

**OXFORD CAMBRIDGE AND RSA EXAMINATIONS  
AS LEVEL**

**H046/02**

**COMPUTER SCIENCE**

**Algorithms and problem solving**

**FRIDAY 9 JUNE 2017: Morning**

**TIME ALLOWED: 1 hour 15 minutes  
plus your additional time allowance**

**MODIFIED ENLARGED**

<b>First name</b>		<b>Last name</b>	
-----------------------	--	----------------------	--

<b>Centre number</b>						<b>Candidate number</b>				
--------------------------	--	--	--	--	--	-----------------------------	--	--	--	--

**DO NOT USE:  
a calculator**

**READ INSTRUCTIONS OVERLEAF**



## **INSTRUCTIONS**

**Use black ink.**

**Complete the boxes on the front page with your name, centre number and candidate number.**

**Answer ALL the questions.**

**Write your answer to each question in the space provided.**

**Additional paper may be used if required but you must clearly show your candidate number, centre number and question number(s).**

## **INFORMATION**

**The total mark for this paper is 70.**

**The marks for each question are shown in brackets [ ].**

**Quality of extended responses will be assessed in questions marked with an asterisk (\*).**

**BLANK PAGE**

- 1 A 2-dimensional (2D) array, `data`, holds numeric data that Karl has entered. The declaration for the array is:

```
array data[16,11]
```

The array `data`, has 16 'rows' and 11 'columns'.

Fig. 1.1 shows an extract from `data`.

Fig. 1.1

	0	1	2	3	...	10
0	1	5	7	12	...	36
1	3	4	15	16	...	48
2	0	0	1	3	...	10
3	12	16	18	23	...	100
...	...	...	...	...	...	...
15	6	10	15	25	...	96

The data in each 'row' is in ascending numerical order.

Karl needs to analyse the data.

**(a) Karl needs to find out if a number he enters appears in a given row of the array. He is going to use a search algorithm to do this.**

**(i) State the name of TWO different search algorithms that Karl could consider using.**

**1** \_\_\_\_\_

**2** \_\_\_\_\_

**[2]**

**(ii) Choose ONE search algorithm from those you gave in PART (i), and describe how this algorithm works.**

**Algorithm** \_\_\_\_\_

**Description** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**[5]**

- (b) Karl needs to output the median value of each 'row' of the array.

The median is found by having all the numbers in a row in ascending numerical order, and taking the middle value.

For example, in Fig. 1.2 below, the median element in the row is the third element, so the median value is 7.

Fig. 1.2

1	5	7	12	15
---	---	---	----	----

Write an algorithm to output the median value of each 'row' of the 2D array data.

---

---

---

---

---

---

---

---

[3]

- (c) Karl needs to find the mean average of each 'column' of the array. The mean is calculated by adding together the numbers in the column, and dividing by the quantity of numbers in the column.

For example, in Fig. 1.3 the first 'column' mean would be:  $(1+3+0+12)/4 = 4$

**Fig. 1.3**

1	5	7	12
3	4	15	16
0	0	1	3
12	16	18	23

**Write an algorithm to output the mean value of each 'column' in the array data.**

---

---

---

---

---

---

---

---

---

---

**[5]**

- 2 A group of A-level students are working together to program a computer game.**

**In the game, the player controls a character who moves through a virtual world. The game starts with a load-up screen. The player can select which area to move to on an on-screen map, and then they control the movements of their character using a keyboard to solve puzzles on the screen.**

- (a) Identify TWO inputs that the user could enter to control the character and describe each input's function.**

**Input 1 \_\_\_\_\_**

**Use \_\_\_\_\_**

**Input 2 \_\_\_\_\_**

**Use \_\_\_\_\_**

**[4]**

- (b) The game is to be created using sub-procedures. The following table identifies and describes one sub-procedure the students could use.**

**Complete the table below, identifying THREE additional sub-procedures that the students could create from the description at the start of question 2.**

**Describe the purpose of each sub-procedure you have identified. [6]**

	Sub-procedure	Purpose
e.g.	characterMovement	Takes the key the player pressed and moves the character in that direction
1		
2		
3		

- (c) The following pseudocode algorithm is for the sub-procedure characterMovement.

```
procedure characterMovement
(inputKey:byVal, characterx:byRef,
characterY:byRef)
    if inputKey == 37 then
        characterx = characterx + 1
    elseif inputKey == 38 then
        characterY = characterY + 1
    elseif inputKey == 39 then
        characterx = characterx - 1
    elseif inputKey == 40 then
        characterY = characterY - 1
    endif
endprocedure
```

- (i) Identify the THREE parameters in the procedure characterMovement.

