+45$ = `00101101`; invert $\rightarrow$ `11010010`; add 1 $\rightarrow$ `11010011`.
(b) sign bit is 1 $\rightarrow$ negative; invert `11101110` $\rightarrow$ `00010001`; add 1 $\rightarrow$ `00010010` = 18; so the value is -18.

3.4 (a) Any **two**: Unicode can represent characters from (almost) every language / many more characters than ASCII; it includes symbols and emoji; it avoids the regional clashes of extended ASCII.
(b) `0010 0111` $= 32 + 4 + 2 + 1 = 39$.

3.5 (a) Binary Coded Decimal.
(b) Each denary digit is stored exactly, so values like money are not changed by rounding errors when converting fractions to binary (or: BCD digits map directly to a display).

1 (a) (i) Convert the binary number into hexadecimal.

101100111010

..... [1]

(ii) Convert the denary number into Binary Coded Decimal (BCD).

108

..... [1]

(iii) Convert the 12-bit two's complement binary integer into denary.

Show your working.

111110111100

Working .....

.....

.....

.....

Denary value .....

[2]

(b) (i) The following binary addition is performed using 8-bit registers.

Complete the calculation using binary addition.

```
  1 0 1 1 0 0 1 1
+ 0 1 1 1 1 0 0 0
```

[1]

(ii) Name and describe the error that can occur when binary addition is performed.

.....

.....

.....

..... [2]

2 **(a)** Data in a computer system is represented in binary.

Put **one** tick (✓) in each row to identify the minimum number of bits used to store each example of data.

Example of data	Number of bits						
	4	**8**	**16**	**24**	**32**	**64**	**128**
the hexadecimal value F139							
16 000 000 unique amplitude values							
an IPv4 address							
256 unique colours							
an IPv6 address							
the denary value 65 000							

[3]

(b) Convert the denary number –108 into a 12-bit two's complement binary number.

..... [1]

(c) A three-place arithmetic shift to the right is performed on the following two's complement negative integer.

Show the result of this arithmetic shift.

10010011

..... [1]

(d) Convert the following positive binary integer into hexadecimal.

1110001100111011

..... [1]

2 **(a)** Convert the denary integer 558 into 12-bit binary and hexadecimal.

Binary .....

Hexadecimal .....
[2]

(b) **(i)** Convert the two's complement binary integer 11100010 into denary.

.....

.....

..... [1]

(ii) Write the smallest and the largest two's complement binary integers that can be represented in 8 bits.

Smallest .....

Largest .....
[2]

(c) Give **one** application where Binary Coded Decimal (BCD) is used and justify its use.

Application .....

Justification .....

.....
[2]

6 A computer system stores text, images and sound.

(a) A character set is used to represent characters in a computer.

Identify **and** describe **one** character set.

Character set .....

Description .....

.....

[2]

(b) The colour of each pixel in a bitmapped image is represented by 8 bits.

(i) State the largest number of different colours that can be represented by 8 bits.

..... [1]

(ii) State **one** drawback of increasing the number of bits that represents each pixel in the bitmap image.

.....

..... [1]

(iii) A bitmap image can be compressed using lossy compression.

Explain the reasons why lossy compression is often suitable for a bitmap image.

.....

.....

.....

..... [2]

(c) (i) Explain how an analogue sound wave is converted into digital data.

.....

.....

.....

..... [2]

(ii) Describe **one** method of compressing a sound file using lossy compression.

.....

.....

.....

..... [2]

8 **(a)** Convert the hexadecimal number 1FAB into denary.

Working .....

.....

.....

Denary value ..... [1]

(b) Explain how to convert the two's complement integer 10011111 into denary. Give the denary value after conversion.

Explanation .....

.....

.....

.....

.....

Denary value .....
[3]

(c) Describe the difference between a right logical binary shift and a right arithmetic binary shift.

.....

.....

.....

..... [2]

7 Complete the binary addition. Show your working.

```
  1 0 0 1 1 1 1 0
  0 1 1 0 0 0 0 1
+ 0 0 0 1 1 0 0 1
```

[3]

7 A computer stores binary data.

(a) Tick (✓) **one** box only to identify the **largest** file size.

- ☐ 3300 kibibytes
- ☐ 0.3 megabytes
- ☐ 3 mebibytes
- ☐ 3300 kilobytes

[1]

(b) Subtract the denary number 10 from the denary number 100 using binary subtraction.

Show your working.

Working .....

.....

.....

.....

.....

.....

Answer .....

[3]

(c) Convert the hexadecimal number C0F into denary.

Show your working.

Working .....

.....

.....

