

A LEVEL

Examiners' report

COMPUTER SCIENCE

H446

For first teaching in 2015

H446/02 Summer 2023 series

Contents

Examiners' report template..... i

Contents.....2

Introduction4

Paper 2 series overview5

Section A overview.....6

 Question 1 (a) (i).....6

 Question 1 (a) (ii)7

 Question 1 (a) (iii)7

 Question 1 (a) (iv)8

 Question 1 (a) (v)8

 Question 1 (a) (vi)9

 Question 1 (b)9

 Question 2*11

 Question 3 (a)12

 Question 3 (b)13

 Question 3 (c)13

 Question 3 (d)14

 Question 3 (e)15

 Question 4 (a) (i)16

 Question 4 (a) (ii)16

 Question 4 (a) (iii)17

 Question 4 (b)17

 Question 4 (c)18

 Question 5 (a)19

 Question 5 (b)20

 Question 621

 Question 7* (a).....23

 Question 7* (b) (i).....26

 Question 7* (b) (ii).....26

 Question 7* (c) (i).....27

 Question 7* (c) (ii)28

 Question 7* (d).....29

 Question 8 (a)30

 Question 8 (b)30

Section B overview31

 Question 9 (a) (i)31

 Question 9 (a) (ii)32

 Question 9 (b) (i)33

 Question 9 (b) (ii)34

 Question 9 (b) (iii)34

 Question 9 (c)35

 Question 9 (d)36

 Question 9 (e)37

 Question 9 (f)*38

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. A selection of candidate answers is also provided. 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 and the mark scheme can be downloaded from OCR.

Would you prefer a Word version?

Did you know that you can save this PDF as a Word file using Acrobat Professional?

Simply click on **File > Export to** and select **Microsoft Word**

(If you have opened this PDF in your browser you will need to save it first. Simply right click anywhere on the page and select **Save as . . .** to save the PDF. Then open the PDF in Acrobat Professional.)

If you do not have access to Acrobat Professional there are a number of **free** applications available that will also convert PDF to Word (search for PDF to Word converter).

Paper 2 series overview

Paper 2 is about algorithms and problem solving. It tests candidates' computational thinking ability to analyse and solve problems. Candidates are expected to be able to write algorithms fluently in either pseudocode or program code and to be able to trace algorithms. This means that candidates require a solid grounding in standard data structures backed by practical experience. This paper also covers Object Oriented Programming principles and candidates need to have extensive experience of Object Oriented Programming (OOP) to be able to tackle section B of the paper successfully.

Candidates who did well on this paper generally:	Candidates who did less well on this paper generally:
<ul style="list-style-type: none"> • displayed an ability to competently interpret and write pseudocode • demonstrated a clear understanding of OOP principles allied to an ability to write code that defined classes and used methods provided for a given scenario • displayed a good knowledge of a range of data structures and standard algorithms for operations on the given data structures, the data structures covered were linked lists, trees and graphs • where levels of response questions were set it was noticeable that stronger candidates tried to make more AO3 evaluative comparisons that were relevant to the scenarios given. 	<ul style="list-style-type: none"> • displayed little ability to interpret code or to be able to write basic algorithms using pseudocode • showed a very limited understanding of OOP that exposed little practical experience of relevant programming • displayed limited knowledge of standard data structures such as lists and trees, but had some AO1 knowledge, but struggled more with AO2 application • demonstrated very limited knowledge in levels of response questions and struggled to apply their learning to the scenarios given.

Section A overview

Section A of the paper consisted of 8 questions, each with a separate theme.

Question	Topics covered
1	Tree and graph data structures, including Dijkstra's algorithm
2	Computational thinking including decomposition and problem recognition
3	List data structures
4	Interpretation of procedural code
5	Recursion and tracing recursive functions
6	Writing pseudocode algorithms
7	Big O notation and algorithmic complexity Concurrency Merge sorting IDE features
8	Computation thinking – thinking procedurally

Question 1 (a) (i)

1 A tree is one example of a data structure.

(a) (i) Give **two** characteristics of a tree data structure.

1

.....

2

.....

[2]

This question was generally well answered by most candidates with the most common answers including different types of nodes such as root, child and leaf nodes.

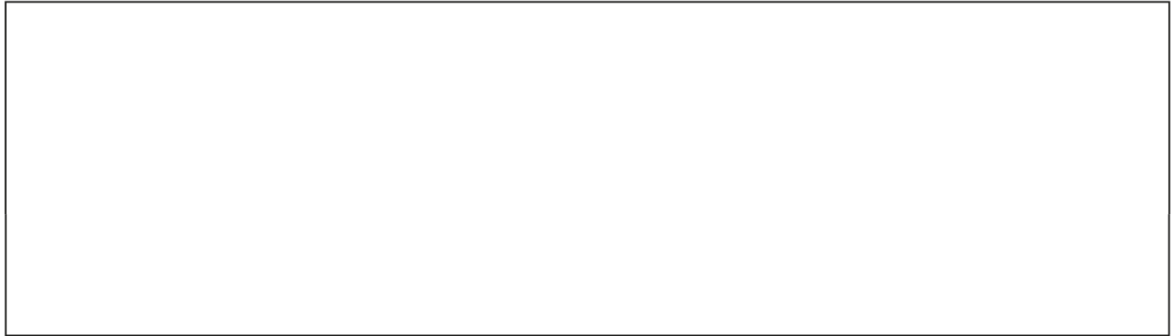
Some candidates used vague terminology or used terminology more commonly associated with graphs such as vertices instead of nodes, or edges instead of branches. Imprecise language such as undirected/non-directed was used whereas it would have been clearer to have indicated that trees are hierarchical structures that are rooted. Some candidates erroneously confused binary trees with general trees stating that nodes could only have a maximum of two child nodes, as the tree in this question was not specifically limited to the special case of a binary tree.

