

Exercise walk-through

# 2.4

## Using Programs with Data

---

AP Computer Science Principles

# Using programs with data

A program **processes** data: records in, new information out.

## Iterate to summarise

A loop uses **iteration** to visit each **element** : summing 70, 85, 90, 60, 95 gives  $400; \div 5 = 80$  (the average).

A **filter** keeps a **subset** that meets a condition; programs also **combine** sources, **transform** values, and **clean** first – dirty data gives a wrong answer.

## Worked example – choosing a compression

### Which compression for a source-code file? For a holiday photo?

**Code** → **lossless**. Every bit matters – a single changed character breaks the program. Restoring the **exact** original is non-negotiable. **Photo** → **lossy**. A small quality loss is invisible, and the file gets **much** smaller.

The trade is **size against fidelity**. Lossy loss is **permanent** – you can never get the discarded data back.

# Key points

**Binary** place values are powers of two; watch **overflow**

**Lossless** keeps every bit;  
**lossy** trades quality for size

**Correlation** is not **causation**; clean data first

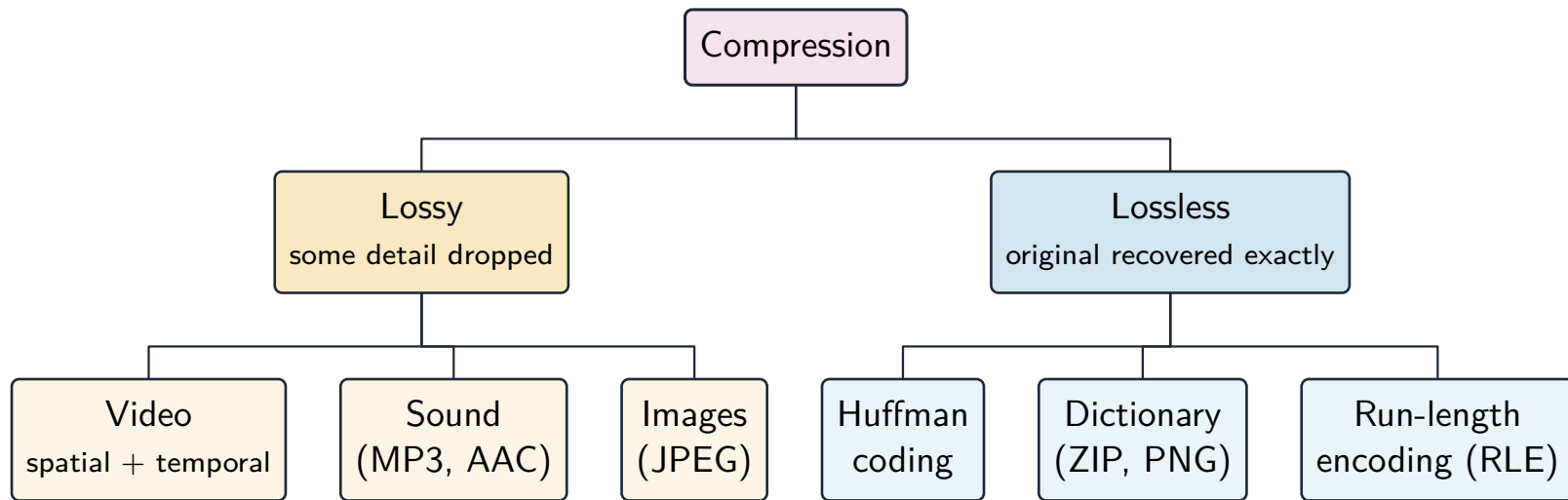
Programs **iterate**, **filter** & transform data