Answer .....

[2]

A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

1.2 Multimedia

Name: _____ Class: _____ Date: _____ **Total: 41 marks**

KEY WORDS bitmap 位图 • pixel 像素 • resolution 分辨率 • file size 文件大小 • sampling 采样

Objective

Build the skills to answer exam questions on **multimedia** 多媒体 — how bitmap images and sound are stored, how to calculate their **file size** 文件大小, and when to use a bitmap or a vector graphic.

You must be able to:

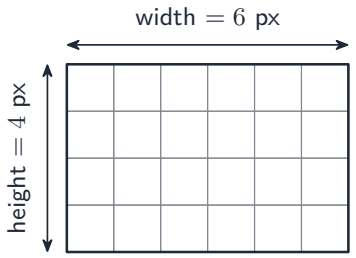
- describe how a **bitmap** 位图 image is encoded, and calculate its file size
- explain how **resolution** 分辨率 and **colour depth** 颜色深度 affect quality and size
- describe how a **vector graphic** 矢量图形 is encoded, and choose bitmap *or* vector for a task
- describe how sound is **sampled** 采样, and how the sampling rate and resolution affect quality and size

1 Worked examples

Study these first. Each one shows the method for a question type used later — follow the steps and you can do the Practice and Exam-style questions yourself.

■ Bitmap images and file size

A **bitmap** image stores the colour of every **pixel** 像素 in a grid. Two numbers describe it: the **resolution** (width × height in pixels) and the **colour depth** (bits per pixel). Multiply all three to get the size in bits.



pixels = $6 \times 4 = 24$

size = $24 \times \mathbf{8} = \mathbf{192}$ bits

(**8** = colour depth, bits per pixel)

$$\text{size (bits)} = \text{width} \times \text{height} \times \text{colour depth}$$

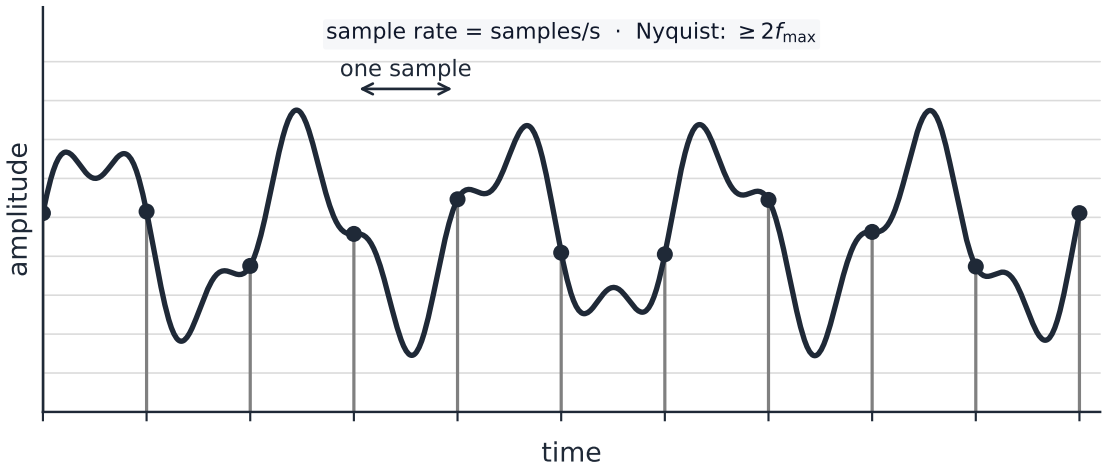
Then ÷ 8 for **bytes**, ÷ 1024 for **KiB**, ÷ 1024 again for **MiB**.

Example. A 200×150 image at 8 bits per pixel:

$$200 \times 150 \times 8 = 240\,000 \text{ bits} = 30\,000 \text{ bytes} = \frac{30\,000}{1024} \approx 29.3 \text{ KiB}.$$

Sound and file size

Sound is **sampled**: the wave's height (**amplitude** 振幅) is measured at regular time intervals. The **sampling rate** 采样率 is how many samples per second (Hz); the **resolution** is how many bits store each sample.



The faint horizontal lines are the amplitude levels — more bits of resolution means more levels and finer detail.

$$\text{size (bits)} = \text{sampling rate} \times \text{resolution} \times \text{duration} \times \text{channels}$$

Example. A 5-second **mono** (1 channel) clip at 8000 Hz, 16-bit resolution:

$$8000 \times 16 \times 5 \times 1 = 640\,000 \text{ bits} = 80\,000 \text{ bytes} \approx 78.1 \text{ KiB}.$$

Bitmap or vector?