Question 1 (a) (ii)

(ii) The following data is entered into a binary search tree.

22 13 5 36 55 14 8

Draw the binary search tree when the given data is entered in the order given.



[4]

Most candidates gained some marks, with many gaining full marks. Occasionally the left/right ordering of child nodes was unclear, so marks could not be given in such instances.

Question 1 (a) (iii)

(iii) Describe how a **leaf node** is deleted from a binary search tree.

.....

.....

.....

..... [2]

This question was generally less successfully answered by many candidates. There was considerable vagueness in some responses such as 'finding the end node and removing it'. This left questions such as 'Which end node?', 'Removed how?' for the examiner to infer, so were too vague to gain marks.

When candidates identified the need to locate the leaf node, most omitted the need for the parent node pointer to be set to null to break the link. Very few candidates indicated that a deleted node would then be added to a free list or that the memory freed could be retrieved via garbage collection.

Question 1 (a) (iv)

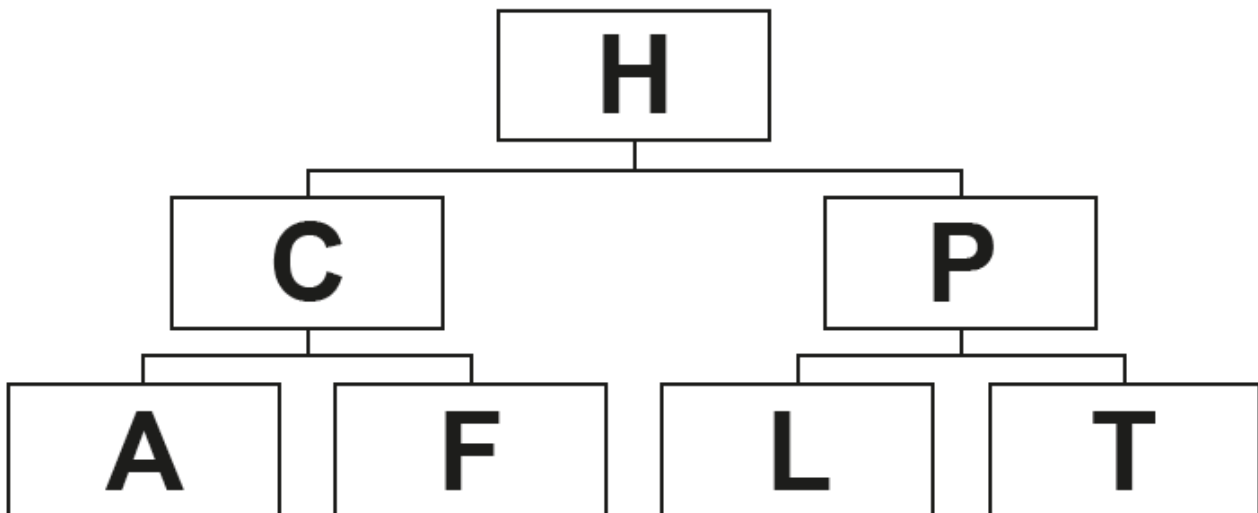
(iv) Describe how a binary search tree can be searched for a value.

.....
.....
.....
.....
.....
..... [4]

This question was generally well answered by many candidates who appreciated that a Binary Search Tree (BST) could be efficiently searched because it is ordered, rather than traversed.
Weaker candidate often confused the search of a BST structure with traversal algorithms such as breadth first or depth first traversals.

Question 1 (a) (v)

(v) Identify the order that the nodes will be visited in a **depth-first (post-order)** traversal of this binary search tree.



..... [4]

Some candidates confused Depth First Search with other traversal algorithms, most commonly Breadth First Search, but this question was generally answered well by most candidates. Another common mistake was to output the nodes in the order in which they were first encountered rather than the order in which they would be visited/output.

Question 1 (a) (vi)

(vi) Explain how backtracking is used in depth-first (post-order) traversals.

.....
.....
.....
..... [2]

Some candidates described the term 'backtracking' in general, so did not answer the question, which required a response in the context of a depth first tree traversal. Many candidates thought that backtracking always began at the leftmost node, which is not true, it starts when a leaf node is reached. However, a pleasing number of candidates did identify the first example of backtracking in the given tree, from leaf A to parent node C.

Question 1 (b)

(b) A graph is another type of data structure.

An example graph is shown in Fig. 1.

