Qualification Accredited

Oxford Cambridge and RSA

AS LEVEL

Examiners' report

COMPUTER SCIENCE

H046
For first teaching in 2015

H046/02 Summer 2019 series

Version 1

www.ocr.org.uk/computerscience

Contents

Introduction

Our examiners' reports are produced to offer constructive feedback on candidates' performance in the examinations. They provide useful guidance for future candidates. The reports will include a general commentary on candidates' performance, identify technical aspects examined in the questions and highlight good performance and where performance could be improved. The reports will also explain aspects which caused difficulty and why the difficulties arose, whether through a lack of knowledge, poor examination technique, or any other identifiable and explainable reason.

Where overall performance on a question/question part was considered good, with no particular areas to highlight, these questions have not been included in the report. A full copy of the question paper can be downloaded from OCR.

Paper 2 series overview

Candidates found the paper accessible and, in most cases, attempted all questions set. The paper differentiated candidates effectively.

The paper targets three specific areas: Knowledge and Understanding, Application and Evaluation. Questions that targeted Knowledge and Understanding required candidates to have studied the whole specification and to have learnt the relevant definitions. Some candidates had not been prepared by covering the whole specification and thus did not achieve marking points targeted at lower grades for basic recall for standard algorithms e.g. bubble sort, insertion sort.

Questions targeting Application required higher order skills to be able to use knowledge gained in context to solve problems – and often, a good understanding of programming and logic would have enabled candidates to access a number of marks. There was clear differentiation between candidates who understood the concepts and who could apply them, and those who displayed little ability to apply what they had learnt.

A significant number of candidates struggled to write pseudocode, but, overall, the standard of pseudocode presented was better than in previous sessions. Structured English is insufficient for examination questions that specifically require pseudocode to be written. Candidates are not required to write pseudocode to the standard presented in the specification, and minor variations in terms of influences from programming languages are taken account of.

Question 1 (a) (i)

1 Janet is designing a piece of software for a furniture company.

The software will allow a user to plan the position of furniture in a room. Users will be able to set the size and shape of a room, and then choose furniture from a library of furniture items. These pieces of furniture will have set sizes and designs and the user will be able to view the room in 3D to see how it looks from a variety of angles.

(a) Janet is using computational thinking techniques during the design process.

(i) Janet is removing some aspects during the design of the software to simplify it and to make it easier to produce.

State the name of the computational thinking technique that Janet is using.

.... **[1]**

The concept of abstraction was widely understood and the vast majority of candidates answered correctly.

Question 1 (a) (ii)

(ii) The computational thinking technique in **part (a)(i)** makes it easier to produce the software.

Identify **one** additional reason why this technique is necessary.

....

.... **[1]**

Many candidates answered the question successfully. Where candidates were not successful, they had often not read the question clearly. The most common erroneous responses either given a definition of abstraction instead of identifying why it was necessary or had explained why it would be easier to use the software produced rather than why it would have been easier to produce the software in the first place.

Question 1 (a) (iii)

(iii) Explain, with examples, **two** ways in which Janet will apply the computational thinking technique in **part (a)(i)** to this project.

1 ....

....

....

....

2 ....

....

....

....

[4]

Many candidates found it difficult to apply their knowledge of abstraction to the question context and gave responses that were not directly related to modelling a room with furniture in.

	AfL	Candidates should to be encouraged to read the stem of the question to consider how they could apply their knowledge to the context of the scenario set. Candidates need to practice applying computational thinking methods to a wide variety of problem types.

Question 1 (b) (i)

(b) Janet is planning the inputs and outputs for the software.

(i) Identify **two** inputs that the software will need to take.

1 ....

2 ....

[2]

Candidates generally answered the question successfully. A small number of candidates identified input devices rather than inputs into the software that would be required for the model to run.

Question 1 (b) (ii)

(ii) Identify **two** outputs that the software will need to produce.

1 ....

2 ....
[2]

Candidates generally answered the question successfully. A small number of candidates identified output devices rather than outputs that the program would produce.

Question 1 (c)

(c) Janet is going to decompose the problem to produce a set of subprograms.

Explain the benefits of using subprograms to produce this software.

....

....

....

....

....

....

....