A **vector graphic** stores the *instructions* to draw the image — geometric shapes (lines, curves, circles, polygons), each with attributes like position, size, colour and line width. The program draws ("renders") them at whatever size is needed, so a vector **scales without losing quality**. It cannot, however, capture the detail of a photograph.

Task	Better choice	Why
Photograph	**bitmap**	per-pixel detail can't be described as shapes
Logo, icon, road sign	**vector**	sharp at any size; small file
Engineering drawing	**vector**	exact geometry, scales precisely

2 Practice

Now apply the methods above.

2.1 State what **colour depth** means, and write the formula for a bitmap's file size. [2]

2.2 Calculate the file size of a 1024×768 bitmap at 8 bits per pixel. Give your answer in **KiB**, showing your working. [3]

2.3 A 30-second **mono** sound clip is sampled at 16 kHz (16 000 Hz) with 8-bit resolution. Calculate its file size in **KiB**. [3]

2.4 Give one reason a **vector** graphic is a good choice for a company logo. [1]

3 Exam-style questions

3.1 A photo is 1920×1080 pixels with a colour depth of 24 bits.

(a) Calculate the file size in **MiB**. Show your working. [3]

(b) State **one** change to the image that would reduce its file size, and explain its effect on quality. [2]

3.2 A 3-minute song is recorded in **stereo** (2 channels), sampled at 44.1 kHz with 16-bit resolution.

(a) Calculate the file size in **MiB**. Show your working. [3]

(b) Explain the effect on the recording of **halving** the sampling rate. [2]

3.3 (a) State **two** pieces of data a vector graphic would store to describe a single circle. [2]

(b) A museum scans old paintings to put online. Justify whether a bitmap or a vector format is the better choice. [2]

3.4 Explain why a bitmap logo looks blocky when it is enlarged, but a vector logo stays sharp. [2]

3.6 A school records a 45-second announcement and also stores a scanned photograph of the school badge.

(a) The sound is sampled at 44.1 kHz with a sample resolution of 16 bits, in **stereo** (two

channels). **Estimate** the file size in **mebibytes**, showing your working. [4]

(b) The recording is re-sampled at 22.05 kHz with an 8-bit resolution. **Describe** the effect on the file size and on the **accuracy** of the recording. [3]

(c) The badge is stored as a bitmap 600 pixels wide and 400 pixels high, using a colour depth of 24 bits. **Estimate** the file size in **kibibytes**. [3]

(d) **Explain** why the actual file on disk is **larger** than your answer to (c). [2]

(e) The badge uses only 6 distinct colours in large flat areas. **Explain** how **run-length encoding** would compress it, and **state** why it works far better on this image than on a photograph. [4]

Solutions

2.1 Colour depth = the number of **bits used per pixel**. Size = width × height × colour depth.

2.2 $1024 \times 768 \times 8 = 6\,291\,456$ bits; $\div 8 = 786\,432$ bytes, $\div 1024 =$ **768 KiB**.

2.3 $16\,000 \times 8 \times 30 \times 1 = 3\,840\,000$ bits; $\div 8 = 480\,000$ bytes, $\div 1024 =$ **468.75 KiB**.

2.4 e.g. it stays **sharp at any size** (a logo is used large and small) / the file is small.

3.1 (a) $1920 \times 1080 \times 24 = 49\,766\,400$ bits; $\div 8 \div 1024 \div 1024 \approx$ **5.93 MiB**.
(b) e.g. **lower the colour depth** (or resolution) → smaller file, but fewer colours / less detail.

3.2 (a) $44\,100 \times 16 \times 180 \times 2 = 254\,016\,000$ bits; $\div 8 \div 1024 \div 1024 \approx$ **30.28 MiB**.
(b) Fewer samples per second: the file is roughly **half the size**, but high-pitched sounds are lost / quality drops.

3.3 (a) Any two of: centre position (x, y); radius; line/fill colour; line width.
(b) **Bitmap** — a painting has continuous, per-pixel tonal detail that cannot be described as a few shapes.

3.4 A bitmap has a fixed number of pixels; enlarging it makes each pixel bigger, so the squares show. A vector stores shapes and is **re-drawn** at the new size, so edges stay smooth.