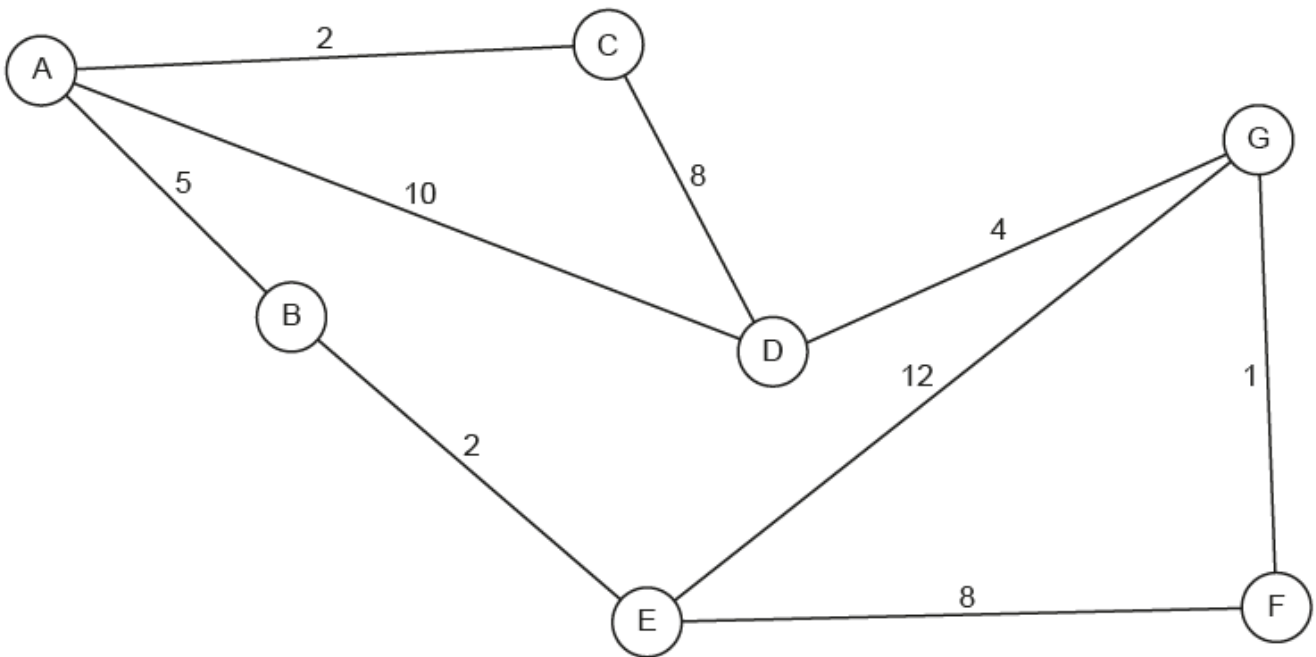


Fig. 1

.....
.....
.....
.....
.....

Node	Distance travelled	Previous node

Final path:

Distance:

[6]

Candidates answering 'by inspection' could gain 2 marks for the final path and distance, and many less successful responses thus scored 1 or 2 marks by doing so, demonstrating little real understanding of how Dijkstra's algorithm operates. A common error was the incorrect solution ACDG that showed ignorance of not continuing the search from the node with the least distance travelled that has not already been marked as visited. For full marks there had to be an indication that the first path to G (19 from E) was overwritten by G (14 from D) for the last mark in the table. Relatively few candidates demonstrated this.

Question 2*

- 2* A company needs a new computer program that will create schedules for delivery drivers. It will need to identify a possible order that the drivers can deliver items and possible routes they could take.

Discuss how programmers could make use of problem recognition and problem decomposition when designing this system.

You should include the following in your answer:

- a description of both problem recognition and decomposition
- how each method can be used when designing the solution
- the benefits of using each method when designing the solution.

[9]

Many Level 1 responses gave simplistic definitions of problem recognition and/or decomposition, often with more successful descriptions for decomposition. Relatively few candidates could clearly state that problem recognition starts with identifying inputs and outputs. Some degree of application to the question scenario context was required for Level 2. Most candidates' marks clustered around Level 2 with 4 or 5 marks. Many generic answers focused on problem recognition and/or decomposition as learnt terms, with associated benefits given, but little more than that, which limited marks to Level 2. For Level 3 an evaluation of ideas relevant to the context of the scenario given was expected.

Some good responses indicated recognition of a pathfinding type problem with a need for shortest path/A* algorithm, approximation to travelling salesman type solutions or the need to employ heuristics. Candidates giving this level of insight often accessed Level 3.

Question 3 (a)

3 A program stores data in a linked list.

The current contents of the linked list are shown in Fig. 3, along with the linked list pointers.

headPointer	1
freeListPointer	4

location	data	pointer
0	"blue"	6
1	"red"	0
2	"green"	8
3	"orange"	NULL
4		5
5		7
6	"grey"	2
7		9
8	"purple"	3
9		NULL

Fig. 3

(a) State the purpose of headPointer and freeListPointer in the linked list shown in Fig. 3.

headPointer

.....

freeListPointer

.....

[2]

While many candidates did gain full marks, a noticeable number of candidates struggled with the use of technical language in this question. Where a candidate's intention or meaning was clear, for example, 'headPointer is the first item' or 'freeListPointer is the first free space', marks were given. Some candidates did not show an understanding that these pointers represented the start of two separate linked lists within the static array structure.

Question 3 (b)

(b) State the meaning of the pointers with the value `NULL` in the linked list shown in **Fig. 3**.

.....
..... [1]

This question was generally well answered, with many candidates gaining marks. A generic 'no assigned value' was not given marks, as the response had to be answered in the context of the linked list, so end of list/not pointing to another node was required, rather than stating pointing to nothing.

Question 3 (c)

(c) A procedure outputs the data in the linked list shown in **Fig. 3** from the first item in the list, to the last item.

Give the output from the procedure.

.....
..... [2]

Again, this question was generally well answered by most candidates. The most common erroneous answer was 'Blue' as it was the first item in Figure 3 given at index 0. This was invariably given as a response when candidates did not know what the function of the `headPointer` was.

Question 3 (d)

(d) A new item needs to be added to the linked list.

Describe how a new item is added to a linked list.

.....

.....

.....

.....

.....

.....

