Bishop Challoner
Computer Science Department

A Level Computer Science Revision Pack

02 – Algorithms and Programming

The mark scheme for each paper follows the questions

Included (in order of appearance)
2019
2018
2017

How to revise Computer Science
Practice questions from past papers are one of the best methods of revising topics from the course. This approach, accompanied by creating notes and reading the course textbook as a source for information, has proven successful for many of our previous students.

How to revise a particular topic
this is generic and by no means a one size fits all approach

1. On a single sheet of A4, write down everything you currently know about the topic. Do this prior to reading the course textbook or seeking help from previous notes.
2. Now consult course textbook for the topic and add to this sheet, anything you did not know that is necessary – once complete, highlight these points – these are the areas you need to learn.
3. Locate questions based around this topic in the past paper pack and attempt to answer them.
4. Confirm with the mark scheme as to your success in answering the question.

The end goal of this approach would be that you are comfortably able to produce a piece of A4 for each topic of the course and then apply this information to the past paper questions.

Obtaining feedback for answers
The students who succeed the best in computer science are those who seek constant feedback from teachers, not just in the scope of a lesson. Any work you produce out of lesson such as past paper question answers or programming challenges, you should want to seek feedback for. This can be achieved by:

1. Taking work to a teacher during school time.
2. Emailing a teacher your answers, questions etc.

Mr Ravenscroft – l.ravenscroft@bishopchalloner.bham.sch.uk
Mr Ebrahim – b.ebrahim@bishopchalloner.bham.sch.uk

As your teachers we want to give you feedback!

Study Skills and Support

Exam board: OCR

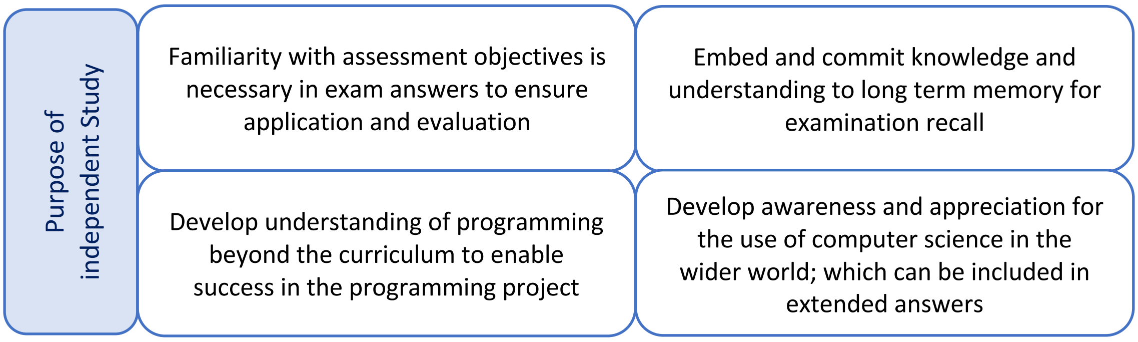
Course length: 2 years

How is it assessed? 2 written exams on 01 – Computer Systems and 02 – Algorithms and Programming (each worth 40%) and a programming project worth 20%.

Modules covered:

01 – Processors, Input – Output and Storage, Systems software, software development, compression, databases, networks, web technologies, data types, data structures, Boolean algebra, morals and ethics.

02 – Thinking abstractly; ahead; procedurally; logically; concurrently, programing techniques, computational methods, algorithms.



Purpose of independent Study

Familiarity with assessment objectives is necessary in exam answers to ensure application and evaluation

Embed and commit knowledge and understanding to long term memory for examination recall

Develop understanding of programming beyond the curriculum to enable success in the programming project

Develop awareness and appreciation for the use of computer science in the wider world; which can be included in extended answers

Resource	Link	Useful For…	Requirements
Course Textbook	N/A	Independent revision & study	Course textbook from the school library.
YouTube	YouTube	Knowledge booster, second voice	N/A
Past Paper Packs	N/A	Exam style question practice, independent study	Past paper pack from class teacher
Departmental resources	All stored within the Microsoft Teams Team for the group.	Accessing departmental materials and lessons	School email and password login.
Mr Fraser	www.mrfraser.org	Accessing resources and work sheets	mrfraser.org login account (free to create)
Craig n Dave	craigndave.org	Resources for topics – broken down by spec	Access is free for most content – school has a paid account
AQA Past Papers	Search 'AQA A Level Computer Science Past Papers' on Google.	Different phrasing of exam style questions.	N/A
Class Teachers	l.ravenscroft@bishopchalloner.bham.sch.uk b.ebrahim@bishopchalloner.bham.sch.uk		

OCR
Oxford Cambridge and RSA

Tuesday 11 June 2019 – Morning

A Level Computer Science

H446/02 Algorithms and programming

Time allowed: 2 hours 30 minutes

You may use:
- a ruler (cm/mm)
- an HB pencil

Do not use:
- a calculator

Please write clearly in black ink. **Do not write in the barcodes.**

Centre number □□□□□ Candidate number □□□□

First name(s) ____________________

Last name ____________________

INSTRUCTIONS
- Use black ink.
- 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 **140**.
- The marks for each question are shown in brackets **[ ]**.
- Quality of extended responses will be assessed in questions marked with an asterisk (*).
- This document consists of **24** pages.

© OCR 2019 [601/4911/5]

OCR is an exempt Charity

Turn over

Section A

1 The temperatures of an ocean are input into a computer system. They are recorded, and will be accessed, in the order in which they arrive. The data for one week is shown:

5, 5.5, 5, 6, 7, 6.5, 6

(a) The data is to be stored in a data structure. The programmer stores the data in a queue.

Explain why a queue is used instead of a stack.

....

....

....

.... **[2]**

(b) The data is processed. After processing, the value for the first day is stored as 0. The value for each following day is stored as an increase, or decrease, from the first day.

For example: if the first day was 7, the second was 6 and the third was 9, after processing it would be stored as 0, –1, 2.

(i) The queue uses `dequeue()` to return the first element of the queue.

`dequeue()` is a function.

Explain why `dequeue()` is a function, not a procedure.

....

.... **[1]**

(ii) Complete the algorithm to process the data in the queue and store the results in an array called `processedData`.

```
processedData[0] = 0

firstDay = ..............................

for count = 1 to 6

    processedData[........................] = dequeue() – ..........................

next count
```

[3]

(iii) The contents of `processedData` are shown.