# Binary Numbers

```
01000100 01000101 01010011 01010000 01001001 01000101 01010010 01010100 01000001 00100001 00100000 01001110
01001111 01010011 00100000 01000101 01010011 01010100 11000011 10000001 01001110 00100000 01001101 01000001
01001110 01001001 01010000 01010101 01001100 01000001 01001110 01000100 01001111 00100000 01000001 00100000
01010011 01010101 00100000 01000001 01001110 01010100 01001111 01001010 01001111 00101110 00100000 01000101
01001100 00100000 01001110 01010111 01001111 00100000 01001001 01001101 01010000 01001100 01001001 01000011
01000001 00100000 01001100 01000001 00100000 01010000 01000101 01010010 01000100 01001001 01000100 01000001
00100000 01000100 01000101 00100000 01010100 01001111 01000100 01000001 01010011 00100000 01001110 01010101
01000101 01010011 01010100 01010010 01000001 01010011 00100000 01001100 01001001 01000010 01000101 01010010
01010100 01000001 01000100 01000101 01010011 00100000 01011001 00100000 01000101 01001100 00100000 01000011
01001111 01001110 01010100 01010010 01001111 01001100 00100000 01001101 01001001 01001110 01010101 01000011
01001001 01001111 01010011 01001111 00100000 01000100 01000101 00100000 01010100 01001111 01000100 01000001
00100000 01001100 01000001 00100000 01010000 01001111 01000010 01001100 01000001 01000011 01001001 11000011
10010011 01001110 00100000 01000001 01010011 11000011 10001101 00100000 01000011 01001111 01001101 01001111
00100000 01010101 01001110 01000001 00100000 01010010 01000101 01000100 01010101 01000011 01000011 01001001
11000011 10010011 01001110 00100000 01000100 01010010 11000011 10000001 01010011 01010100 01001001 01000011
01000001 00100000 01000100 01000101 00100000 01001100 01000001 00100000 01010000 01001111 01000010 01001100
01000001 01000011 01001001 11000011 10010011 01001110 00100000 01001101 01010101 01001110 01000100 01001001
01000001 01001100 00100001 00100000 01010010 01000101 01010011 01001001 01010011 01010100 01000101 01001110
01000011 01001001 01000001 00100000 01001001 01001110 01010100 01000101 01001100 01000101 01000011 01010100
01010101 01000001 01001100 00100001 01100100 01100101 01110011 01110000 01100101 01110010 01110100 01100001
01100100 00100001 01100100 01100101 01110011 01110000 01100101 01110010 01110100 01100001 01100100 00100001
01100100 01100101 01110011 01110000 01100101 01110010 01110100 01100001 01100100 00100001 01100100 01100101
01110011 01110000 01100101 01110010 01110100 01100001 01100100 00100001 01100100 01100101 01110011 01110000
01100101 01110010 01110100 01100001 01100100 00100001 01100100 01100101 01110011 01110000 01100101 01110010
01110100 01100001 01100100 00100001 01100100 01100101 01110011 01110000 01100101 01110010 01110100 01100001
01100100 00100001 01100100 01100101 01110011 01110000 01100101 01110010 01110100 01100001 01100100 00100001
```

Binary digits on a display – all digital data is ultimately stored as 0s and 1s

## Check yourself, in detail



Compression methods: lossless versus lossy, with common examples

## Exercises 2.1-2.3

---

**2.1** State what a filter does to a data set.

[1]

---

**2.2** State why a program uses iteration on a data set.

[1]

---

**2.3** State how cleaning data improves the results.

[1]

## Exercises 2.1-2.3

---

**2.1** State what a filter does to a data set.

[1]

**2.1**

- it selects the subset of records that meet a given condition
- 

**2.2** State why a program uses iteration on a data set.

[1]

**2.2**

- to examine or process each element in turn
- 

**2.3** State how cleaning data improves the results.

[1]

**2.3**

- it removes errors and inconsistencies, so the results are more reliable

## Exercises 3.1-3.2

---

**3.1** A filter applied to a data set

[1]

☐

- A** deletes all the data
- B** selects the records that meet a condition
- C** sorts the data randomly
- D** adds random noise

---

**3.2** Looping through every record in a list is an example of

[1]

☐

- A** a filter
- B** iteration
- C** a variable
- D** metadata

## Exercises 3.1-3.2

---

**3.1** A filter applied to a data set

[1]

☐

- A deletes all the data
- B selects the records that meet a condition
- C sorts the data randomly
- D adds random noise

**3.1 B**

- selects the records that meet a condition.

---

**3.2** Looping through every record in a list is an example of

[1]

☐

- A a filter
- B iteration
- C a variable
- D metadata

**3.2 B**

- iteration.

## Exercise 3.3 (a)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(a) Name the operation used to keep only teenagers.

[1]

## Exercise 3.3 (a)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(a) Name the operation used to keep only teenagers.

[1]

(a)

- a filter

## Exercise 3.3 (b)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(b) Name the process of looping over all the records.

[1]

## Exercise 3.3 (b)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(b) Name the process of looping over all the records.

[1]

(b)

- iteration

## Exercise 3.3 (c)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(c) State one reason to clean the data first.

[1]

## Exercise 3.3 (c)

---

**3.3** A program reads 10 000 survey records and keeps only those from teenagers.

(c) State one reason to clean the data first.

[1]

(c)

- to remove errors or duplicates for reliable results