..... [4]

More successful responses gave good clear examples to illustrate a potential sequence of operations to describe the process, and many therefore achieved 3 or 4 marks. For example 'new item added to location 4 indicated by `FreeListPointer` and its pointer set to `Null`; Existing list searched from `headPointer` to its end at *Orange* at index 3, and its pointer updated to the new last item in the list at index 4.'

Some candidates lacked appreciation that this was a static record structure in this scenario, so locations from the free list had to be used. Many candidates were unclear as to how the end of the current linked list was found, and just gave answers from that point onward without saying how it was found by traversing to the end of the existing list. There were relatively few prepend type solutions, but they were accepted as valid where seen.

Question 3 (e)

- (e) The function `findNode` will search the linked list and return either the position of the node that contains the data item, or -1 if the data item is not found.

The data held in a node at location `x` can be accessed with `linkedList[x].data`. The pointer of the node at location `x` can be accessed with `linkedList[x].pointer`.

For example, using the linked list shown in **Fig. 3**:

`linkedList[2].data` returns green.

`linkedList[2].pointer` returns 8.

Complete the function, using pseudocode or program code.

```
function findNode(toFind, headPointer, linkedList)
    currentNode = .....
    while(currentNode != ..... )
        if linkedList[currentNode]. ..... == toFind then
            return currentNode
        else
            currentNode = linkedList[.....].pointer
        endif
    endwhile
    return .....
endfunction
```

[5]

This question required exact answers only. Many candidates gained some marks, and there was a good distribution of marks. More successful programmers tended to get most of the marks available.

Question 4 (a) (i)

4 A programmer has designed a program that includes a reusable program component.

- (a) The reusable program component is a function called `isInteger()`. This will take a string as an argument and then check that each digit is between 0 and 9. For example if 103 is input, it will check that the digits 1, 0 and 3 are each between 0 and 9.

The `asc()` function returns the ASCII value of each digit. For example `asc("1")` returns 49.

The ASCII value for 0 is 48. The ASCII value for 9 is 57.

```

01  function isInteger(number)
02      result = true
03      for count = 0 to number.length-1
04          asciiValue = asc(number.substring(count, 1))
05          if not(asciiValue >= 48 and asciiValue <= 57) then
06              result = false
07          endif
08      next count
09      return result
10  endfunction
    
```

- (i) Identify **one** identifier used in the function `isInteger()`.

..... [1]

This question was generally well done (although slightly less well done than parts (a) (ii) and (a) (iii)), but many candidates did very well. The most common erroneous responses were giving the names of predefined functions/properties or giving relational operators.

Question 4 (a) (ii)

- (ii) Give the line number where the branching (selection) construct starts in the function `isInteger()`.

..... [1]

This question required an exact answer only and was answered correctly by the majority of candidates.

Question 4 (a) (iii)

(iii) Give the line number where the iteration construct starts in the function `isInteger()`.

..... [1]

This question also required an exact answer only and was answered correctly by the majority of candidates.

Question 4 (b)

(b) Describe the purpose of the following lines in the function `isInteger()`.

Line 03

.....

Line 04

.....

Line 09

.....

[3]

This question required the *purpose* of the lines of code to be described, but many candidates just described the functionality of the lines of code such as 'line 03 is a counter controlled loop from 0 to `number.length - 1`'. The expected purpose of this line of code was to set up a loop to iterate through each character in the input string parameter.

While many candidates could describe the purpose of at least one of the lines of code given, few could clearly describe the purpose of all three lines.

Question 4 (c)

(c) Give **two** reasons why reusable program components are used in programs.

1

.....

2

.....

[2]

Many less successful responses gave vague generalities such as 'saves time' or 'more efficient' without specifying why or how. Points given must be qualified in some way at A Level. For example, 'saves development time as pre-written routines are available'.

Pre-written or pre-tested, and saving development time due to already being written, were the most popular answers.

Question 5 (a)

5 A recursive pseudocode function, recursiveAlgorithm(), is shown.

```
01 function recursiveAlgorithm(value)
02     if value <= 0 then
03         return 1
04     elseif value MOD 2 = 0 then
05         return value + recursiveAlgorithm(value - 3)
06     else
07         return value + recursiveAlgorithm(value - 1)
08     endif
09 endfunction
```

(a) Describe the key features of a recursive algorithm.

You may refer to the function, recursiveAlgorithm() in your answer.

.....

.....

.....

.....

.....

.....

..... [3]

Weaker responses were limited to one or two points for 'calls itself' with example such as 'line 05'. Fewer candidates went on to identify the requirement for a base case. A variety of terminology was used for the base case such as terminating case and stopping case. If candidates were clearly referring to the case where the recursive calls stopped, and the recursion started to unwind, the mark was given.

Question 5 (b)

(b) Trace the recursive function, `recursiveAlgorithm()`, and give the final return value when called with `recursiveAlgorithm(10)`. You may choose to use the table below to give your answer.

.....

.....

.....

.....

.....

.....

Function call	value	return

Final return value

[5]

While many candidates continue to find recursion a challenging topic there were many who encouragingly achieved full marks. Weaker candidates traced the initial sequence of calls but found it harder to identify the last call to `recursiveAlgorithm(-1)` that triggered the base case and then found it harder again to calculate the unwind sequence.

Question 6

6 Octal is a base 8 number system.

To convert a denary number to base 8:

- the denary value is divided by 8 and the remainder is stored
- the integer value after division is divided by 8 repeatedly until 0 is reached
- the remainders are then displayed in reverse order.

