ANSWERS

F9.9 Operating systems and utilities

Logic and computer systems · GCSE · OCR J277 1.5.1, AQA 8525 3.4.3, Edexcel 1CP2 3.2.1 · about 15 min

What this lesson is about

What an operating system does, scheduling, and utility software.

- What an operating system does
- Sharing the processor
- Utility software

Questions

6 marks in all

1. What is a device driver? [1 mark]

- ☐ **A** A program that lets the operating system talk to a particular device
- ☐ **B** A person who fixes hardware
- ☐ **C** A utility that speeds up the disk
- ☐ **D** The part of the CPU that controls devices

> **Answer:** A. Each printer or camera needs its own driver.

2. How does an operating system run several programs on one core? [1 mark]

- ☐ **A** It gives each a short time slice and switches between them quickly
- ☐ **B** It runs them all at the same instant
- ☐ **C** It runs one program to the end before starting the next
- ☐ **D** It copies each program to a separate core

> **Answer:** A. The switching is so fast they seem to run at once.

3. What does defragmentation do? [1 mark]

- ☐ **A** Moves the pieces of each file on a hard disk back together
- ☐ **B** Removes viruses
- ☐ **C** Makes files smaller
- ☐ **D** Copies files to another drive

> **Answer:** A. Fewer head movements means faster reading.

4. Why should an SSD not be defragmented? [1 mark]

- ☐ **A** It gains no speed and the extra writes wear it out
- ☐ **B** It deletes the files
- ☐ **C** SSDs cannot store files in pieces
- ☐ **D** It makes the SSD volatile

> **Answer:** A. There is no moving head to benefit.

5. What does an incremental backup copy? [1 mark]

- ☐ **A** Only the files changed since the last backup
- ☐ **B** Every file on the computer
- ☐ **C** Only the operating system
- ☐ **D** Only files that have been deleted

> **Answer:** A. Faster than a full backup, but restoring needs every backup.

6. What does this program print? [1 mark]

```
queue = ["a", "b", "c"]
job = queue.pop(0)
queue.append(job)
print(queue)
```

> **Answer:**
>
> ```
> ['b', 'c', 'a']
> ```
>
> The job at the front moves to the back: round robin.

The task: a round-robin scheduler

Write a round-robin scheduler with a time slice of 2 units. Each job in `jobs` needs some units of processor time. Take the job at the front of the queue, run it for 2 units or for what it has left, whichever is less, and print `<name> ran for <n>`. If it is not finished, put it at the back of the queue; if it is, print `<name> finished at <t>`, where `t` is the total time so far.

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

jobs = [("motors", 3), ("camera", 5), ("radio", 2)]
time_slice = 2
```

The hint students can ask for: Take the job at the front of the queue and run it for the slice, or for what it has left if that is less. Add the time on, then either put it back at the end of the queue or announce it has finished.

A solution

```
from bugbot import *
connect()
jobs = [("motors", 3), ("camera", 5), ("radio", 2)]
time_slice = 2
queue = list(jobs)
t = 0
while queue:
    name, left = queue.pop(0)
    run = min(time_slice, left)
    t = t + run
    print(name, "ran for", run)
    if left - run > 0:
        queue.append((name, left - run))
    else:
        print(name, "finished at", t)
```

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