.... **[4]**

Some candidates spent time defining decomposition which was not required. Many candidates scored some credit for their response, but there was often repetition in many responses. Fewer candidates could identify three or more distinct points.

Question 1 (d)

(d) The program allows the user to enter dimensions of the room and the furniture. There are preconditions that must be met before the software will draw the room and furniture.

Suggest **two** preconditions that must be met before the software will run.

1 ....

....

2 ....

....

[2]

Many candidates gave responses that were quite vague that could not be evaluated e.g. 'Room dimensions'. Stronger candidates phrased their responses with more precision plus a condition that could be evaluated e.g. 'All room measurements entered must be greater than zero'.

?	**Misconception**	Many candidates did not know that a precondition is something that can be evaluated to give either a True or False outcome.

Question 1 (e)*

(e)* Janet is planning the testing strategy for the software.

Discuss the different testing methods available to Janet, and make a recommendation for which ones she should use.

....

....

....

....

....

.... **[9]**

Most candidates could identify some form of testing with a description. Weaker candidates struggled to define two or more categories of testing with accurate definitions. Often descriptions of testing were generic, but some candidates did give good responses that were relevant within the context of testing a 3D room-modelling system. Few candidates exhibited higher order levels of understanding of the relevant appropriateness of the different testing methods to different stages of the project development, and fewer still could compare the appropriateness of the different methods in an analytical way. Candidates continue to find the response structure and the level of logical reasoning required in the banded response question challenging.

	AfL	Many candidates would benefit from having a clear and logical structure to their responses. Candidates must show knowledge, Application and Evaluation. First, the initial paragraph(s) needs to exhibit the relevant knowledge (definitions). Secondly, the next paragraph(s) should give responses that are relevant to the given scenario. Finally, the last paragraph(s) should make an evaluation that compares the different methods in terms of relative advantages / disadvantages or impacts / consequences.

Question 2 (a) (i)

2 A procedure is shown in the following pseudocode.

The arrays that are passed to the procedure store integer values.

`length` returns the total number of elements the array can hold.

```
01 procedure calculateOnce(data[]:byRef, nextData[]:byRef)
02      if data.length > nextData.length then
03                loopCount = nextData.length - 1
04      else
05                loopCount = Data.length - 1
06      endif
07      count = 0
08      while count <= loopCount
09                data[count] = data[count] + nextData[count]
10                count = count + 1
11      endwhile
12 endprocedure
```

(a) A decision is made on line `02`.

(i) Identify the line where the second decision is made.

.... **[1]**

Many candidates confused the else clause of the first *if* statement on line 04 with the second decision that was based on the evaluation of a condition in line `08`.

Question 2 (a) (ii)

(ii) Explain the purpose of the code in lines `02` to `06`.

....

....

....

....

....

.... **[3]**

Candidates often described the lines of code literally rather than demonstrating a real understanding of the underlying purpose of the code. Stronger candidates demonstrated that they understood that the *data* and *nextData* were arrays whose size was being compared to ascertain which was the smaller sized array. Weaker candidates did not explain the underlying context that was required.

Question 2 (b) (i)

(b) The procedure has parameters passed by reference.

(i) Give the identifiers of the **two** parameters.

1 ....

2 ....
[2]

Many candidates correctly identified the identifiers of the parameters as `data` and `nextData` but a significant number gave the incorrect responses `data[]` and `nextData[]`.

	AfL	Candidates need to know how valid identifier names are constructed. Candidates should know that identifier names cannot include spaces or brackets () [].

Question 2 (b) (ii)

(ii) State the effect of the array `data[]` being passed by reference and not by value.

....

.... **[1]**

Passing parameters by value and passing by reference continue to cause confusion among many candidates. Those candidates that knew the difference sometimes gave definitions rather than identifying the actual effect.

Question 2 (c) (i)

(c) The program needs a second procedure, `sortData`. It will be called taking the array `data[]` as a parameter by reference.

The procedure will then perform a bubble sort on the data in the array.

(i) Show each stage of a bubble sort on the following contents of `data[]`:

95	10	5	33	100	77	45

....

....

....

....

....