- 1 \_\_\_\_\_
- 2 \_\_\_\_\_
- 3 \_\_\_\_\_

[3]

- (ii) Describe the decision that is made in this procedure and how the decision affects the flow through the procedure.

---

---

---

---

---

---

---

---

[3]

- (iii) Explain why `characterx` and `charactery` are passed `byRef` and not `byVal`.

---

---

---

---

---

---

---

---

[3]

[illegible]

---

---

---

---

---

---

---

---

---

---

**(e) Explain, using examples, how abstraction would be used to create the virtual world.**

---

---

---

---

---

---

---

---

---

---

**[4]**

- 3 A procedure takes as input a number between 1 and 100. It calculates and outputs the square of each number starting from 1, to the number input. The square of a number is the result of multiplying a number by itself.

```
procedure squares()  
  do  
    number = int(input("Enter a number  
    between 1 and 100"))  
    until number >= 1 AND number <= 100  
  
    for x = 1 to number  
      print(x * x)  
    next x  
  endprocedure
```

- (a) The procedure uses one programming construct twice.

State whether the construct that is used twice, is iteration or branching.

\_\_\_\_\_ [1]

- (b) State why the algorithm is a procedure and not a function.

\_\_\_\_\_  
\_\_\_\_\_ [1]

**(c) The procedure needs to be tested.**

**(i) Describe how black box testing can be used to test a program.**

---

---

---

---

---

---

[3]

**(ii) For each type of test given in the table, identify TWO examples of test data that can be used to test the program. [3]**

Test Type	Test Data 1	Test Data 2
Normal		
Extreme		
Invalid		

- 4 A program stores a queue of mathematical questions to be asked to a user. The questions are asked in the order they are added. Once a question has been asked it cannot be asked again. New questions are continually added to the end of the queue.

The program will use a non-circular queue, `questions`, (implemented using an array) to store the questions.

The pointer, `head`, stores the index of the first element in the queue.

The pointer, `tail`, stores the index of the last element in the queue.

- (a) Describe why a queue is a suitable structure for this program.

---

---

---

---

---

---

---

[3]

- (b) Fig. 4.1 shows an example of the data in the queue. `head` is currently 0, `tail` is currently 4.

Fig. 4.1

"2*3"	"1+4"	"3-1"	"10/2"	"3+6"			
-------	-------	-------	--------	-------	--	--	--

- (i) Show the contents of the queue shown in Fig. 4.1, after the following code is run.  
`add ("6+1")`

--	--	--	--	--	--	--	--

[2]

- (ii) State the values stored in `head` and `tail` after the code in PART (i) has run.

`head` \_\_\_\_\_

`tail` \_\_\_\_\_

[2]

- (c) Complete the following algorithm, to remove, and output, the first element in the queue.

`procedure remove()`

---

---

---

---

---

---

---

---

`endprocedure`

[4]

- (d) Complete the following algorithm, to ask the user to input a new question and then either add it to the queue, or report that the queue is full.

```
procedure add()  
    maxElements = 10
```

---

---

---

---

---

---

---

---

```
endprocedure
```

[4]

**END OF QUESTION PAPER**

## **Copyright Information**

**OCR is committed to seeking permission to reproduce all third-party content that it uses in its assessment materials. OCR has attempted to identify and contact all copyright holders whose work is used in this paper. To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced in the OCR Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download from our public website ([www.ocr.org.uk](http://www.ocr.org.uk)) after the live examination series.**

**If OCR has unwittingly failed to correctly acknowledge or clear any third-party content in this assessment material, OCR will be happy to correct its mistake at the earliest possible opportunity.**

**For queries or further information please contact the Copyright Team, First Floor, 9 Hills Road, Cambridge CB2 1GE.**

**OCR is part of the Cambridge Assessment Group; Cambridge Assessment is the brand name of University of Cambridge Local Examinations Syndicate (UCLES), which is itself a department of the University of Cambridge.**