3.6 (a) Bits $= 44\,100 \times 16 \times 2 \times 45 = 6.35 \times 10^7$ bits; bytes $= \dfrac{6.35 \times 10^7}{8} = 7.94 \times 10^6$; mebibytes $= \dfrac{7.94 \times 10^6}{1\,048\,576} = 7.6$ MiB.
(b) Halving both the sampling rate and the resolution divides the size by **four**, to about 1.9 MiB. The lower rate captures fewer samples per second, so high frequencies are lost; the smaller resolution records each sample more coarsely, so the waveform is a poorer match to the original and quantisation noise increases.
(c) Bits $= 600 \times 400 \times 24 = 5\,760\,000$; bytes $= 720\,000$; kibibytes $= \dfrac{720\,000}{1024} = 703$ KiB.
(d) The file also stores **metadata** in a header — the dimensions, the colour depth, the file format — and may include a colour palette or padding, none of which the pixel calculation counts.
(e) Run-length encoding replaces each run of identical pixels with **one colour value and a count**. Large flat areas of one colour become very short records, so the file shrinks a great deal. In a photograph adjacent pixels almost always differ slightly, so runs are one pixel long and the encoding stores a count beside every pixel, which can make the file **bigger** than the original.

1 **(a)** A student creates sound recordings.

Making changes to a sound recording may impact the accuracy of the resulting sound file or the size of the file.

Draw **two** lines from each change to the impacts it has on the sound file.

Change	Impact
increase the duration of the recording	the file size gets bigger
increase the sampling rate	no change in the file size
decrease the sampling resolution	the file size gets smaller
	the accuracy of the sound file improves
	no change to the accuracy of the sound file
	the accuracy of the sound file worsens

[3]

(b) The file names of the sound recordings are stored using the ASCII character set.

Explain how text is represented by the ASCII character set.

.....

.....

.....

..... [2]

(c) Give **two** differences between the ASCII and UNICODE character sets.

1 .....

.....

.....

2 .....

.....

.....

[2]

7 A digital video camera records students in a classroom. The data is transferred from the digital video camera over the internet to a server. Artificial Intelligence (AI) is used to analyse the video on the server to identify when students are interacting with the lesson and the teacher.

(a) Describe **one** ethical impact of this use of AI in the classroom.

.....

.....

.....

.....

..... [2]

(b) A video is made of many bitmap images called frames. 30 frames are recorded every second.

Each frame is 4000 pixels wide by 3000 pixels high. The video records using 16-bit colour depth.

Calculate an estimate for the file size for **one** second of the video in gigabytes.

Show your working.

.....

.....

.....

.....

.....

.....

File size ..... gigabytes

[2]

(c) Identify **and** describe **one** method of data verification that can be used when transferring data from the digital video camera to the server.

Method .....

Description .....

.....

.....

.....

[3]

(d) Describe **two** reasons why the server that stores the videos uses magnetic hard disks instead of solid state (flash) memory.

1 .....

.....

.....

2 .....

.....

.....

[4]

(e) The table contains three algorithms that perform data validation. The third algorithm uses the functions `MID` and `LENGTH`.

The functions `MID` and `LENGTH` are defined as follows:

- `MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING` returns a string of length `y` starting at position `x` from `ThisString`
- `LENGTH(ThisString : STRING) RETURNS INTEGER` returns the integer value representing the length of `ThisString`

Complete the table to identify the method of data validation for each algorithm. Each method of data validation must be different.

Algorithm	Method of data validation

`INPUT x`
 	
`IF x < 1 or x > 26 THEN`
 	
`  OUTPUT "Invalid"`
 	
`ENDIF`	

`INPUT x`
 	
`IF x <> 'R' AND x <> 'G' AND x <> 'B' THEN`
 	
`  OUTPUT "Invalid"`
 	
`ENDIF`	

`INPUT x`
 	
`FLAG ← FALSE`
 	
`FOR INDEX ← 1 TO LENGTH(x)`
 	
`  IF MID(x, INDEX, 1) = "@" THEN`
 	
`  FLAG ← TRUE`
 	
`  ENDIF`
 	
`NEXT INDEX`
 	
`IF NOT FLAG THEN`
 	
`  OUTPUT "Invalid"`
 	
`ENDIF`	

[3]

3 A computer stores images and text files.

(a) One of the images is a bitmapped image.

Complete the table by writing the answer for each statement.

Statement	Answer
the term for the smallest element that makes up an image	
the largest number of different colours that can be represented with a bit depth of 8 bits	
the term for the dots per inch (dpi) when an image is displayed	

[3]

(b) The text in the files is stored using the Unicode character set.

(i) Give **two** advantages of using the Unicode character set instead of using the ASCII character set.

1 .....

.....

2 .....

.....

[2]

(ii) The Unicode character '☾' has the binary value: 0010 0111 0110 1110

Convert the binary value for the character '☾' into denary.