0	0.5	0	1	2	1.5	1

The data needs to be sorted into ascending order.

Explain how a bubble sort algorithm sorts data. Use the current contents of `processedData` in your explanation.

....

....

....

....

....

....

....

....

....

.... **[5]**

(iv) A bubble sort has the following complexities:

Best time	O(n)
Average and worst time	O(n^2)
Worst space	O(1)

Describe what each of these complexities mean.

Best time O(n)

....

....

....

....

Average and worst time O(n^2)

....

....

....

....

Worst Space O(1)

....

....

....

....

[6]

© OCR 2019

2 A program needs to store the names of plants that are in a garden, so they can be easily found and accessed in alphabetical order.

The data is stored in a tree structure. Part of the tree is shown.

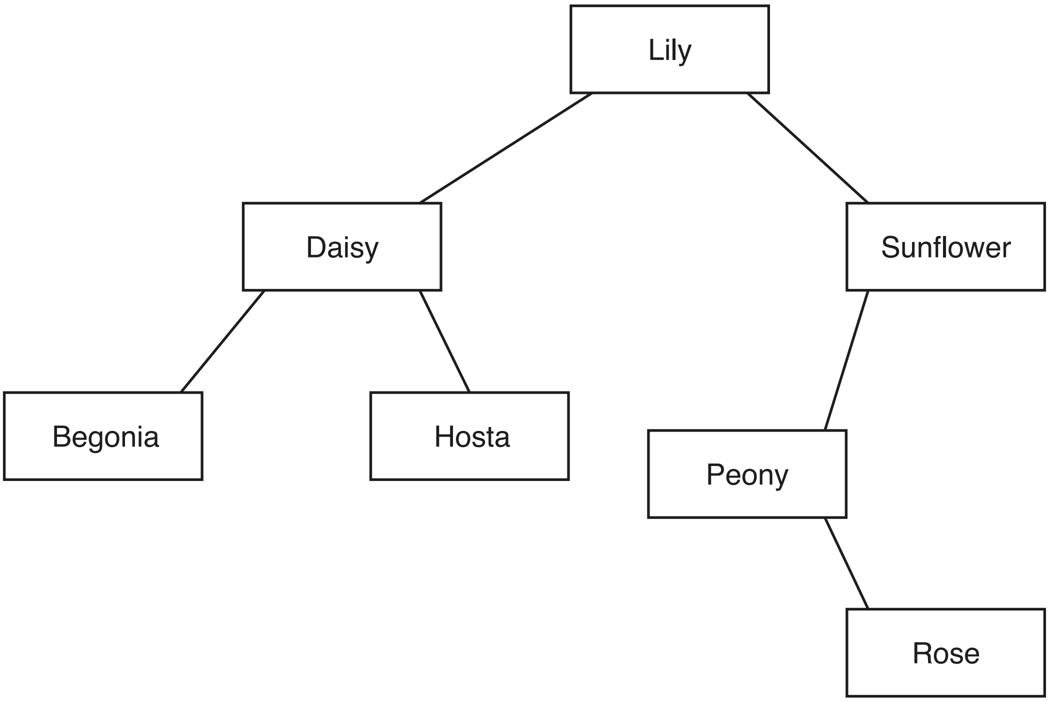


Fig. 2.1

(a) **(i)** State the type of tree shown in Fig. 2.1.

.... **[1]**

(ii) Show the output of a breadth-first traversal of the tree shown in Fig. 2.1.

....

....

....

....

....

....

....

.... **[3]**

© OCR 2019

(iii) Explain how backtracking is used in a depth-first (post-order) traversal. Use the tree in Fig. 2.1 in your explanation.

....

....

....

....

....

....

....

....

....

....

....**[4]**

(b) The elements in the tree in Fig. 2.1 are read into a linked list producing an alphabetised list.

(i) Complete the following table to show the linked list for the data.

Data item	Data	`NextPointer`
0	Begonia	
1	Daisy	
2	Hosta	
3	Lily	
4	Peony	
5	Rose	
6	Sunflower	

[2]

(ii) A new plant, Lavender, needs adding to the linked list. The linked list needs to retain its alphabetical order.

Complete the table to show the linked list after Lavender is added.

Data item	Data	`NextPointer`
0	Begonia	
1	Daisy	
2	Hosta	
3	Lily	
4	Peony	
5	Rose	
6	Sunflower	

[3]

(iii) Hosta needs removing from the linked list.

Explain how a data item is removed from a linked list. Use the removal of Hosta in your answer.

....

....

....

....

....

....

....

.... **[4]**

© OCR 2019

(iv) The linked list is stored as a 2D array with the identifier `plantList`. The index of the first element of the linked list is stored in the identifier `firstElement`.

All contents of the linked list need to be output in alphabetical order.

Write an algorithm to follow the pointers to output the contents of the linked list in alphabetical order.

Add comments to explain your code.

....

....

....

....

....

....

....

....

....

....

....

.... **[5]**

3 A recursive function, `GCD`, is given in pseudocode.

```
function GCD(num1, num2)
        if num2 == 0 then
                return num1
        else
                return GCD(num2, num1 MOD num2)
        endif
endfunction
```

(a) The function uses branching.

(i) Identify the type of branching statement used in the function.

.... **[1]**

(ii) Explain the difference between branching and iteration.

....

....

....

.... **[2]**

(iii) Identify the **two** parameters in the function.

1 ....

2 ....
[1]

(iv) State whether the parameters should be passed by value, or by reference. Justify your answer.

....

....

....

.... **[2]**

© OCR 2019

(v) Describe the arithmetic operation of `MOD`. Use an example in your answer.

....

....

....

....

....

.... **[2]**

(b) Trace the recursive function when it is called by the statement `GCD(250, 20)`. Give the final value returned.

....

....

....

....

....

....

....

....

....

Final return value: ....

[3]

(c) The function has been rewritten using iteration instead of recursion.

(i) State **one** benefit and **one** drawback of using iteration instead of recursion.

Benefit ....

....

Drawback ....

....

[2]

© OCR 2019

(ii) Complete the missing statements in this iterative version of the function.

```
function newGCD(num1, num2)
  temp = 0
  while (num2 != ..................................)
    .................................. = num2
    num2 = num1 MOD ..................................
    num1 = temp
  endwhile
  return ..................................
endfunction
```

[4]

4 Mabel is a software engineer. She is writing a computer game for a client. In the game the main character has to avoid their enemies. This becomes more difficult as the levels of the game increase.