Example 1:

Denary 38

$38 / 8 = 4$ remainder 6	6
$4 / 8 = 0$ remainder 4	4

Octal = 46

Example 2:

Denary 57

$57 / 8 = 7$ remainder 1	1
$7 / 8 = 0$ remainder 7	7

Octal = 71

Write an algorithm to:

- take a denary value as input from the user
- convert the number to octal
- output the octal value.

You do **not** need to validate the input from the user.

Write your algorithm using pseudocode or program code.

.....

.....

.....

.....

.....

.....

.....

.....

.....

[6]

In general, there was a very poor standard of pseudocode algorithms from less successful responses. This demonstrated poor programming and limited problem-solving ability.

While there was no requirement to define a function the strongest candidates often did so, but then some forgot the requirement specified in the question for user input, rather than just presenting a parameterised function in isolation.

There were several common errors that were observed. A few candidates performed the octal conversion for just two digits rather than generalising the solution for a denary number of any length. Many candidates simply used '/' for division without specifying integer division through //, DIV or floor functions. Many candidates did not assign the result of a calculation to a variable for later use, presenting lines of code such as 'denaryNum MOD 8' which was not given marks. The order required for appending the remainder was frequently incorrect in many of the solutions that were presented, although some candidates did reverse the string at the end of the process if they did 'outString += remainder' style solutions, which was acceptable.

Most candidates achieved at least 1 mark for taking user input, and then the majority also used either modulus to calculate the remainder or integer division to calculate the value for the next iteration. Only the strongest candidates scored 5 or 6 marks.

Exemplar 1

```

Value = input ("enter a number:")
Output = ""
while Value != 0
    Output = str str.concat(Value MOD 8) + Output
    Value = Value DIV 8
end while
if Output == "" then
    Output = 0
endif
Print (Output)

```

This candidate response is not language specific but the logic and the operations used are clear. It shows clear logical structure with appropriate indentation of logical blocks. It was given full marks.

Question 7* (a)

- 7* (a) A program designer needs to decide on an algorithm to use from a choice of three. The table shows the worst-case Big O complexities for each algorithm.

Algorithm	Time Complexity	Space Complexity
1	Linear	Exponential
2	Exponential	Constant
3	Logarithmic	Logarithmic

The program will be used to analyse data that can range from 2 items to 2 billion items.

Compare the use of all **three** algorithms and suggest which the programmer should use.

You should include the following in your answer:

- the meaning of constant, logarithmic, linear and exponential complexity
- how well each algorithm scales as the amount of data increases
- which algorithm is the most suitable for the given task.

[9]

Several candidates confused logarithmic and exponential growth rates, and a significant number of candidates thought that exponential complexity was the polynomial function $O(n^2)$ rather than $O(2^n)$. Many candidates also defined terms such as exponential complexity as 'where it grows exponentially', such circular definitions were not given marks.

Level 1 responses identified some characteristics of Big O for constant, logarithmic, linear and exponential complexity with some occasional errors or inconsistencies.

Level 2 responses gave reasonable descriptions of both time and space complexity alongside accurate definitions for the complexity terms with an identification of algorithm 3 as being the best overall choice.

Level 3 responses had far more evaluative AO3 analysis but were few and far between. In this scenario candidates identified that it depended on the value of n , as if n was very small, exponential time growth was not an issue, so algorithm 2 might be preferential, but as n could grow to 2 billion this would be intractable, so algorithm 3 was chosen as the most practical. The data set in question had 2 billion items at most so $\log_2 2,000,000,000 = 30$ meant space overheads in algorithm 3 were manageable with modern storage capacities.

Exemplar 2

Order is used to evaluate the complexity of an algorithm, in respect to how its performance changes with a change to the size of input. This may be measuring how much longer it takes to complete, or how much additional load is placed on components such as memory.

Linear complexity means that the time taken is proportional to the size of the input, and will grow at a steady rate as input size increases. ^{space}

Constant complexity means that no matter the size of input, the time/load will always be the same regardless of the data being processed.

Exponential complexity grows very quickly as input increases, whilst logarithmic increases, though at a very slow rate.

As such, Algorithm 3 is definitely ideal, since the data may grow so large, to 2 billion items.

Algorithm 1 scales ~~poorly~~ poorly - though linear time is acceptable for small inputs, it quickly becomes unreasonable ~~for~~ with larger data sets, and the exponential space complexity will mean the data quickly requires much more processing power for just a small increase in input volume.

Though Algorithm 2's constant space complexity is ideal, the exponential time complexity means many of the larger possible data inputs will take exceptionally long to complete, and so is impractical and shouldn't be used.

Since algorithm 3 has both Time and space complexity as logarithmic, it will scale very well, as input volume can increase greatly with only a small decline to performance which shouldn't be noticeable, and is therefore most suitable to the task.

This response clearly demonstrates that a high scoring Level 3 response is possible within the space provided. The second page of the response shows clear AO3 evaluation between the merits of the three different algorithms and the scenario.

Question 7* (b) (i)

(b) The program designer is investigating the use of concurrent processing.

(i) Describe what is meant by the term 'concurrent processing'.

.....

.....

.....

..... [2]

Several candidates confused parallel processing (executing more than one instruction simultaneously) with concurrent processing (where most than one process/task is running at the same time). A lack of technical vocabulary was observed with many candidates giving responses such as 'many things processed at the same time', without specifying what the 'things' were. Occasionally candidates erroneously thought that processor pipelining was an example of concurrent processing.

