ANSWERS

F9.7 Secondary storage

Logic and computer systems · GCSE · OCR J277 1.2.2, AQA 8525 3.4.5, Edexcel 1CP2 3.1.2 · about 15 min

What this lesson is about

Magnetic, optical, solid state and cloud storage, and choosing between them.

- Why secondary storage?
- Three technologies
- Choosing storage
- Cloud storage
- Speed matters too

Questions

6 marks in all

1. Why do computers need secondary storage? [1 mark]

- ☐ **A** It keeps programs and files when the power is off
- ☐ **B** It is faster than RAM
- ☐ **C** The CPU runs programs directly from it
- ☐ **D** It holds the boot program

Answer: A. Secondary storage is non-volatile.

2. Which storage suits a robot that vibrates across a mat? [1 mark]

- ☐ **A** Solid state, because it has no moving parts
- ☐ **B** A hard disk, because it has high capacity
- ☐ **C** Optical, because discs are cheap
- ☐ **D** Magnetic tape, because it is reliable

Answer: A. Durability matters most: a spinning disk could be damaged.

3. How does optical storage read data? [1 mark]

- ☐ **A** A laser reads pits and flat areas on a disc
- ☐ **B** A head reads magnetised areas
- ☐ **C** Electrical charges are read from chips
- ☐ **D** A camera photographs the disc

Answer: A. CDs, DVDs and Blu-ray are optical.

4. What is an advantage of a hard disk over an SSD? [1 mark]

- ☐ **A** Lower cost per gigabyte
- ☐ **B** Faster access
- ☐ **C** No moving parts
- ☐ **D** Lighter

Answer: A. Hard disks are cheap for large capacities.

5. What is a drawback of cloud storage? [1 mark]

- ☐ **A** It needs an internet connection
- ☐ **B** Files cannot be shared
- ☐ **C** It cannot be reached from a phone
- ☐ **D** It has a fixed, small capacity

> **Answer:** A. No connection, no files; the data is also held by another company.

6. Copying a 3000 MB file at 150 MB per second takes how many seconds? [1 mark]

> **Answer:** 20. 3000 ÷ 150.

The task: the storage chooser

Write `choose(needed_gb, no_moving_parts)`. It returns the name of the cheapest device in `devices` whose capacity is at least `needed_gb`, and which has no moving parts if `no_moving_parts` is True. A device's price in pence is its capacity times its pence per GB. Print one line for each job in `jobs`, in the form `robot logs: SD card`.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

# name, capacity in GB, pence per GB, has moving parts
devices = [
    ("hard disk", 2000, 2, True),
    ("SSD", 1000, 6, False),
    ("SD card", 256, 10, False),
    ("Blu-ray", 50, 4, True),
]
# job, GB needed, needs no moving parts
jobs = [("robot logs", 64, True), ("school backups", 1500, False), ("video editing", 500,
True), ("film archive", 40, False)]
```

The hint students can ask for: Look at every device and skip the ones that are too small or that have moving parts when the job forbids them. Of those left, keep the one whose price, its capacity times its price per gigabyte, is lowest.

A solution

```
from bugbot import *
connect()
devices = [
    ("hard disk", 2000, 2, True),
    ("SSD", 1000, 6, False),
    ("SD card", 256, 10, False),
    ("Blu-ray", 50, 4, True),
]
jobs = [("robot logs", 64, True), ("school backups", 1500, False), ("video editing", 500,
True), ("film archive", 40, False)]

def choose(needed_gb, no_moving_parts):
    best = None
    best_price = 0
    for name, capacity, pence, moving in devices:
        if capacity < needed_gb or (no_moving_parts and moving):
            continue
        price = capacity * pence
        if best is None or price < best_price:
            best = name
            best_price = price
    return best

for job, gb, still in jobs:
    print(f"{job}: {choose(gb, still)}")
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.