CAMBRIDGE IGCSE (0478)

Module L: Standard algorithms

Workbooklet · 3 parts · 62 marks

L2, L4, L5 and L6 are GCSE (OCR) only and not on 0478.

Contents

Part	Topic	Marks
L1	Linear search	21
L3	Bubble sort	20
L7	Totalling, counting, maximum, minimum and average	21
	Total	**62**

MODULE L STANDARD ALGORITHMS · L1

Linear search

Original practice questions · 21 marks · ~25 minutes · Cambridge 0478 syllabus 7.4, 7.7, 7.9

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**. Write all algorithms in pseudocode. The first element of every array on this sheet is at index 1. All data on this sheet is made up.

1

Total: 4 marks

(a) Describe how a linear search is used to find a value in a 1D array. **[3]**

(b) A linear search is carried out on an array of 40 elements. The value being searched for is **not** in the array. State how many elements are checked. **[1]**

2

Total: 3 marks

A bakery stores the batch numbers of today's loaves in the array `Batch[1:8]`.

Index	1	2	3	4	5	6	7	8
Batch	507	213	880	145	213	692	330	471

(a) A linear search that stops when the value is found is used to look for the batch number 145. State, in order, every value in `Batch` that is compared with 145. **[2]**

(b) State the index of the element where 145 is found. **[1]**

3

Total: 5 marks

A wildlife centre puts a numbered tag on each animal. The array `Tag[1:6]` holds these values:

Index	1	2	3	4	5	6
Tag	31	18	44	18	27	52

This algorithm searches `Tag` for a value that is input.

```
01 Position ← 0
02 Index ← 1
03 INPUT Target
04 REPEAT
05     IF Tag[Index] = Target
06       THEN
07         Position ← Index
08     ENDIF
09     Index ← Index + 1
10 UNTIL Position <> 0 OR Index > 6
11 IF Position = 0
12   THEN
13     OUTPUT "Not found"
14   ELSE
15     OUTPUT "Found at ", Position
16 ENDIF
```

(a) Complete the trace table for the algorithm using the input data: **27** **[3]**

Index	**Position**	**OUTPUT**

(b) State the output when the input is **60** **[1]**

(c) State how many times line 05 is executed when the input is **60** **[1]**

4 **Total: 6 marks**

A builders' merchant stores the codes of 200 products in the 1D array `Code[1:200]` and the price of each product in the 1D array `Price[1:200]`. The price of the product in `Code[1]` is in `Price[1]`, and so on.

(a) Write an algorithm using pseudocode to: **[6]**

- input a product code
- use a linear search to look for the product code in `Code`
- stop searching as soon as the product code is found
- output the price of the product if the code is found
- output a suitable message if the code is not found.

You do not need to declare the arrays.

5 **Total: 3 marks**

The algorithm in Question 3 only finds the **first** element of `Tag` that matches `Target`.

(a) Explain how the algorithm in Question 3 could be changed so that it outputs the index of **every** element that matches `Target`, and still outputs "Not found" if there is no match. Any code you use must be fully explained. **[3]**

Total for this worksheet: 21 marks

MODULE L STANDARD ALGORITHMS · L3

Bubble sort

Original practice questions · 20 marks · ~25 minutes · Cambridge 0478 syllabus 7.4, 7.7, 7.8, 7.9

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 **20 marks**. All algorithms are written in pseudocode. Write any code you give in pseudocode.

1

Total: 4 marks

(a) Describe how a bubble sort puts the values stored in an array into ascending order. **[4]**

2

Total: 4 marks

A ferry company records the number of cars carried on six crossings:

`34  21  40  15  28  19`

The values are sorted into ascending order using a bubble sort.

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

	Values					
Start	34	21	40	15	28	19
After pass 1						
After pass 2						
After pass 3						

(b) State the number of swaps made during pass 1. **[1]**

3 **Total: 6 marks**

A plant nursery stores the number of seedlings in each of four trays in the array `Stock[1:4]`. The array already contains these values:

`Stock[1]` = 4 `Stock[2]` = 9 `Stock[3]` = 2 `Stock[4]` = 7

This algorithm sorts the array.

```
01 DECLARE Index : INTEGER
02 DECLARE Temp : INTEGER
03 DECLARE Swapped : BOOLEAN
04 REPEAT
05     Swapped ← FALSE
06     FOR Index ← 1 TO 3
07         IF Stock[Index] > Stock[Index + 1]
08           THEN
09             Temp ← Stock[Index]
10             Stock[Index] ← Stock[Index + 1]
11             Stock[Index + 1] ← Temp
12             Swapped ← TRUE
13         ENDIF
14     NEXT Index
15 UNTIL NOT Swapped
```

(a) Complete the trace table for the algorithm. The first row has been done for you. **[5]**

Swapped	Index	Temp	Stock[1]	Stock[2]	Stock[3]	Stock[4]
			4	9	2	7

(b) The nursery wants the trays sorted into **descending** order. Write the new line 07 needed. **[1]**

