01 The algorithm in Figure 1 is a sorting algorithm

- array indexing starts at 0
- line numbers are included but are not part of the algorithm

Figure 1

```
1    arr <- [7, 1, 2]
2    sorted <- false
3    WHILE sorted = false
4        sorted <- true
5        i <- 0
6        WHILE i < 2
7            IF arr[i+1] < arr[i] THEN
8                t <- arr[i]
9                arr[i] <- arr[i+1]
10               arr[i+1] <- t
11               sorted <- false
12           ENDIF
13           i <- i + 1
14       ENDWHILE
15   ENDWHILE
```

01.1 Choose the line number where **selection** is first used in the algorithm shown in **Figure 1**.

[1 mark]

Line 2 Line 3 Line 7 Line 12

01.2 Choose the line number where **iteration** is first used in the algorithm shown in **Figure 1**.

[1 mark]

Line 1 Line 3 Line 7 Line 13

01.3 Choose the line number where **variable initialisation** occurs in the algorithm shown in **Figure 1**.

[1 mark]

Line 1 Line 3 Line 7 Line 13

01.4 State the data type of the variable `sorted` in the algorithm shown in **Figure 1**.

[1 mark]

Boolean

01.5 State the value of `arr[2]` in the algorithm shown in Figure 1.

[1 mark]

2

01.5 The identifier `sorted` is used in the algorithm shown in Figure 1.
Explain why this is a better choice than using the identifier `s`.

[2 marks]

clearer what the variable is used for [1 mk]

easier for other programmers to understand the algorithm [1 mk] when working as a team [1] when maintaining the code [1]

01.6 Identify which of the following is the only false statement about the algorithm in Figure 1.

[1 mark]

A. The algorithm uses nested iteration

B. The algorithm uses indefinite iteration

C. The algorithm uses a constant (NOTE: this is the FALSE statement)

01.7 Study **line 6** of the algorithm shown in **Figure 1**. Explain what this line does.

[2 marks]

sets up a while loop [1] acts as loop control [1]

loops over the code from lines 7 to 13/14 [1]

repeats lines 7 to 13/14 twice [1]

01.8 Study **line 7** of the algorithm shown in **Figure 1**. Explain what this line does.

[2 marks]

checks to see if arr[i+1] is less than arr[i]

controls whether lines 8-11/12 are executed

compares the current value in arr with the next one

01.9 Study **line 8** of the algorithm shown in **Figure 1**. What value will `t` take the first time the algorithm is run?

[1 mark]

1

01.A What is the purpose of the variable `t` on **line 8** of the algorithm shown in **Figure 1**?

[1 mark]

to allow the two values to be swapped over [1] as a temporary store for arr[i] to store arr[i] so that it can be swapped with arr[i+1]