..... [1]

(iii) The Unicode character '∑' has the hexadecimal value 2140

Convert the hexadecimal code for the character '∑' into denary.

..... [1]

1 **(a)** A student paints a picture in an art class.

The student takes a photograph of the picture using a digital camera. The digital camera creates an image with a resolution of 2 million pixels and uses a bit depth of 16 bits.

(i) Calculate the file size of the image created by the digital camera in megabytes (MB).

Show your working.

Working space .....

.....

.....

Answer ..... MB

[2]

(ii) The student takes a second photograph with a lower bit depth.

Explain the effect of decreasing the bit depth on the image and on the image file.

Effect on image .....

.....

.....

.....

Effect on image file .....

.....

.....

.....

[4]

(b) A teacher records an audio file of the student playing the piano during a music lesson.

Complete the table by giving the term for each description about sound representation on a computer.

Description	Term
the number of times the amplitude is measured per time interval	
the number of bits used to store each amplitude measurement	
the type of sound wave before it is recorded by a computer	

[3]

(c) The student types a report on a computer in a history lesson.

(i) The computer uses the Unicode character set.

Give **two** characteristics of the Unicode character set.

1 .....

.....

2 .....

.....

[2]

(ii) Some of the school's computers use the extended ASCII character set.

The table gives some characters from the extended ASCII character set, their denary, 8-bit binary and hexadecimal numbers.

Complete the table by filling in the missing numbers.

Character	Denary	8-bit Binary	Hexadecimal
!	33		21
L		01001100	4C
ü	252	11111100	

[3]

7 A student takes a photograph of a science experiment.

(a) The photograph is saved as a bitmapped image.

(i) Define the following bitmap terms.

Colour depth .....

.....

File header .....

.....

[2]

(ii) Explain why changing the image resolution will affect the image quality and file size.

Image quality .....

.....

.....

File size .....

.....

.....

[2]

(iii) Identify **one** lossless method of compressing an image.

..... [1]

(b) The student draws a picture on paper that is scanned into the computer and saved as a vector graphic.

Define the vector graphic terms property and drawing list.

Property .....

.....

Drawing list .....

.....

[2]

2 A computer game is being designed that users will be able to play using a virtual reality (VR) headset.

(a) Complete the description of the principal operation of a VR headset.

A headset can have one or two ..... that output the image to the user. The headset has speakers that output surround sound to give a realistic experience.

The user's head movements are detected using a sensor.

This sensor is a ..... . The data is transmitted to a microprocessor that analyses the data to identify the ..... of movement. Some headsets use ..... that record the user's eye movements for analysis.

[4]

(b) The computer uses a buffer when transmitting data to the VR headset.

Explain how a buffer is used when data is transmitted between the computer and the VR headset.

.....

.....

.....

.....

.....

..... [3]

(c) The VR headset has Electrically Erasable Programmable Read Only Memory (EEPROM).

Explain the benefits of using EEPROM instead of other types of Read Only Memory (ROM) in the VR headset.

.....

.....

.....

.....

.....

..... [3]

(d) The computer can transmit a video made from bitmap images and vector graphic animations to the VR headset.

(i) Describe how the data for a bitmapped image is encoded.

.....

.....

.....

.....

.....

..... [3]

(ii) Describe the contents of a vector graphic drawing list.

.....

.....

.....

..... [2]

(iii) The bitmap video is **not** compressed before transmission to the VR headset.

Give **two** reasons why the video does **not** need to be compressed.

1 .....

.....

2 .....

.....

[2]

A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

1.3 Compression

Name: _____ Class: _____ Date: _____ **Total: 35 marks**

KEY WORDS compression 压缩 • lossy 有损 • lossless 无损 • run-length encoding 行程编码 • bitmap 位图

Objective

Build the skills to answer exam questions on **compression** 压缩 — why it is used, the difference between lossless and lossy, and how text, images and sound are compressed.

You must be able to:

- explain the **need** for compression (storage and **bandwidth** 带宽)
- explain the difference between **lossless** 无损 and **lossy** 有损 compression, and justify which to use
- describe how a text file, bitmap image, vector graphic and sound file can be compressed

1 Worked examples

Study these first. Each one shows the method for a question type used later — follow the steps and you can do the Practice and Exam-style questions yourself.

■ Lossless or lossy?