(a) Mabel uses decomposition to design the program.

Explain how decomposition can aid the design of this program.

....

....

....

.... **[2]**

(b) The computer game allows a user to select a character (e.g. name, gender). They can then choose a level for the game (easy, normal, challenging). The user controls their character by moving it left or right. The character can jump using space bar as an input. If the character touches one of the enemies then it loses a life. The character has to make it to the end of the level without losing all their lives.

The game is designed in a modular way.

(i) One sub-procedure will handle the user input.

Describe **three** other sub-procedures Mabel could create for the given game description.

1 ....

....

....

....

2 ....

....

....

....

3 ....

....

....

....
[6]

(ii) Describe the decision that the program will need to make within the user input sub-procedure and the result of this decision.

....

....

....

.... **[2]**

(iii) Define pipelining and give an example of how it could be applied in the program.

....

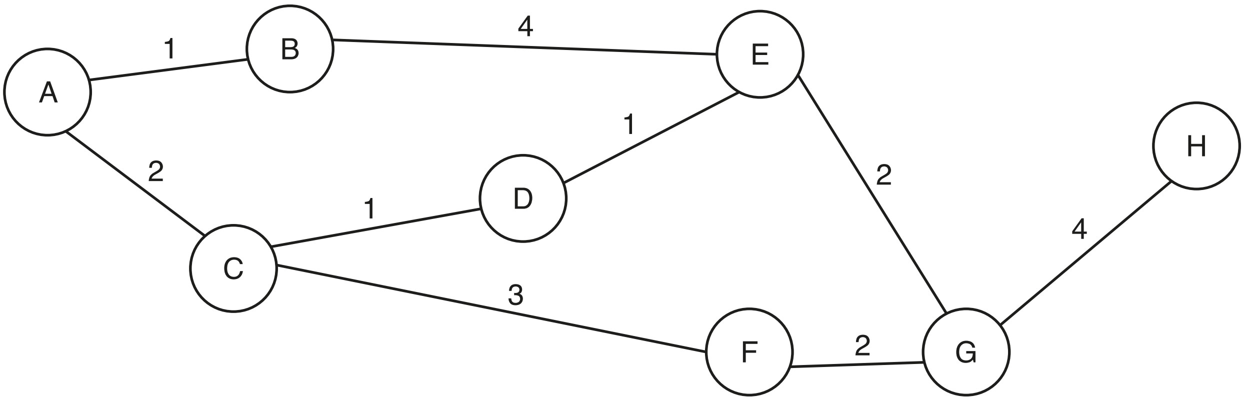
....

....

.... **[2]**

(c) The game's 'challenging' level has intelligent enemies that hunt down the character in an attempt to stop the user from winning. The program plans the enemies' moves in advance to identify the most efficient way to stop the user from winning the game.

The possible moves are shown in a graph. Each node represents a different state in the game. The lines represent the number of moves it will take to get to that state.



Show how Dijkstra's algorithm would find the shortest path from A to H.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

© OCR 2019

(d)* Mabel has been told that true programmers write programs in a text editor, and do not use IDEs. Mabel does not agree with this statement.

Discuss the use of an IDE in the development of this program.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[9]**

5 A 1-dimensional array stores the following data:

Index	**0**	**1**	**2**	**3**	**4**	**5**
Data	2	18	6	4	12	3

(a) The array needs sorting into descending order.

Describe how a merge sort would sort the given array into descending order.

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

(b) An insertion sort can be used to sort the array instead of a merge sort.

Explain why an insertion sort might use less memory than a merge sort.

....

....

....

.... **[2]**

© OCR 2019

6* Benedict runs a social networking website. He has been told he should use data mining to help him enhance and improve his website.

Evaluate the use of data mining to help Benedict enhance and improve his social networking website.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[9]**

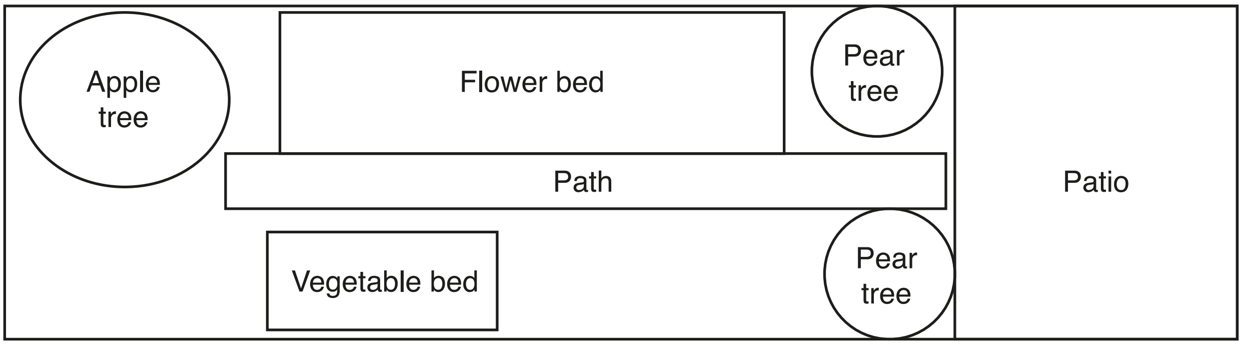
BLANK PAGE

PLEASE DO NOT WRITE ON THIS PAGE

Section B

7 A program is needed to plan the layout of a garden.

The program will allow the user to create an image of the garden, for example:



(a) The programmer will use abstraction to produce the program interface to represent the garden.

(i) Give **two** different examples of how abstraction has been used to produce the layout of the garden.

1 ....

....

2 ....

....
[2]

(ii) Explain the need for abstraction in the production of this program.

....

....

....

....

....

.... **[3]**

© OCR 2019

(iii) The user needs to input data into the program to set up their garden layout.

Identify **three** pieces of data that the user may input into this program.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) The program is to be built using object oriented programming.

All items that can be added to the garden are declared as instances of the class `GardenItem`.

The class has the following attributes:

Attribute	**Description**	**Example**
`itemName`	The name of the item	Flowerbed
`length`	The length of the item in metres	2
`width`	The width of the item in metres	1

(i) The constructor method sets the attributes to values that are passed as parameters.

Write pseudocode or program code to declare the class `GardenItem` and its constructor. All attributes should be private and initialised through the constructor (e.g. `daisies = new GardenItem("Flowerbed",2,1)`). **[4]**

....

....

....

....

....

....

