MODULE L STANDARD ALGORITHMS · L3

# Bubble sort

Original practice questions · 21 marks · ~25 minutes · OCR J277 spec 2.1.3 (with 2.1.2 trace tables)

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. The total for this worksheet is **21 marks**. All code is written in OCR Exam Reference Language. Arrays start at index 0.

## 1

**Total: 4 marks**

**(a)** Tick (✓) **one** box in each row to show whether each statement about a bubble sort into ascending order is true or false. **[4]**

| Statement | True | False |
|---|---|---|
| It compares items that are next to each other in the list. | | |
| It can only be used if the data is already partly in order. | | |
| At the end of the first pass, the largest value is in its final position. | | |
| It splits the list into sub-lists that each hold one item. | | |

## 2

**Total: 5 marks**

A café records the number of orders it takes in each hour of one morning:

```
18  7  25  12  9  30
```

The manager uses a bubble sort to put the numbers into ascending order. The bubble sort stops when a pass makes no swaps.

**(a)** Complete the table to show the order of the numbers after each of the first three passes. **[3]**

| | Numbers | | | | | |
|---|---|---|---|---|---|---|
| **Start** | 18 | 7 | 25 | 12 | 9 | 30 |
| **After pass 1** | | | | | | |
| **After pass 2** | | | | | | |
| **After pass 3** | | | | | | |

**(b)** The numbers are in order after pass 3. Explain why the bubble sort still carries out a fourth pass. **[2]**

**3** **Total: 5 marks**

A market stall keeps the prices of its items in the array `prices`. The algorithm below puts the prices in order.

```
01 do
02     swapped = false
03     for i = 0 to prices.length - 2
04         if prices[i] > prices[i + 1] then
05             temp = prices[i]
06             prices[i] = prices[i + 1]
07             prices[i + 1] = temp
08             swapped = true
09         endif
10     next i
11 until swapped == false
```

**(a)** Give **two** features of this code that show it is a bubble sort. **[2]**

1

2

**(b)** Describe the purpose of the variable `swapped` in this algorithm. **[2]**

**(c)** The stall owner wants the prices sorted into **descending** order instead. Rewrite line 04 so that the algorithm does this. **[1]**

04

**4** **Total: 7 marks**

A weather station stores four wind speed readings. The code below carries out **one pass** of a bubble sort and then outputs a value.

```
01 array speeds = [14, 9, 21, 6]
02 swaps = 0
03 for i = 0 to 2
04     if speeds[i] > speeds[i + 1] then
05         temp = speeds[i]
06         speeds[i] = speeds[i + 1]
07         speeds[i + 1] = temp
08         swaps = swaps + 1
09     endif
10 next i
11 print(swaps)
```

**(a)** Complete the trace table for this code. You may not need to use all the rows. **[4]**

| i | temp | swaps | Output |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

**(b)** State the contents of the array `speeds` after the code has finished. **[1]**

**(c)** Explain why lines 05 to 07 use the variable `temp` to swap two items. **[2]**

**Total for this worksheet: 21 marks**