.... **[4]**

A high proportion of candidates knew the basic principle of a Bubblesort and how it operated, with most candidates obtaining at least the first marking point. Some candidates gave textual descriptions that made it difficult to follow the logic of the response. Those candidates that presented each step in the algorithm, clearly highlighting each swap that took place, demonstrated the clearest understanding.

Exemplar 1

10 95→5 33 100 77 45
10 5 95→33 100 77 45
10 5 33 95 100→77 45
10 5 33 95 77 100→45
10→5 33 95 77 45 100
5 10 33 95→77 45 100
5 10 33 77 95→45 100
5 10 33 77→45 95 100
5 10 33 45 77 95 100

Exemplar 1 illustrates a candidate response that shows each of the swaps in the Bubblesort clearly identified.

Question 2 (c) (ii)

(ii) Write, using pseudocode, the procedure `sortData`.

....

....

....

....

....

.... **[8]**

Most candidates wrote pseudocode, but a few candidates wrote in a vague structured English or in a textual description that was not creditworthy. Bubble sort is one of the few algorithms that candidates must be able to program and to recall, so it was, perhaps, surprising that many candidates were not able to achieve more than 5 marks. Common mistakes included loops that would, when executed, have resulted in array out bounds errors. The strongest candidates wrote an efficient implementation of a bubble sort that stopped executing when no more swaps had occurred during a pass.

Question 2 (c) (iii)

(iii) An alternative sorting method is the insertion sort.

Show how an insertion sort will sort the data in the following array.

95	10	5	33	100	77	45

....

....

....

....

....

.... **[4]**

Those candidates who understood the principles behind insertion sort often clearly delineated their responses into the 'sorted' and the 'unsorted' parts of the array. This made it easier to follow the steps that had been applied. Some candidates wrote long-winded textual descriptions of the principles of the process without actually showing how the principles applied to the actual data, which meant that credit could not be given.

Exemplar 2

sorted	unsorted
95	10, 5, 33, 100, 77, 45
10, 95	5, 33, 100, 77, 45
5, 10, 95	33, 100, 77, 45
5, 10, 33, 95	100, 77, 45
5, 10, 33, 95, 100	77, 45
5, 10, 33, 45, 95, 100	77
5, 10, 33, 45, 77, 95, 100	

Exemplar 2 illustrates a candidate response that shows each of the steps of the insertion sort in a clear diagrammatic format.

Question 3 (a)

3 The current contents of a queue, `colours`, implemented in an array is shown in Fig. 3.1.

red	yellow	green	blue	grey			

```
front = 0

end = 4
```

Fig. 3.1

(a) Describe the purpose of `front` and `end`.

....

....

....

.... **[2]**

Few candidates understood that `front` and `end` were used as pointers into the underlying array data structure.

Question 3 (b) (i)

(b) The queue has the subprograms `enqueue` and `dequeue`. The subprogram `enqueue` is used to add items to the queue and the subprogram `dequeue` removes items from the queue.

(i) Use the following diagram to show the queue shown in Fig. 3.1 after the following program statements have run:

```
enqueue("orange")
dequeue()
enqueue("maroon")
dequeue()
dequeue()
```


front = ...

end = ...

[4]

Candidates were given credit for responses that interpreted the queue as one that was shifted forward when an item was dequeued() or one that was implemented as a circular queue. Many candidates displayed a clear understanding of the use of the enqueue() dequeue() operations.

Question 3 (b) (ii)

(ii) `enqueue` and `dequeue` are both functions.

State the difference between a procedure and a function.

....

.... **[1]**

Most candidates could give a clear definition and demonstrated that they had leant a key term. Few candidates confused procedures with functions.

Question 3 (b) (iii)

(iii) Describe the steps involved in the `enqueue` algorithm.

....

....

....

....

....

....

....

.... **[4]**

Many candidates either described checking for the queue full state or described the process for adding an item to the queue. Fewer gave both parts that were required for a full response.

Question 4 (a)

4 A program corrects the grammar in a line of text. The text is read in from a text file.

(a) The function, `getText`, needs to:

- take the file name as a parameter
- open the file
- read the line of data in the text file into one string
- return the string of data.

Write the function `getText`.

....

....

....

....

....

....

....

....

....