....

....

....

....

....

....

(ii) The trees in the garden layouts are defined by the class `Tree`. This class inherits from `GardenItem`.

The class `Tree` has the additional attributes: `height`, `sun`, `shade`.

If `sun` is `true` then the tree can grow in full sun, if it is `false` then it cannot.

If `shade` is `true` then the tree can grow in full shade, if it is `false` then it cannot.

The length and width of a tree are the same. Only one value for these measurements is passed to the constructor.

Write an algorithm, using pseudocode or program code, to declare the class `Tree`. Declare all attributes as private.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[5]**

(iii) The Common Oak is a type of tree. It has a maximum height, length and width of 40 m. It can grow in sun and shade.

Write a statement, using pseudocode or program code, to declare an instance of tree for the Common Oak. Give the object the identifier `firstTree`.

....

....

.... **[4]**

(iv) The classes `GardenItem` and `Tree` use get and set methods to access and alter their private attributes.

Write the get method `getItemName` and set method `setItemName` for class `GardenItem`. The set method takes the new value as a parameter.

Do not write any other methods, or re-declare the class.

....

....

....

....

....

....

....

....

....

.... **[4]**

(v) The trees in the garden layouts are stored in a 1-dimensional array, `treeArray`. The array can store a maximum of 1000 items. The array has global scope.

A procedure, `findTree`, takes as parameters:

- The maximum height of a tree
- The maximum width of a tree
- Whether the tree can live in full sun
- Whether the tree can live in full shade.

It searches the array, `treeArray`, for all trees that do not exceed the maximum height and width, and that can grow in the conditions available. If there are no suitable trees, a suitable message is output.

It outputs the name and details of the trees found in an appropriate message.

Call the get methods, `getItemName`, `getHeight`, `getWidth`, `getSun`, `getShade`, to access the attributes.

Write, using pseudocode or program code, the procedure `findTree`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

(c)* The programmer is designing the program to make use of caching and re-useable components.

Explain and evaluate the use of caching and re-useable components in the design of the garden program.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[9]**

PLEASE DO NOT WRITE ON THIS PAGE

Oxford Cambridge and RSA

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) 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 OCR Copyright Team, The Triangle Building, Shaftesbury Road, Cambridge CB2 8EA.

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.

© OCR 2019

GCE

Computer Science

H446/02: Algorithms and programming

Advanced GCE

Mark Scheme for June 2019

OCR (Oxford Cambridge and RSA) is a leading UK awarding body, providing a wide range of qualifications to meet the needs of candidates of all ages and abilities. OCR qualifications include AS/A Levels, Diplomas, GCSEs, Cambridge Nationals, Cambridge Technicals, Functional Skills, Key Skills, Entry Level qualifications, NVQs and vocational qualifications in areas such as IT, business, languages, teaching/training, administration and secretarial skills.

It is also responsible for developing new specifications to meet national requirements and the needs of students and teachers. OCR is a not-for-profit organisation; any surplus made is invested back into the establishment to help towards the development of qualifications and support, which keep pace with the changing needs of today's society.

This mark scheme is published as an aid to teachers and students, to indicate the requirements of the examination. It shows the basis on which marks were awarded by examiners. It does not indicate the details of the discussions which took place at an examiners' meeting before marking commenced.

All examiners are instructed that alternative correct answers and unexpected approaches in candidates' scripts must be given marks that fairly reflect the relevant knowledge and skills demonstrated.

Mark schemes should be read in conjunction with the published question papers and the report on the examination.

© OCR 2019

Annotations

Annotation	Meaning
^	Omission mark
BOD	Benefit of the doubt
✗	Incorrect point
FT	Follow through
NAQ	Not answered question
NBOD	No benefit of doubt given
REP	Repeat
✓	Correct point
TV	Too vague
0	Zero (big)
L1	Level 1
L2	Level 2
L3	Level 3

Subject-specific Marking Instructions

INTRODUCTION

Your first task as an Examiner is to become thoroughly familiar with the material on which the examination depends. This material includes:

- the specification, especially the assessment objectives
- the question paper and its rubrics
- the mark scheme.

You should ensure that you have copies of these materials.

You should ensure also that you are familiar with the administrative procedures related to the marking process. These are set out in the OCR booklet **Instructions for Examiners**. If you are examining for the first time, please read carefully **Appendix 5 Introduction to Script Marking: Notes for New Examiners**.

Please ask for help or guidance whenever you need it. Your first point of contact is your Team Leader.

USING THE MARK SCHEME

Please study this Mark Scheme carefully. The Mark Scheme is an integral part of the process that begins with the setting of the question paper and ends with the awarding of grades. Question papers and Mark Schemes are developed in association with each other so that issues of differentiation and positive achievement can be addressed from the very start.

This Mark Scheme is a working document; it is not exhaustive; it does not provide ‘correct’ answers. The Mark Scheme can only provide ‘best guesses’ about how the question will work out, and it is subject to revision after we have looked at a wide range of scripts.

The Examiners’ Standardisation Meeting will ensure that the Mark Scheme covers the range of candidates’ responses to the questions, and that all Examiners understand and apply the Mark Scheme in the same way. The Mark Scheme will be discussed and amended at the meeting, and administrative procedures will be confirmed. Co-ordination scripts will be issued at the meeting to exemplify aspects of candidates’ responses and achievements; the co-ordination scripts then become part of this Mark Scheme.

Before the Standardisation Meeting, you should read and mark in pencil a number of scripts, in order to gain an impression of the range of responses and achievement that may be expected.

In your marking, you will encounter valid responses which are not covered by the Mark Scheme: these responses must be credited. You will encounter answers which fall outside the ‘target range’ of Bands for the paper which you are marking. Please mark these answers according to the marking criteria.

Please read carefully all the scripts in your allocation and make every effort to look positively for achievement throughout the ability range. Always be prepared to use the full range of marks.

LEVELS OF RESPONSE QUESTIONS:

The indicative content indicates the expected parameters for candidates' answers, but be prepared to recognise and credit unexpected approaches where they show relevance.
Using 'best-fit', decide first which set of BAND DESCRIPTORS best describes the overall quality of the answer. Once the band is located, adjust the mark concentrating on features of the answer which make it stronger or weaker following the guidelines for refinement.