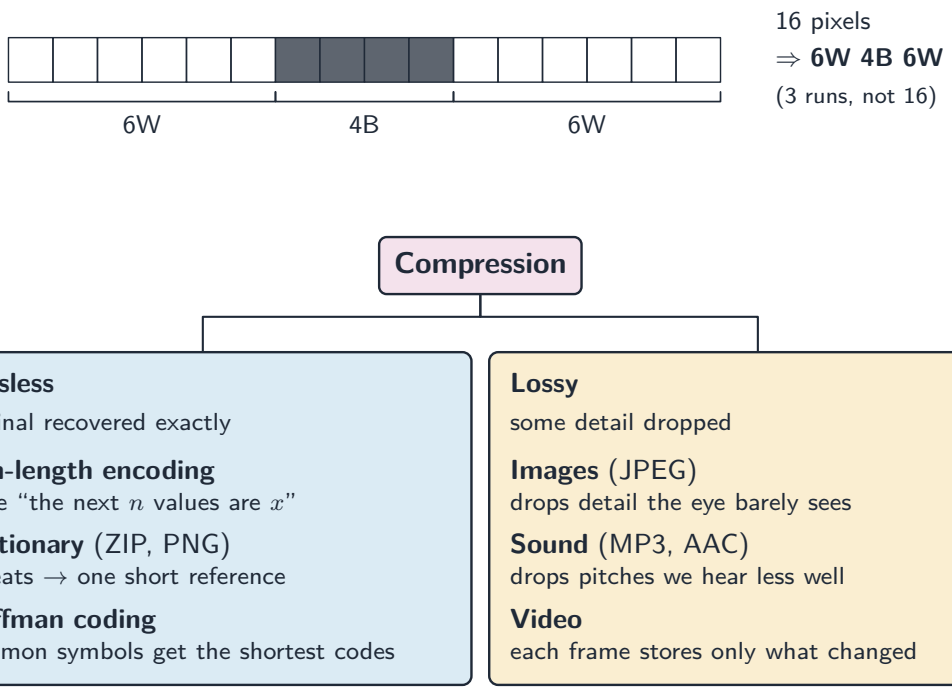
Compression makes a file smaller, so it needs less storage and less bandwidth to send. There are two kinds — the key question in the exam is *which one is suitable and why.*

	keeps every bit?	smaller file?	use for	examples
lossless	yes — exact original back	smaller	text, code, data that must stay exact	ZIP, PNG
lossy	no — drops fine detail	much smaller	photos, music, video	JPEG, MP3

So: choose **lossless** when the data must be recovered exactly; choose **lossy** when a much smaller file matters more than perfect detail (e.g. streaming).

■ Run-length encoding (a lossless method)

Run-length encoding 行程编码 (RLE) stores each *run* of identical values as a (count, value) pair, instead of repeating the value. It shrinks large flat areas a lot, but does nothing for noisy data (where runs are length 1).



Lossless versus lossy compression methods

Compressing text and sound

- **Text (lossless):** a **dictionary** method (ZIP) replaces repeated sequences of characters with a short reference; **Huffman coding** gives the most frequent characters the shortest codes.
- **Sound (lossy, e.g. MP3):** drop pitches the ear barely hears, and quiet sounds masked by louder ones — much smaller, slight quality loss.
- **Video (lossy): spatial** compression shrinks each frame (like JPEG); **temporal** compression stores only what *changed* from the previous frame.

2 Practice

Now apply the methods above.

2.1 State the difference between **lossless** and **lossy** compression. [2]

2.2 Give one situation that needs **lossless** compression and one suited to **lossy** compression, with a reason for each. [2]

2.3 Write the run-length encoding of this pixel row: **W W W W W B B B W** (W = white, B = black). [2]

2.4 State why run-length encoding gives almost no saving on a detailed photograph. [1]

3 Exam-style questions

3.1 A website streams live video. Explain why **lossy** compression is used rather than lossless. [3]

3.2 (a) Describe how run-length encoding compresses a black-and-white bitmap image. [2]

(b) State one kind of image for which run-length encoding gives a large saving. [1]

3.3 Explain how a text file can be compressed **without losing any data**. [2]

3.4 A music app streams songs to phones.

(a) Justify the use of lossy compression for the songs. [2]

(b) State one drawback of using lossy compression here. [1]

3.5 Describe the difference between **spatial** and **temporal** compression in a video file. [3]

3.6 A messaging app compresses the media its users send, and checks every message for transmission errors.

(a) A sound clip is compressed by **reducing the sampling rate**. **Explain** the effect on

the file size and on the quality, and **state** which type of compression this is. [3]

(b) A photograph is compressed with a **lossless** method instead. **Explain** why the app might choose lossless for a scanned document but lossy for a holiday photograph. [3]

(c) The system uses **even parity**. **Complete** each byte by writing the missing parity bit in the space provided. [2]

seven data bits	parity bit
1011010	_____
1110111	_____

(d) The app also uses a **parity block check** with even parity. **Complete** the missing row and column. [3]

	b1	b2	b3	b4	parity
byte 1	1	0	1	1	_____
byte 2	0	1	1	0	_____
byte 3	1	1	0	1	_____
parity byte	_____	_____	_____	_____	

(e) **Explain** why a parity **block** check can locate a single flipped bit, while a parity **byte** check can only detect that something is wrong. [3]

Solutions

2.1 Lossless: the original data is recovered **exactly**. Lossy: some detail is **permanently removed** to make the file smaller.