07

4 **Total: 3 marks**

A climbing centre stores the lengths, in metres, of five ropes in the array `Lengths[1:5]`. This algorithm should sort the lengths into ascending order, but it contains **three** errors.

```
01 DECLARE Pass : INTEGER
02 DECLARE Position : INTEGER
03 DECLARE Hold : REAL
04 FOR Pass ← 1 TO 4
05     FOR Position ← 1 TO 5
06         IF Lengths[Position] < Lengths[Position + 1]
07           THEN
08             Hold ← Lengths[Position]
09             Lengths[Position] ← Hold
10             Lengths[Position + 1] ← Hold
11         ENDIF
12     NEXT Position
13 NEXT Pass
```

(a) Identify the line numbers of the **three** errors and suggest a correction for each error. **[3]**

Error 1: line ..... Correction

Error 2: line ..... Correction

Error 3: line ..... Correction

5 **Total: 3 marks**

The plant nursery in **Question 3** also stores the name of the plant in each tray in a second array, `Plant[1:4]`. `Plant[1]` is the plant in the tray counted in `Stock[1]`, and so on. When the algorithm in Question 3 sorts `Stock`, each plant name must stay matched to its number of seedlings.

(a) Write the pseudocode statements that must be added to the algorithm in Question 3 so that the names stay matched. State where in the algorithm your statements go. You do not need to declare any variables. **[3]**

Total for this worksheet: 20 marks

MODULE L STANDARD ALGORITHMS · L7

Totalling, counting, maximum, minimum and average

Original practice questions · 21 marks · ~25 minutes · Cambridge 0478 syllabus 7.4, 7.7, 7.8, 7.9, 8.1.4(d)

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 algorithms on this sheet are written in pseudocode. Write your own algorithms in pseudocode.

1

Total: 4 marks

A parcel depot uses these pseudocode statements in its programs.

(a) Draw **one** line from each statement to the standard method of solution it is part of. Not all standard methods will be used. **[4]**

Statement

```
TotalWeight ← TotalWeight + Weight
```

```
IF Weight > Heaviest
  THEN
    Heaviest ← Weight
ENDIF
```

```
Parcels ← Parcels + 1
```

```
MeanWeight ← TotalWeight / Parcels
```

Standard method

- counting
- finding the average
- finding the maximum
- finding the minimum
- linear search
- totalling

2 **Total: 7 marks**

A sensor on a solar farm records the midday temperature, in °C, of a solar panel. This algorithm processes the temperatures for five days.

```
01 Total ← 0
02 Count ← 0
03 Lowest ← 100
04 FOR Day ← 1 TO 5
05     INPUT Reading
06     Total ← Total + Reading
07     IF Reading < Lowest
08       THEN
09         Lowest ← Reading
10     ENDIF
11     IF Reading >= 30
12       THEN
13         Count ← Count + 1
14     ENDIF
15 NEXT Day
16 Average ← Total / 5
17 OUTPUT "Lowest: ", Lowest
18 OUTPUT "Hot days: ", Count
19 OUTPUT "Average: ", Average
```

(a) Complete the trace table for the algorithm using this input data: **[5]**

24, 31, 18, 30, 27

Day	Reading	Total	Lowest	Count	Average	OUTPUT

(b) Explain why `Lowest` is given the value 100 on line 03, and not the value 0. **[2]**

3

Total: 4 marks

A taxi firm stores the distances, in km, of 20 trips in the 1D array `Distance`, declared as `DECLARE Distance : ARRAY[1:20] OF REAL`. All the distances are greater than 0. This algorithm should find the longest trip and count the trips that are longer than 25 km. It contains **four** errors.

```
01 Longest ← 0
02 Count ← 1
03 FOR Trip ← 0 TO 20
04     IF Distance[Trip] < Longest
05       THEN
06         Longest ← Distance[Trip]
07     ENDIF
08     IF Distance[Trip] > 25
09       THEN
10         Count ← Count + Distance[Trip]
11     ENDIF
12 NEXT Trip
13 OUTPUT "Longest trip: ", Longest, " km"
14 OUTPUT "Trips over 25 km: ", Count
```

(a) Identify the line numbers of the **four** errors and suggest a correction for each error. **[4]**

Error 1: line ..... correction .....

Error 2: line ..... correction .....

Error 3: line ..... correction .....

Error 4: line ..... correction .....

4 **Total: 6 marks**

A craft stall at a weekend market wants an algorithm to summarise the items it sells in a day.

(a) Write an algorithm using pseudocode to: **[6]**

- input the price of each item sold, one at a time
- stop when a price of 0 is input
- total the prices and count the number of items sold (the 0 is not an item)
- output the total of the prices and the average price, rounded to two decimal places, with suitable messages.

You do not need to declare any variables.

Total for this worksheet: 21 marks