- **Highest mark:** If clear evidence of all the qualities in the band descriptors is shown, the HIGHEST Mark should be awarded.
- **Lowest mark:** If the answer shows the candidate to be borderline (i.e. they have achieved all the qualities of the bands below and show limited evidence of meeting the criteria of the band in question) the LOWEST mark should be awarded.
- **Middle mark:** This mark should be used for candidates who are secure in the band. They are not 'borderline' but they have only achieved some of the qualities in the band descriptors.

Be prepared to use the full range of marks. Do not reserve (e.g.) high Band 3 marks 'in case' something turns up of a quality you have not yet seen. If an answer gives clear evidence of the qualities described in the band descriptors, reward appropriately.

	AO1	**AO2**	**AO3**
High (thorough)	Precision in the use of question terminology. Knowledge shown is consistent and well-developed. Clear appreciation of the question from a range of different perspectives making extensive use of acquired knowledge and understanding.	Knowledge and understanding shown is consistently applied to context enabling a logical and sustained argument to develop. Examples used enhance rather than detract from response.	Concerted effort is made to consider all aspects of a system / problem or weigh up both sides to an argument before forming an overall conclusion. Judgements made are based on appropriate and concise arguments that have been developed in response resulting in them being both supported and realistic.
Middle (reasonable)	Awareness of the meaning of the terms in the question. Knowledge is sound and effectively demonstrated. Demands of question understood although at times opportunities to make use of acquired knowledge and understanding not always taken.	Knowledge and understanding applied to context. Whilst clear evidence that an argument builds and develops through response there are times when opportunities are missed to use an example or relate an aspect of knowledge or understanding to the context provided.	There is a reasonable attempt to reach a conclusion considering aspects of a system / problem or weighing up both sides of an argument. However the impact of the conclusion is often lessened by a lack of supported judgements which accompany it. This inability to build on and develop lines of argument as developed in the response can detract from the overall quality of the response.
Low (basic)	Confusion and inability to deconstruct terminology as used in the question. Knowledge partial and superficial. Focus on question narrow and often one-dimensional.	Inability to apply knowledge and understanding in any sustained way to context resulting in tenuous and unsupported statements being made. Examples if used are for the most part irrelevant and unsubstantiated.	Little or no attempt to prioritise or weigh up factors during course of answer. Conclusion is often dislocated from response and any judgements lack substance due in part to the basic level of argument that has been demonstrated throughout response.

	Assessment Objective
AO1	Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.
AO1.1	Demonstrate **knowledge** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO1.2	Demonstrate **understanding** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO2	Apply knowledge and understanding of the principles and concepts of computer science including to analyse problems in computational terms.
AO2.1	Apply knowledge and understanding of the principles and concepts of computer science.
AO2.2	Analyse problems in computational terms.
AO3	Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.
AO3.1	Design computer systems that solve problems.
AO3.2	Program computer systems that solve problems.
AO3.3	Evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

Question	Answer	Marks	Guidance
1a	1 mark per bullet - Queue outputs data in a First In First Out fashion - It will retrieve the temperature values in the order they were recorded or - Stack outputs the data in a Last In First Out fashion - It will retrieve the temperature values in the reverse of the order they were recorded	**2** **AO1.2 (1)** **AO2.2 (1)**	Mark Point 1 is the definition Mark Point 2 is for context of the temperature values
1bi	It returns a value	**1** **AO2.1 (1)**	

1bii	1 mark per completed word
	`processedData[0] = 0`
	`firstDay =` **`dequeue()`**
	`for count = 1 to 6`
	`    processedData[`**`count`**`] = dequeue() -` **`firstDay`**
	`next count`
		3
		AO2.2 (1)
		AO3.2 (2)
			Exact answers only

<table>
<tr><td>1biii</td><td>1 mark per bullet to max 5.
Max 3 if no application to data in <code>processedData</code>
• Compares each pair of data e.g. 0 and 0.5
• If they are in the correct order it moves to the next pair e.g. 0.5 and 0
• If they are in the wrong order it swaps them e.g. 0.5 and 0 becomes 0 and 0.5
• Continues to the end of the array e.g. Pass 1 complete
• If there has been a swap it checks again e.g. Pass 2 complete
• If there have been no swaps it is sorted
<table>
<tr><td>0</td><td>0.5</td><td>0</td><td>1</td><td>2</td><td>1.5</td><td>1</td><td rowspan="4">Pass 1</td></tr>
<tr><td>0</td><td>0</td><td>0.5</td><td>1</td><td>2</td><td>1.5</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0.5</td><td>1</td><td>1.5</td><td>2</td><td>1</td></tr>
<tr><td>0</td><td>0</td><td>0.5</td><td>1</td><td>1.5</td><td>1</td><td>2</td></tr>
<tr><td>0</td><td>0</td><td>0.5</td><td>1</td><td>1</td><td>1.5</td><td>2</td><td>Pass 2</td></tr>
<tr><td>0</td><td>0</td><td>0.5</td><td>1</td><td>1</td><td>1.5</td><td>2</td><td>No swaps</td></tr>
</table></td><td><b>5</b>

<b>AO1.1 (2)</b>
<b>AO1.2 (1)</b>
<b>AO2.1 (1)</b>
<b>AO2.2 (1)</b></td><td>Allow (full) credit for tables showing the bubblesort being completed.</td></tr>
<tr><td>1biv</td><td>1 mark per bullet
O(n)
• <u>Linear</u>
• Best time grows at the same rate as the number of elements
• This is the case when the data is already in order
O(n²)
• <u>Polynomial</u> / <u>Quadratic</u>
• Worst and average time is proportional to the square (polynomial) of the number of elements
• Worst case is when the data is initially in the reverse order
O(1)
• <u>Constant</u>
• Will always take the same amount of memory (in addition to the list itself).</td><td><b>6</b>

<b>AO1.1 (3)</b>
<b>AO1.2 (3)</b></td><td>Note: First Mark Point is for the identification, second Mark Point is for the description

Note: Do not allow descriptions relating to time complexity for 'Worst Space O(1)'

Note: Do not allow ‘equal to’ in descriptions, O(n) and O(n²) grow in <i>proportion</i> to the number of items</td></tr>
</table>

