ANSWERS

F5.7 Bubble sort

Algorithms · GCSE · OCR J277 2.1.3, AQA 8525 3.1.4, Edexcel 1CP2 1.2.6 · about 15 min

What this lesson is about

Passes and swaps, stopping early, and hearing a tune become a scale.

- The algorithm
- In Python
- Hear it sort
- How much work it does

Questions

5 marks in all

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

```
items = [5, 2, 4, 1]
for i in range(len(items) - 1):
    if items[i] > items[i + 1]:
        items[i], items[i + 1] = items[i + 1], items[i]
print(items)
```

Answer:

```
[2, 4, 1, 5]
```

One pass: the largest item, 5, bubbles to the end.

2. When can bubble sort stop early? [1 mark]

- ☐ **A** After a whole pass with no swaps
- ☐ **B** After the first swap
- ☐ **C** When it reaches the middle of the list
- ☐ **D** It can never stop early

Answer: A. A pass with no swaps proves the list is already sorted.

3. What does bubble sort compare? [1 mark]

- ☐ **A** Each pair of neighbouring items
- ☐ **B** Each item with the first item
- ☐ **C** The middle item with the target
- ☐ **D** Two halves of the list

Answer: A. It compares neighbours and swaps them if they are in the wrong order.

4. A list of 6 items. How many comparisons does one pass of bubble sort make? [1 mark]

Answer: 5. n - 1 pairs of neighbours in a list of n items.

5. Why does a swap in pseudocode usually need a temporary variable? [1 mark]

- ☐ **A** Copying one item over the other would lose the first value
- ☐ **B** Pseudocode has no lists
- ☐ **C** It makes the sort faster
- ☐ **D** Python requires it

> **Answer:** A. temp = a; a = b; b = temp keeps both values. Python's a, b = b, a does it in one line.

The task: sort the tune

Sort `notes = [392, 262, 523, 330, 294, 440, 349, 494]` with **bubble sort**, written yourself (no `sort` or `sorted`). Then print `sorted: <the list>` and play every note of the sorted list for 0.2 seconds.

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

notes = [392, 262, 523, 330, 294, 440, 349, 494]
```

The hint students can ask for: Go through the list comparing each pair of neighbours and swapping them when they are the wrong way round. Repeat until a whole pass makes no swaps.

A solution

```
from bugbot import *
connect()
notes = [392, 262, 523, 330, 294, 440, 349, 494]
swapped = True
while swapped:
    swapped = False
    for i in range(len(notes) - 1):
        if notes[i] > notes[i + 1]:
            notes[i], notes[i + 1] = notes[i + 1], notes[i]
            swapped = True
print("sorted:", notes)
for note in notes:
    tone(note, 0.2)
```

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