While many candidates successfully identified that multiple processes are run simultaneously, fewer went on give the mechanism by which this could be achieved. Where they did so, these included time-slicing switching between different processes on a single processor, or use of multiple cores/parallelism to simultaneously execute different processes.

Question 7* (b) (ii)

(ii) Give **two** benefits of using concurrent processing.

1

.....

2

.....

[2]

Candidates found it difficult to give well-qualified responses to this question. Many candidates gave the definition for concurrency running multiple tasks at the same time as a benefit, which was not mark worthy.

Quicker processing/improved performance was not enough on its own without specifying that this was within a given time unit. Concurrent processing does not increase the actual speed of the CPU. Efficiency as a benefit on its own was insufficient, whereas 'less CPU idle time' was a well-qualified example of a benefit.

Question 7* (c) (i)

(c) The programmer needs to use a merge sort in one part of the problem to sort items in ascending order.

(i) Describe how a merge sort works.

.....

.....

.....

.....


.....

.....

..... [5]

Many candidates were not precise in describing how the initial data set was repeatedly halved and did not explain how the data set was actually broken down into individual items. Very few candidates could accurately describe how the merge phase of a merge sort combines two separate ordered lists together, and frequently this was omitted altogether. A number of candidates continued to demonstrate the misconception that data is sorted within sub-lists, rather than by the actual merging of two separate ordered lists together.

Misconception

 Many candidates continue to think that a merge sort performs sorting of data within lists. Merge sort performs the sorting part of the operation when it merges together two separate sub-lists that are already in a sorted order. This is done with the use of a pointer to each of the two sub-lists that acts as a placeholder. The two positions in the separate lists are compared, and the smaller item is added to the new sorted list and the pointer is incremented. The merging process then repeats until the sub-lists have been completely merged.

Question 7* (c) (ii)

- (ii) Give **one** benefit and **one** drawback of the programmer using a merge sort instead of a bubble sort.

Benefit

.....

Drawback

.....

[2]

Most candidates recognised that merge sort is typically quicker than bubble sort, but there were some unqualified answers. Clear responses were phrased in terms of Big O complexities or the numbers of data items to be sorted. Some candidates demonstrated greater understanding by commenting on the best/average/worst case Big O complexities. The most common disadvantages for merge sort given were additional memory overheads and the increased complexity of implementation.

A few candidates erroneously thought that bubble sort was always better for small sets of data, rather than appreciating that it is a viable algorithm in terms of time complexity when the number of items n is small. Few identified that bubble sort is better suited to when the list is already nearly sorted.

Question 7* (d)

(d) The programmer uses an Integrated Development Environment (IDE).

Complete the table by identifying **and** describing **three** IDE features that can help the programmer to develop, or debug a program.

IDE feature	Description

[6]

Most candidates scored well on this question which lent itself to a wide variety of potential answers. Some candidates quoted 'autocorrect' rather than auto complete or suggested predictive code. IDEs can have integrated run-time environments and translation software built-in or attached, but some responses were rather vague or included repetition and did not make it clear that they were referring to an integrated system. Sometimes the descriptive expansion or the initial identification of the point was vague and required the identification plus the description boxes to be taken in combination to give the mark. This was particularly evident when candidates gave a very vague 'debugger/debugging' identification.

Question 8 (a)

8 A program is being designed that will allow a user to log into an account on a website using a username and password.

(a) Identify **two** possible inputs and **one** output this program will need.

Input 1

Input 2

Output

[3]

This question was generally well answered by most candidates, but answers had to be given in the context of the scenario. Some less successful responses mistakenly identified input/output devices rather than specific examples of inputs and outputs.

Question 8 (b)

(b) Identify **two** possible sub-procedures that could be used in this program.

1

2

[2]

This question was also generally well answered, but answers had to be given in the context of the scenario. Thinking procedurally is specification point 2.1.3 which requires suitable procedures for a given context to be identified. The context was explicitly for logging in to a system, not for creating user accounts or setting up/validating password construction. The most common responses were checking if the username existed and checking whether the username/password combination was correct.

Section B overview

Section B referred to one specific scenario that revolved around a problem that was solved by using OOP techniques. Those candidates who had very limited practical experience of OOP often produced very little code when it was required.

Question 9 (a) (i)

9 A text-based computer game allows a user to dig for treasure on an island. The island is designed as a grid with 10 rows and 20 columns to store the treasure. Each square is given an x and y coordinate. Some of the squares in the grid store the name of a treasure object. Each treasure object has a value, e.g. 100 and a level, e.g. "Bronze."

(a) The computer game makes use of abstraction.

(i) Describe what is meant by the term abstraction and give an example of how abstraction can be used in the treasure game.

Description:
.....
.....
.....

Example:
.....

[3]

Many candidates identified that abstraction simplified or removed unnecessary detail to gain some marks, but then found it harder to give an example relevant to the scenario. Many candidates gave examples that related to graphics whereas the scenario explicitly stated that the game was text based.

Question 9 (a) (ii)

(ii) Give **three** benefits of using abstraction when writing a program.

1

.....

2

.....

3

.....

[3]

Many candidates gained some marks, but most found it difficult to identify three distinct reasons, and repetition was often observed in responses. Again, some candidate answers did not provide suitable levels of qualification for the points given.

Question 9 (b) (i)

(b) The treasure game is being programmed using an object-oriented paradigm.

A class, `Treasure`, is used to store the treasure objects.

The design for the `Treasure` class, its attributes and methods is shown here.