2ai	<u>Binary Tree</u> / <u>Binary Search Tree</u>	1 **AO2.1 (1)**	
2aii	1 mark per bullet • 1st layer: Lily • 2nd layer: Daisy, Sunflower • 3rd layer: Begonia, Hosta, Peony 4th layer: Rose	3 **AO1.2 (1)** **AO2.1 (1)** **AO2.2 (1)**	
2aiii	1 mark per bullet to max 4. Max 2 marks for no application to the tree. • Depth first starts at the root (Lily) • …and goes all the way down one branch to the bottom (Begonia) • It stores which nodes it has visited / pushes nodes visited onto a stack • When it cannot go any further • …It then backtracks/returns to the previous node • And continues to backtrack until a node is reached with unvisited children. • …and checks down that branch • In the tree shown, after visiting Begonia, the algorithm would backtrack to Daisy… • …and would then visit Hosta *(Accept any other example)*	4 **AO1.1 (1)** **AO1.2 (1)** **AO2.1 (1)** **AO2.2 (1)**	

<table>
<tr>
<td>2bi</td>
<td>1 mark per bullet
• Correct NextPointer values
• Suitable end/null pointer
<table>
<tr><th>Data item</th><th>Data</th><th>NextPointer</th></tr>
<tr><td>0</td><td>Begonia</td><td><b>1</b></td></tr>
<tr><td>1</td><td>Daisy</td><td><b>2</b></td></tr>
<tr><td>2</td><td>Hosta</td><td><b>3</b></td></tr>
<tr><td>3</td><td>Lily</td><td><b>4</b></td></tr>
<tr><td>4</td><td>Peony</td><td><b>5</b></td></tr>
<tr><td>5</td><td>Rose</td><td><b>6</b></td></tr>
<tr><td>6</td><td>Sunflower</td><td>null</td></tr>
</table>
</td>
<td>2
<b>AO2.1</b>
<b>(2)</b></td>
<td>Exact values only. Allow -1 for null pointer or equivalent such as Φ.
Do not allow a blank or 0.</td>
</tr>
<tr>
<td>2bii</td>
<td>1 mark per bullet
• Lavender added in position 7
• …Hosta points to 7
• …Lavender points to 3
<table>
<tr><th>Data item</th><th>Data</th><th>NextPointer</th></tr>
<tr><td>0</td><td>Begonia</td><td>1</td></tr>
<tr><td>1</td><td>Daisy</td><td>2</td></tr>
<tr><td>2</td><td>Hosta</td><td><b>7</b></td></tr>
<tr><td>3</td><td>Lily</td><td>4</td></tr>
<tr><td>4</td><td>Peony</td><td>5</td></tr>
<tr><td>5</td><td>Rose</td><td>6</td></tr>
<tr><td>6</td><td>Sunflower</td><td>null</td></tr>
<tr><td>7</td><td>Lavender</td><td><b>3</b></td></tr>
</table>
</td>
<td>3
<b>AO1.2</b>
<b>(1)</b>
<b>AO2.2</b>
<b>(2)</b></td>
<td>Do not credit answers that do not place lavender in position 7 and then update pointer positions</td>
</tr>
</table>

2biii	1 mark per bullet - Traverse the list to the item immediately prior to the item to be removed (1) - … which is DataItem 1 - Daisy - Find the value of the NextPointer of the item to be removed - … which is the NextPointer of DataItem 2 - Hosta, value 7 - Set the nextPointer of the item prior to the item to be removed to the NextPointer value of the DataItem to be removed - … update the NextPointer of DataItem 1 - Daisy from 2 to 7 (Lavender)	4 **AO1.1 (1)** **AO1.2 (1)** **AO2.2 (2)**	Find the *item before* item to be deleted (Daisy) Find nextPtr of item to be deleted (Hosta) Update nextPtr of the *item before (Daisy)* to the nextPtr of item to be deleted (Hosta) i.e. Daisy 2 is updated to Daisy 7 Allow FT from 2b(ii/iii) if candidate has used table in fig 2.1 (e.g. Daisy would now point to Lily at position 3)

2biv	1 mark per bullet
	- Start at the `firstElement` in the list
	- Correctly looping until null pointer found / end of list
	- Outputting the data element
	- Accessing the pointer to the next element
	- Appropriate comment(s)
	
	e.g.
	`currentElement = firstElement`
	`while(currentElement != null) //Continue until last node`
	`  print(plantList[currentElement,0])`
	`  currentElement = plantList[currentElement,1]`
	`endwhile`
		5
		
		AO2.1 (1)
		AO2.2 (2)
		AO3.2 (2)
			Note: Solution must utilise pointers in a linked list; it cannot use a FOR loop as the number of elements is not known and the data is not in order by index number
			
			Note: identifiers given in the question as `plantList` and `firstElement` should be used accurately in the solution
			
			Note: allow credit for answers that interpret the data structure as an array of records/structures with data/pointer fields

3ai	<u>if</u>	1 **AO1.1 (1)**	
3aii	1 mark per bullet • Branching decides which code is run / only runs code once • Iteration repeatedly runs the same code in the same sequence	2 **AO1.2 (2)**	
3aiii	num1, num2	1 **AO2.1 (1)**	Exact identifier names required
3aiv	1 mark per bullet • By Value • … the original values do not need to be modified • … byRef would not work / would cause the routine to crash	2 **AO2.2 (2)**	
3av	1 mark per bullet • Gives the remainder after division • E.g. 10 MOD 3 = 1	2 **AO1.1 (1)** **AO1.2 (1)**	
3b	1 mark per bullet to max 3 • Num2 != 0 therefore return GCD(20,10) • Num2 != 0 therefore return GCD(10,0) • Final return value = 10	3 **AO2.1 (1)** **AO2.2 (2)**	Allow FT for numerical errors

3ci	1 mark for benefit, 1 mark for drawback Benefit: • The program can/might run faster • Cannot run out of stack space/memory • Easier to trace/follow Drawback: • Iteration can lead to lengthier code • Iteration can lead to code that looks more complex / is harder to understand • some problems are more elegantly coded with a recursive solution	2 **AO1.1 (1)**	

3cii	1 mark for each correct statement
	
	`function newGCD(num1, num2)`
	  `temp = 0`
	  `while (num2 != `**`0`**`)`
	    `temp`` = num2`
	    `num2 = num1 MOD `**`num2`**
	    `num1 = temp`
	  `endwhile`
	  `return `**`num1`**
	`endfunction`
		4
		
		AO2.2 (2)
		AO3.2 (2)	

