Exercise walk-through

Compression ▪ 1.3

eXercise

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

Method — 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*.

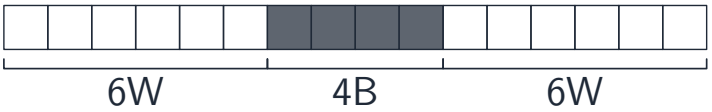
	keeps every bit?	**smaller file?**	**use for**	**examples**
lossless	yes — exact original back	smaller	text, code, data that must stay exact	ZIP, PNG
lossy	no — drops fine detail	much smaller	photos, music, video	JPEG, MP3

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).

Method — 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).

Method — Run-length encoding (a lossless method)

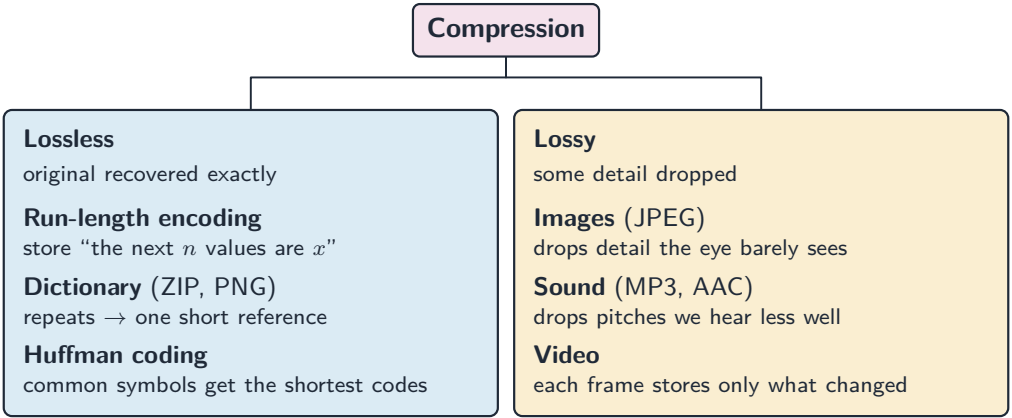


16 pixels

$\Rightarrow$ **6W 4B 6W**

(3 runs, not 16)

Method — Run-length encoding (a lossless method)



Lossless versus lossy compression methods

Method — 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.

Practice 2.1 ▪ 2 marks

State the difference between **lossless** and **lossy** compression.

Solution 2.1

Worked steps

Lossless: the original data is recovered **exactly** B1. Lossy: some detail is **permanently removed** to make the file smaller B1.

Practice 2.2 ▪ 2 marks

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

Solution 2.2

Method: Lossless or lossy?

Worked steps

Lossless — e.g. a program / document / spreadsheet, because every bit must be exact B1. Lossy — e.g. a streamed photo / song / video, because a much smaller file matters more than perfect detail B1.

Practice 2.3 ▪ 2 marks

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

Solution 2.3

Method: Run-length encoding (a lossless method)

Worked steps

5W 3B 1W M1 A1 *for grouping the runs.*

Practice 2.4 ▪ 1 mark

State why run-length encoding gives almost no saving on a detailed photograph.

Solution 2.4

Method: Run-length encoding (a lossless method)

Worked steps

A photograph has detail that changes pixel to pixel, so runs are only 1 long — storing (count, value) is no shorter B1.

Exam-style 3.1 ▪ 3 marks

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

Solution 3.1

Method: Lossless or lossy?

Worked steps

Video is a huge amount of data and must be sent in **real time** over limited **bandwidth** B1; lossless would not shrink it enough, so it would keep buffering/freezing B1; lossy makes it small enough to stream with only minor quality loss B1.

Exam-style 3.2 ▪ 2 marks

(a) Describe how run-length encoding compresses a black-and-white bitmap image.
(b) State one kind of image for which run-length encoding gives a large saving.

Solution 3.2

Method: Run-length encoding (a lossless method)

Worked steps

(a) Each row is split into runs of the same colour; each run is stored as a (count, colour) pair instead of every pixel **B1**, so long runs of one colour take far less space **B1**.

(b) e.g. a simple logo / icon / large areas of one colour **B1**.

Exam-style 3.3 ▪ 2 marks

Explain how a text file can be compressed **without losing any data**.

Solution 3.3

Method: Lossless or lossy?

Worked steps

Use a lossless method: a dictionary replaces repeated character sequences with a short reference, **or** Huffman coding gives frequent characters shorter codes B1; the exact text can still be rebuilt B1.

Exam-style 3.4 ▪ 2 marks

A music app streams songs to phones.
(a) Justify the use of lossy compression for the songs.
(b) State one drawback of using lossy compression here.

Solution 3.4

Method: Lossless or lossy?

Worked steps

(a) Songs are large and streamed over mobile data, so a much smaller file downloads faster and uses less data B1; the listener barely notices the dropped detail B1.
(b) e.g. some audio quality is lost permanently B1.

Exam-style 3.5 ▪ 3 marks

Describe the difference between **spatial** and **temporal** compression in a video file.

Solution 3.5

Method: Compressing text and sound

Worked steps

Spatial compression reduces the data **within a single frame** (like a JPEG image) B1. Temporal compression stores only the **differences between frames** B1, since most of one frame is the same as the previous one B1.

Exam-style 3.6 ▪ 3 marks

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

seven data bits	**parity bit**
1011010	_____
1110111	_____

Exam-style 3.6 ▪ 3 marks

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

	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	_____	_____	_____	_____	

Exam-style 3.6 ▪ 3 marks

(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.

Solution 3.6

Method: Lossless or lossy?

Worked steps

(a) Fewer samples are taken each second, so there is less data and the file is **smaller** B1; the higher frequencies can no longer be represented, so the sound is duller and less like the original B1. Data has been thrown away permanently, so this is **lossy** compression B1.
(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 B1 B1. A photograph has continuous tones where small changes are invisible to the eye, so lossy compression saves far more space at no

Solution 3.6

noticeable cost **B1**.
(c) 1011010 has four 1s, already even, so the parity bit is 0 **B1**. 1110111 has six 1s, also even, so the parity bit is 0 **B1**.
(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 **B1** **B1**. The parity byte holds the column parities: columns are 1,0,1 $\rightarrow$ 0; 0,1,1 $\rightarrow$ 0; 1,1,0 $\rightarrow$ 0; 1,0,1 $\rightarrow$ 0 **B1**.
(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 **B1**. In a block check the flipped bit breaks the parity of **both** its row and its column **B1**, and the row and column intersect at exactly one position, so the faulty bit can be found and corrected **B1**.