.... **[4]**

Many candidates had some knowledge of what was required, but fewer could write recognisable pseudocode with precision. Opening and reading from files required a variable for the file to be initialised, but many candidates executed a file open statement without assigning the file to an identifier that could then later be used. Only the strongest candidates subsequently closed the file before returning the input that had been read – a number of candidates erroneously returned the value before the file had been closed.

Question 4 (b)

(b) The procedure, `fullStop`, needs to:

- ask for a file name as input
- read the data from the file using the function `getText`
- replace the first letter after each full stop with a capital letter if it is currently lower case (if the next character is a space, it must check each successive character until it finds a letter)
- write the edited data back to the text file.

You can assume the text file only contains upper and lower case letters, spaces and full stops.

Part of the ASCII table has been provided:

ASCII Value	Character
65	"A"
90	"Z"
97	"a"
122	"z"
32	" " (space)
46	"." (full stop)

The following functions may be used in your answer:

`asc(character)` returns the ASCII value for a single character, e.g. `asc("A")` would return 65.

`upper(character)` returns the single character in upper case, e.g. `upper("a")` would return "A".

Write the procedure `fullStop`. **[7]**

....

....

....

Most candidates achieved 2 of the first 3 marking points for initialising the procedure and reading the text file. Few scored nothing. Many candidates looped through the input text with a loop and matched where a full-stop was located. Fewer then went on to accurately move forward to the first letter after the full-stop was found, and fewer still then successfully converted the next letter to upper case.

Supporting you

For further details of this qualification please visit the subject webpage.

Review of results

If any of your students' results are not as expected, you may wish to consider one of our review of results services. For full information about the options available visit the OCR website. If university places are at stake you may wish to consider priority service 2 reviews of marking which have an earlier deadline to ensure your reviews are processed in time for university applications.

activeresults

Review students' exam performance with our free online results analysis tool. Available for GCSE, A Level and Cambridge Nationals.

It allows you to:

- review and run analysis reports on exam performance
- analyse results at question and/or topic level*
- compare your centre with OCR national averages
- identify trends across the centre
- facilitate effective planning and delivery of courses
- identify areas of the curriculum where students excel or struggle
- help pinpoint strengths and weaknesses of students and teaching departments.

*To find out which reports are available for a specific subject, please visit ocr.org.uk/administration/support-and-tools/active-results/

Find out more at ocr.org.uk/activeresults

CPD Training

Attend one of our popular CPD courses to hear exam feedback directly from a senior assessor or drop in to an online Q&A session.

Please find details for all our courses on the relevant subject page on our website.

www.ocr.org.uk

The small print

OCR Resources: *the small print*

OCR's resources are provided to support the delivery of OCR qualifications, but in no way constitute an endorsed teaching method that is required by OCR. Whilst every effort is made to ensure the accuracy of the content, OCR cannot be held responsible for any errors or omissions within these resources. We update our resources on a regular basis, so please check the OCR website to ensure you have the most up to date version.

This resource may be freely copied and distributed, as long as the OCR logo and this small print remain intact and OCR is acknowledged as the originator of this work.

Our documents are updated over time. Whilst every effort is made to check all documents, there may be contradictions between published support and the specification, therefore please use the information on the latest specification at all times. Where changes are made to specifications these will be indicated within the document, there will be a new version number indicated, and a summary of the changes. If you do notice a discrepancy between the specification and a resource please contact us at: resources.feedback@ocr.org.uk.

Whether you already offer OCR qualifications, are new to OCR, or are considering switching from your current provider/awarding organisation, you can request more information by completing the Expression of Interest form which can be found here: www.ocr.org.uk/expression-of-interest

Please get in touch if you want to discuss the accessibility of resources we offer to support delivery of our qualifications: resources.feedback@ocr.org.uk

Looking for a resource?

There is now a quick and easy search tool to help find **free** resources for your qualification:

www.ocr.org.uk/i-want-to/find-resources/

www.ocr.org.uk

OCR Customer Support Centre

General qualifications

Telephone 01223 553998

Facsimile 01223 552627

Email general.qualifications@ocr.org.uk

OCR is part of Cambridge Assessment, a department of the University of Cambridge. *For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored.*

Cambridge Assessment

CERTIFIED ISO 9001

UKAS MANAGEMENT SYSTEMS 001