<code>class: Treasure</code>
<code>attributes: private value : integer private level : string</code>
<code>methods: new() function getValue() function getLevel()</code>

(i) The constructor method takes a value as an integer, e.g. 100, and a level, e.g. "bronze", as parameters and assigns these to the attributes.

Write pseudocode or program code to declare the class `Treasure`.

You should define the attributes and constructor method in your answer.

You do **not** need to write the get methods.

.....

.....

.....

.....

.....

.....

.....

..... **[5]**

This question either tended to be very poorly answered or very well answered. There was a clear division between candidates who were comfortable writing OOP code and class constructors and those that were not.

Weaker candidates with little OOP experience often confused the class declaration with the instantiation method or offered no response. Many candidates often assigned the parameters to the private attribute values rather than setting the attributes to the parameters being passed in.

Where candidates gave responses in programming languages and the intent of the response was clear, marks were given. However, sometimes Python programmers did not make it clear (by convention/comment) that class attributes were private.

Question 9 (b) (ii)

(ii) The get method `getLevel ()` will return the appropriate attribute.

Write the method `getLevel ()` using either pseudocode or program code.

.....
.....
.....
..... [2]

While there were many good responses it was also noted that a significant number of candidates erroneously tried to send a parameter to a `getter()` function and then return the same parameter. Some candidates did not return a value or simply tried to print the value. Other errors included writing the getter method as a function call by omitting any indication that a function was being defined.

Question 9 (b) (iii)

(iii) Describe the object-oriented programming technique being used in part 9(b)(ii).

.....
.....
.....
..... [2]

Some candidates randomly picked an erroneous OOP term like polymorphism. Those candidates who could correctly identify the use of encapsulation did not always go on to specifically state that this meant access was controlled by a public `getter()` method.

Question 9 (c)

(c) A class, Board, is used to store the 10 row (x coordinate) by 20 column (y coordinate) grid.

The design for the Board class, its attributes and methods is shown here.

class: Board
attributes: private grid : Array of Treasure
methods: new() function getGridItem(x, y) function setGridItem(x, y, treasureToInsert)

The constructor initialises each space in the grid to a treasure object with value as -1 and level as an empty string.

Complete the following pseudocode for the constructor method.

```
public procedure new()
  for row = ..... to 9
    for column = 0 to .....
      ..... [row, column] = new Treasure(....., "")
    next .....
  next row
endprocedure
```

[5]

Most candidates scored at least the first 2 mark-points for the grid dimensions, but far fewer could construct OOP code requiring the attribute and the relevant default parameter value to be given.

Question 9 (d)

- (d) A procedure, `guessGrid()`:
- takes a `Board` object as a parameter
 - accepts the row (x) and column (y) coordinates from the user
 - outputs "No treasure" if there is no treasure found at the coordinate (level is an empty string)
 - if there is treasure at that coordinate, it outputs the level and the value of the treasure in an appropriate message.

Write the procedure `guessGrid()` using either pseudocode or program code.

.....

.....

.....

.....

.....

.....

..... [7]

Many candidate responses gave very little beyond the procedure declaration and taking in the x and y coordinates as inputs, thus scoring 2 marks at most. There was less successful understanding of how to use the `get()` methods provided in the scenario to access the data. This is an area of the specification that candidates require extensive practical experience of to be able to fluently answer questions like this in examination conditions.

Many candidates tried to used `grid[x,y]` by inserting the coordinates into the parameter values and did not identify it as an instance of an object that needed its attributes to be accessed via the appropriate getter methods. Those candidates who tried to access the attributes directly without the appropriate getter methods did achieve some further marks with follow through marks.

Competent coders often gave responses worthy of full marks within just a few lines of code.

Exemplar 3

```

def guess.Grid (Board):
Board = get Board
    x = int(input("enter x"))
    y = int(input("enter y"))
    place = Board.get GridItem(x, y)
    if place.get Level() = "":
        print("No treasure")
    else:
        print("Treasure found! valued at" + place.getValue()
            + " it is a " + place.getLevel())

```

This response clearly shows a logical and well-structured algorithm that uses the appropriate get() methods to access the relevant attributes of the passed parameter object. Candidates who are fluent in the use of OOP can present elegant and concise solutions.

Question 9 (e)

(e) Describe **two** benefits of using an object-oriented paradigm rather than a procedural paradigm.

- 1
-
-
-
- 2
-
-
-

[4]

Many candidates were more successful on this part of Section B as it was an AO1 book learnt topic rather than the preceding AO2 OOP programming application elements. However, benefits were often poorly described, or OOP components were just named without a relevant expansion as to how they could be used.

Question 9 (f)*

- (f)* The main program initialises a new instance of `Board`. The programmer is considering declaring this as a global variable or as a local variable and then passing this into the subroutines that control the game.

Compare the use of variables and parameters in this game.

You should include the following in your answer:

- what is meant by a local variable and global variable
- how local and global variables can be used in this program
- the use of passing parameters by value and by reference.

[9]

A good range of Level 2 responses were observed with competent descriptive definitions of local versus global and byVal/byRef parameter passing.

There were far fewer high level AO3 evaluations of relative memory use within the context of the scenario when passing byVal/byRef for a relatively large grid object. Good, contextualised responses took the elements of the scenario in section B of the paper and talked about the grid object and passing parameters like x and y coordinates. Responses such as this were relatively rare.

Assessment for learning



Candidates would benefit from producing solutions centred around the scenario presented in Section B of the paper. Having extensive OOP programming experience in a relevant high level language will help candidates to successfully tackle questions where algorithms have to be presented. Sample Python code based on Section B is presented below for consideration.