2.2 Lossless — e.g. a program / document / spreadsheet, because every bit must be exact. Lossy — e.g. a streamed photo / song / video, because a much smaller file matters more than perfect detail.

2.3 5W 3B 1W.

2.4 A photograph has detail that changes pixel to pixel, so runs are only 1 long — storing (count, value) is no shorter.

3.1 Video is a huge amount of data and must be sent in **real time** over limited **bandwidth**; lossless would not shrink it enough, so it would keep buffering/freezing; lossy makes it small enough to stream with only minor quality loss.

3.2 (a) Each row is split into runs of the same colour; each run is stored as a (count, colour) pair instead of every pixel, so long runs of one colour take far less space.
(b) e.g. a simple logo / icon / large areas of one colour.

3.3 Use a lossless method: a dictionary replaces repeated character sequences with a short reference, **or** Huffman coding gives frequent characters shorter codes; the exact text can still be rebuilt.

3.4 (a) Songs are large and streamed over mobile data, so a much smaller file downloads faster and uses less data; the listener barely notices the dropped detail.
(b) e.g. some audio quality is lost permanently.

3.5 Spatial compression reduces the data **within a single frame** (like a JPEG image). Temporal compression stores only the **differences between frames**, since most of one frame is the same as the previous one.

3.6 (a) Fewer samples are taken each second, so there is less data and the file is **smaller**; the higher frequencies can no longer be represented, so the sound is duller and less like the original. Data has been thrown away permanently, so this is **lossy** compression.
(b) A scanned document must stay **exactly** readable — losing detail can turn one character into another, and the image is mostly flat white, which lossless methods compress very well anyway. A photograph has continuous tones where small changes are invisible to the eye, so lossy compression saves far more space at no noticeable cost.
(c) `1011010` has four 1s, already even, so the parity bit is `0`. `1110111` has six 1s, also even, so the parity bit is `0`.
(d) Parity bits by row: byte 1 has three 1s so its parity bit is `1`; byte 2 has two 1s so `0`; byte 3 has three 1s so `1`. The parity byte holds the column parities: columns are `1,0,1` → `0`; `0,1,1` → `0`; `1,1,0` → `0`; `1,0,1` → `0`.
(e) A single byte's parity bit only says that the number of 1s is wrong — any one of the bits could be the culprit. In a block check the flipped bit breaks the parity of **both** its row and its column, and the row and column intersect at exactly one position, so the faulty bit can be found and corrected.

6 **(a)** A sound file is compressed by reducing the sampling rate.

State whether this is lossless or lossy compression. Justify your choice.

Type of compression .....

Justification .....

.....

.....

.....

[1]

(b) The following table shows some words and corresponding denary values.

Word	Denary value
Computing	55
Science	56
Computers	57
are	58
Brilliant!	59
is	60
Fun!	61
Amazing!	62

The following table shows three bytes of data that have been received.

Use the table to find the corresponding words from the binary values received.

Binary value	00111000	00111100	00111110
Word			

Working .....

.....

.....

.....

[1]

(c) A computer system uses even parity. The least significant (rightmost) bit of each byte is the parity bit.

(i) Complete the byte by writing the missing parity bit:

parity bit ↓

0	1	0	1	1	1	0	

[1]

(ii) The computer also uses parity block check. The parity block check uses even parity. Computer A transmits four bytes of data to computer B, followed by a parity byte. Computer B receives the following sequence of bytes.

parity bit ↓

	1	0	1	1	0	1	1	1
	0	1	1	1	0	0	0	0
	0	0	0	1	1	0	1	1
	0	1	1	1	0	1	0	0
parity byte →	1	0	1	0	0	0	0	0

Following transmission, one of the four bytes of data has an error in one of the bits.

Circle the bit that has been altered during the data transfer. [1]

(d) A bitmap image has a resolution of 1000 pixels wide by 2000 pixels high. The colour depth is 16 bits.

Calculate an estimate of the file size in megabytes.

Show your working.

.....

.....

.....

.....

.....

.....