4a	1 mark per bullet • She can split the problem down into sub problems • It will creates a more manageable problem / simpler to understand / maintain • can tackle each sub problem independently	2 **AO1.2 (1)** **AO2.2 (1)**	
4bi	1 mark per bullet, max 2 per sub-procedure e.g. • Select character (name, gender) • Gives the user options for choosing a character • Choose level • Give the user the choice of level (easy, normal, challenging) and take the user input • Touch enemy • Called to determine if the character touches an enemy • Lose life • Remove a life, if <0 then game over • End level • Move onto next level One mark for identifying sensible subroutine, 1 mark for description	6 **AO2.1 (2)** **AO2.2 (2)** **AO3.2 (2)**	Do not award any user *input* related procedures e.g. Left/Right input (but character movement *output* on screen left/right would be valid) Allow other reasonable responses from the scenario e.g. generate enemy()
4bii	1 mark per bullet • Decision based on what the user has input • E.g. If they click left move the character left // if they click right move the character right // if they click space bar make the character jump	2 **AO2.1 (1)** **AO2.2 (1)**	

<table>
<tr>
<td>4biii</td>
<td>1 mark per bullet<ul><li>The result from one process / procedure feeds into the next</li><li>E.g. the result of detecting a character touching an enemy feeds into reducing the number of lives</li></ul></td>
<td>2
<b>AO1.2 (1)</b>
<b>AO2.2 (1)</b></td>
<td>Note: 1 Mark Max for a generic description of pipelining</td>
</tr>
<tr>
<td>4c</td>
<td>1 mark for final solution, max 5 for showing the stages<ul><li>Mark A as the current node initially</li><li>Record B = 1, C = 2 (mark A as visited)</li><li>Record E = 5 (and mark B as visited)</li><li>(Record D = 3, F = 5 (and mark B as visited)</li><li>Change E to 4 (overriding previous value, and mark D as visited)</li><li>Record G = 6 (and mark E as visited)</li><li>…Do not change G as greater than current (mark F as visited)</li><li>(G as visited) H = 10 (Mark G as visited)</li><li>Solution: A-C-D-E-G-H path length 10</li></ul>
<table>
<tr><th>Node</th><th>Visited</th><th>From A</th><th>Previous Node</th><th></th></tr>
<tr><td>A</td><td>✓</td><td>0</td><td>-</td><td>1 Mark</td></tr>
<tr><td>B</td><td>✓</td><td>1</td><td>A</td><td rowspan="2">1 Mark</td></tr>
<tr><td>C</td><td>✓</td><td>2</td><td>A</td></tr>
<tr><td>D</td><td>✓</td><td>3</td><td>C</td><td rowspan="2">1 Mark</td></tr>
<tr><td>F</td><td>✓</td><td>5</td><td>C</td></tr>
<tr><td>E</td><td>✓</td><td>~~5~~ 4</td><td>~~B~~ D</td><td>2 Marks
Initial visit, plus override values</td></tr>
<tr><td>G</td><td>✓</td><td>6</td><td>E</td><td>1 Mark</td></tr>
<tr><td>H</td><td></td><td>10</td><td>G</td><td>1 Mark</td></tr>
</table>
</td>
<td>6
<b>AO1.2 (1)</b>
<b>AO2.2 (3)</b>
<b>AO2.2 (2)</b></td>
<td>Guidance – 1 mark only for stating the solution of A-C-D-E-G-H length 10</td>
</tr>
</table>

4d	**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of IDEs; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up the context which results in a supported and realistic judgment as to whether IDEs are useful in this context. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knoledge and understanding of IDEs; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether IDEs are useful in this context. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of IDEs with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** Tools to aid writing • Coloured font • Predictive text • Auto-correct Tools to aid de-bugging • Stepping • Break points • Variable watch window **AO2: Application** e.g. • Can write subroutines for the program and it will tell you what parameters are needed • Allow you to run the program without exiting the software / having to load a separate compiler • Integrates other tools such as version control. • Can reduce spelling errors • Can use to fix errors that might occur / debug **AO3: Evaluation** e.g. • User friendly for novices • Increase speed of writing • Fewer mistakes • Increase speed of testing / finding errors • Collaborative team working facilitated

5a	1 mark per bullet to max 6. Max 4 if generic description given with no application Max 4 if a diagrammatic solution is given with no description • Splits the list in half repeatedly… • … until it is in independent arrays / elements e.g. 2, 18, 6, 4, 12, 3 • Compare the first two items (index 0 and 1) e.g. 2, 18 • … and combine to create a new array in descending order i.e. 18, 2 • Repeat with indexes 2 and 3 (6, 4), then 4 and 5 (12, 3) • Compare the first element in the first two new arrays • …Choose the largest element, writing this to the new array first • …repeat until no elements left • Combine the two remaining lists into one list e.g. [2, 18, 6, 4, 12, 3] [2, 18, 6] [4, 12, 3] [2, 18] [6] [4, 12] [3] [2] [18] [6] [4] [12] [3] [18, 2] [6, 4] [12, 3] [18, 6, 4, 2] [12, 3] [18, 12, 6, 4, 3, 2] e.g. [2, 18, 6, 4, 12, 3] [2, 18, 6] [4, 12, 3] [2, 18] [6] [4, 12] [3] [2] [18] [6] [4] [12] [3] [18 2] [6] [12 4] [3] [18 6 2] [12 4 3] [18, 12, 6, 4, 3, 2]	6 **AO1.2 (3)** **AO2.2 (3)**	Allow max 5 if correct description but in ascending order.
5b	1 mark per bullet • Merge sort might create a new array each time it splits and merges / often implemented recursively which places additional data on the stack • Insertion sort does not use any additional arrays//Insertion sort is an in-place algorithm.	2 **AO1.2 (2)**	

6	**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of data mining; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up the context which results in a supported and realistic judgment as to whether it is possible to use data mining in this context. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of data mining; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether it is possible to use data mining. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of data mining with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** • Extracting data from databases • Using large data sets • Looking for patterns/specific occurrences of data • Gathering data that can be analysed and used to inform decisions **AO2: Application** e.g. • Use to find out what his users do • Find features that are used most often • Find features that are not used • Find out what people in his target age group do on other sites • Find out characteristics of people who use the site **AO3: Evaluation** e.g. • Can identify areas to focus attention • Save time and money by identifying areas that are not popular/used • New features targeted at specific groups could bring in new business e.g. advertising • But care would need to be applied to privacy issues / GDPR and potential impact on the users