Assessment for learning

```
# 2023 Section B

class Treasure:

    def __init__(self, pValue: int, pLevel: str):
        self.__value = pValue
        self.__level = pLevel

    def getValue(self):
        return self.__value

    def setValue(self, pValue: int):
        return self.__pValue

    def getLevel(self):
        return self.__level

    def setLevel(self, pLevel: str):
        return self.__pLevel

class Board():

    def __init__(self):
        self.__grid = [[Treasure(-1, "") for _ in range(20)] for _ in range(10)]

    def getGridItem(self, x: int, y: int):
        return self.__grid[x][y]

    def setGridItem(self, x: int, y: int, pTreasure: Treasure):
        self.__grid[x][y] = pTreasure

def guessGrid(b: Board):
    row = int(input("Row: "))
    col = int(input("Col: "))
    t = b.getGridItem(row, col)
    if t.getLevel() == "":
        print("No treasure")
    else:
        print(f"Treasure found at {row} {col}!")
        print(f"Level {t.getLevel()} Value {t.getValue()}")

# Test code - Treasure @ 2,2 all other locations no treasure
island = Board()
prize = Treasure(5, "Prize!")
island.setGridItem(2, 2, prize)
guessGrid(island)
```

Supporting you

Teach Cambridge

Make sure you visit our secure website [Teach Cambridge](#) to find the full range of resources and support for the subjects you teach. This includes secure materials such as set assignments and exemplars, online and on-demand training.

Don't have access? If your school or college teaches any OCR qualifications, please contact your exams officer. You can [forward them this link](#) to help get you started.

Reviews of marking

If any of your students' results are not as expected, you may wish to consider one of our post-results services. For full information about the options available visit the [OCR website](#).

Access to Scripts

For the June 2023 series, Exams Officers will be able to download copies of your candidates' completed papers or 'scripts' for all of our General Qualifications including Entry Level, GCSE and AS/A Level. Your centre can use these scripts to decide whether to request a review of marking and to support teaching and learning.

Our free, on-demand service, Access to Scripts is available via our single sign-on service, My Cambridge. Step-by-step instructions are on our [website](#).

Keep up-to-date

We send a monthly bulletin to tell you about important updates. You can also sign up for your subject specific updates. If you haven't already, [sign up here](#).

OCR Professional Development

Attend one of our popular CPD courses to hear directly from a senior assessor or drop in to a Q&A session. Most of our courses are delivered live via an online platform, so you can attend from any location.

Please find details for all our courses for your subject on **Teach Cambridge**. You'll also find links to our online courses on NEA marking and support.

Signed up for ExamBuilder?

ExamBuilder is the question builder platform for a range of our GCSE, A Level, Cambridge Nationals and Cambridge Technicals qualifications. [Find out more](#).

ExamBuilder is **free for all OCR centres** with an Interchange account and gives you unlimited users per centre. We need an [Interchange](#) username to validate the identity of your centre's first user account for ExamBuilder.

If you do not have an Interchange account please contact your centre administrator (usually the Exams Officer) to request a username, or nominate an existing Interchange user in your department.

Active Results

Review students' exam performance with our free online results analysis tool. It is available for all GCSEs, AS and A Levels and Cambridge Nationals.

[Find out more](#).

Need to get in touch?

If you ever have any questions about OCR qualifications or services (including administration, logistics and teaching) please feel free to get in touch with our customer support centre.

Call us on
01223 553998

Alternatively, you can email us on
support@ocr.org.uk

For more information visit

 **ocr.org.uk/qualifications/resource-finder**

 **ocr.org.uk**

 **facebook.com/ocrexams**

 **twitter.com/ocrexams**

 **instagram.com/ocrexaminations**

 **linkedin.com/company/ocr**

 **youtube.com/ocrexams**

We really value your feedback

Click to send us an autogenerated email about this resource. Add comments if you want to. Let us know how we can improve this resource or what else you need. Your email address will not be used or shared for any marketing purposes.



I like this



I dislike this

Please note – web links are correct at date of publication but other websites may change over time. If you have any problems with a link you may want to navigate to that organisation's website for a direct search.



OCR is part of Cambridge University Press & 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. © OCR 2023 Oxford Cambridge and RSA Examinations is a Company Limited by Guarantee. Registered in England. Registered office The Triangle Building, Shaftesbury Road, Cambridge, CB2 8EA. Registered company number 3484466. OCR is an exempt charity.

OCR operates academic and vocational qualifications regulated by Ofqual, Qualifications Wales and CCEA as listed in their qualifications registers including A Levels, GCSEs, Cambridge Technicals and Cambridge Nationals.

OCR provides resources to help you deliver our qualifications. These resources do not represent any particular teaching method we expect you to use. We update our resources regularly and aim to make sure content is accurate but please check the OCR website so that you have the most up to date version. OCR cannot be held responsible for any errors or omissions in these resources.

Though we make every effort to check our resources, there may be contradictions between published support and the specification, so it is important that you always use information in the latest specification. We indicate any specification changes within the document itself, change the version number and provide a summary of the changes. If you do notice a discrepancy between the specification and a resource, please [contact us](#).

You can copy and distribute this resource freely if you keep the OCR logo and this small print intact and you acknowledge OCR as the originator of the resource.

OCR acknowledges the use of the following content: N/A

Whether you already offer OCR qualifications, are new to OCR or are thinking about switching, you can request more information using our [Expression of Interest form](#).

Please [get in touch](#) if you want to discuss the accessibility of resources we offer to support you in delivering our qualifications.