File size ..... megabytes

[2]

2 A program reads string data from a text file into a linear queue and then compresses the data.

The queue is created using the global 1D array `Queue`. The queue can store up to 100 elements. Each element is initialised to the empty string `""`

The queue has two pointers that are global to the program:

- `QueueHead` that points to the index of the first element in the queue, initialised to `-1`
- `QueueTail` that points to the index of the last element in the queue, initialised to `-1`

The global variable `NumberItems` stores the number of elements stored in the queue, initialised to `0`

(a) Write program code to declare and initialise `Queue`, `QueueHead`, `QueueTail` and `NumberItems`

Save your program as **Question2_N25**.

Copy and paste the program code into part **2(a)** in the evidence document.

[2]

(b) The function `Enqueue()` takes a string as a parameter. The function checks if the queue is full, and returns Boolean `FALSE` if the queue is full.

If the queue is not full, the parameter is inserted into the next position in `Queue`. The function updates the appropriate pointer(s), updates `NumberItems` and then returns Boolean `TRUE`

Write program code for `Enqueue()`

Save your program.

Copy and paste the program code into part **2(b)** in the evidence document.

[5]

(c) The function `Dequeue()` returns the string `"False"` if the queue is empty.

If the queue is not empty, the function returns the next element in the queue. The function updates the appropriate pointer(s) and updates `NumberItems`

Write program code for `Dequeue()`

Save your program.

Copy and paste the program code into part **2(c)** in the evidence document.

[3]

(d) The text file `BinaryData.txt` stores individual binary digits, `'1'` and `'0'`. Each digit is on a new line. For example, the first line in the text file stores `'1'`, the second line stores `'1'`

The procedure `ReadData()` reads in each line from the text file `BinaryData.txt` and inserts it into the queue using the appropriate method.

The procedure needs to work for a text file with any number of lines up to a maximum of 100.

Write program code for `ReadData()`

Save your program.

Copy and paste the program code into part **2(d)** in the evidence document.

[5]

(e) The string data in the text file is compressed.

The compression algorithm counts the number of times each binary digit appears consecutively, then stores the binary digit followed by the number of times it appears. The algorithm stores the compressed data in a single string.

For example, if the text file contains the data:

```
1
1
0
0
0
1
1
1
```

The compression algorithm will create the string `"120313"` because there are two `'1'` digits, followed by three `'0'` digits, followed by three `'1'` digits.

The procedure `Compress()` uses `Dequeue()` to remove each element from the queue in turn. The procedure then compresses the data following the compression algorithm described. The new compressed string is stored in the global variable `NewString`

You can assume that one binary digit will never appear more than nine times consecutively in the sequence.

You can assume that there will always be at least one item in the queue.

Write program code for `Compress()`

Save your program.

Copy and paste the program code into part **2(e)** in the evidence document.

[6]

(f) **(i)** Write program code for the main program to:

- call `ReadData()`
- call `Compress()`
- output the content of the compressed string.

Save your program.

Copy and paste the program code into part **2(f)(i)** in the evidence document.

[2]

(ii) Test your program.

Take a screenshot of the output(s).

Save your program.

Copy and paste the screenshot(s) into part **2(f)(ii)** in the evidence document.

[1]

6 A computer system stores text, images and sound.

(a) A character set is used to represent characters in a computer.

Identify **and** describe **one** character set.

Character set .....

Description .....

.....

[2]

(b) The colour of each pixel in a bitmapped image is represented by 8 bits.

(i) State the largest number of different colours that can be represented by 8 bits.

..... [1]

(ii) State **one** drawback of increasing the number of bits that represents each pixel in the bitmap image.

.....

..... [1]

(iii) A bitmap image can be compressed using lossy compression.

Explain the reasons why lossy compression is often suitable for a bitmap image.

.....

.....

.....

..... [2]

(c) (i) Explain how an analogue sound wave is converted into digital data.

.....

.....

.....

..... [2]

(ii) Describe **one** method of compressing a sound file using lossy compression.

.....

.....

.....

..... [2]

7 A student takes a photograph of a science experiment.

(a) The photograph is saved as a bitmapped image.

(i) Define the following bitmap terms.

Colour depth .....

.....

File header .....

.....

[2]

(ii) Explain why changing the image resolution will affect the image quality and file size.

Image quality .....

.....

.....

File size .....

.....

.....

[2]

(iii) Identify **one** lossless method of compressing an image.

..... [1]

(b) The student draws a picture on paper that is scanned into the computer and saved as a vector graphic.

Define the vector graphic terms property and drawing list.

Property .....

.....

Drawing list .....

.....

[2]