7ai	1 mark per example e.g. • No actual images shown • Items are named / labelled • Simplified layout with shapes	2 **AO2.1 (2)**	Allow any reasonable examples, but they must be for different aspects
7aii	1 mark per bullet to max 3 e.g. • Reduces complexity of design • Reduces complexity of programming • Reduce memory/processing requirements • Could involve a large number of images that would take excessive memory • Reality contains things that aren't relevant to a computer program	3 **AO1.1 (1)** **AO1.2 (1)** **AO2.1 (1)**	Note: do not allow answers related to the user experience / user interpretation, the question is about the production of the system
7aiii	1 mark per example e.g. • Garden dimensions/width/length • Number of items in the garden • Name of items in the garden • Location of items in the garden	3 **AO2.1 (3)**	

<table>
<tr>
<td>7bi</td>
<td>1 mark per bullet to max 4

- Class declaration
- 3 attributes declared
- Constructor
- …taking parameters
- …setting the attributes to the parameters

<pre>
e.g.
class GardenItem
  private itemName
  private length
  private width
  public procedure new(pItemName, pLength, pWidth)
    itemName = pItemName
    length = pLength
    width = pWidth
  endprocedure
endclass
</pre>
</td>
<td>4

<b>AO1.1
(1)
AO2.1
(1)
AO2.2
(1)
AO3.2
(1)</b></td>
<td>Note that example answers are given in the specification pseudocode. Any pseudocode answer that can be understood by a ‘competent programmer’ should be accepted</td>
</tr>
</table>

7bii	1 mark per bullet
	• Class declaration inheriting from `gardenItem`
	• Additional 3 properties declared as private
	• Constructor takes all 5 parameters
	• Use of super (or equivalent) to set super class parameters
	• Remainder of properties set to parameters
	
	e.g.
	`class Tree inherits GardenItem`
	`  private height`
	`  private sun`
	`  private shade`
	`  public procedure new(pName, pHeight, pLenWidth, pSun, pShade)`
	`    super.itemName = pName`
	`    super.length = pLenWidth`
	`    super.width = pLenWidth`
	`    height = pHeight`
	`    sun = pSun`
	`    shade = pShade`
	`  endprocedure`
	`endclass`
		5
		
		AO1.1 (1)
		AO2.2 (1)
		AO3.2 (3)
			Accept solutions that call the parent's constructor.
			
			`class Tree inherits GardenItem`
			`  private height`
			`  private sun`
			`  private shade`
			`  public procedure new(pName, pHeight, pLenWidth, pSun, pShade)`
			`    height = pHeight`
			`    sun = pSun`
			`    shade = pShade`
			
			`super.new(pName,pLenWidth,pLenWidth)`
			`  endprocedure`
			`endclass`

7biii	1 mark per bullet • Declaration of instance of tree (i.e. `new Tree`) • Storing result in `firstTree` • All parameters included and in the same order as 7bii • …with appropriate data types e.g. `firstTree = new Tree("Common Oak",40,40,true, true)`	4 **AO1.1 (1)** **AO2.2 (1)** **AO3.2 (2)**	

7biv	1 mark per bullet
	• Get method declaration
	• Returns `itemName`
	• Set method declaration
	• …takes value as a parameter
	• …assigns parameter to `itemName`
	e.g.
	`function getItemName()`
	`  return itemName`
	`endfunction`
	
	`procedure setItemName(newname)`
	`  itemName = newname`
	`endprocedure`
		4
		
		AO1.2 (2)
		AO2.2 (2)	

<table>
<tr>
<td>7bv</td>
<td>

1 mark per bullet to max 6

- Procedure declaration with all four parameters
- Looping 1000 times / to end of array
- Checking if height and width are less than or equal to the maximum height and width
- Checking if sun and shade match
- Outputting value(s) using get methods
- Output an appropriate message
- Outputs a message if no matching tree is found

e.g.

```
procedure findTree(pHeight, pWidth, pSun, pShade)
  flag = false
  for i = 0 to 999
    if treeArray[i].getHeight() <= pHeight then
      if treeArray[i].getWidth() <= pWidth then
        if treeArray[i].getSun() == pSun then
          if treeArray[i].getShade() == pShade then
             flag = true
             print(treeArray[i].getItemName() + "
height: " + treeArray[i].getHeight + " width: " +
treeArray[i].getWidth() + " Sun?: " +
treeArray[i].getSun() + " Shade?: " +
treeArray[i].getShade()
          endif
        endif
      endif
    endif
  next count
  if flag == false then
    print("No suitable trees")
  endif
endprocedure
```

</td>
<td>6

<b>AO1.2
(1)
AO2.2
(2)
AO3.2
(3)</b></td>
<td></td>
</tr>
</table>

7c	**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of caching and reuseable components; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up the use of both caching and reusable components which results in a supported and realistic judgment as to whether it is possible to use them in this context. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knoledge and understanding of caching and reuseable components; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether it is possible to use caching and reusable componments in this context. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of caching and reuseable components with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** Caching: • Data that has been used is stored in cache/RAM in case it is needed again • Allows faster access for future use Reusable components • One piece of code can be used in multiple places / called many times • Use of subroutines / procedures / functions • Use of classes • Use of external libraries **AO2: Application** • Store items in cache • Store requirements in cache • Store garden layout in cache • Reuse shapes / designs • The use of a class allows replication **AO3: Evaluation** e.g. • Faster development • Faster/easier future adaptation • Better performance of program • Takes more time to plan/design to make use of both

OCR (Oxford Cambridge and RSA Examinations)
The Triangle Building
Shaftesbury Road
Cambridge
CB2 8EA

OCR Customer Contact Centre

Education and Learning
Telephone: 01223 553998
Facsimile: 01223 552627
Email: general.qualifications@ocr.org.uk

www.ocr.org.uk

For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored

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 (Oxford Cambridge and RSA Examinations)
Head office
Telephone: 01223 552552
Facsimile: 01223 552553

© OCR 2019

OCR
Oxford Cambridge and RSA

A Level Computer Science

H446/02 Algorithms and programming

Friday 15 June 2018 – Morning

Time allowed: 2 hours 30 minutes

You may use:

- a ruler (cm/mm)
- an HB pencil

Do not use:

- a calculator

First name										
Last name										
Centre number						Candidate number				

INSTRUCTIONS

- Use black ink.
- Complete the boxes above 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).
- Do **not** write in the barcodes.

INFORMATION

- The total mark for this paper is **140**.
- The marks for each question are shown in brackets **[ ]**.
- Quality of extended responses will be assessed in questions marked with an asterisk (*).
- This document consists of **28** pages.

© OCR 2018 [601/4911/5]

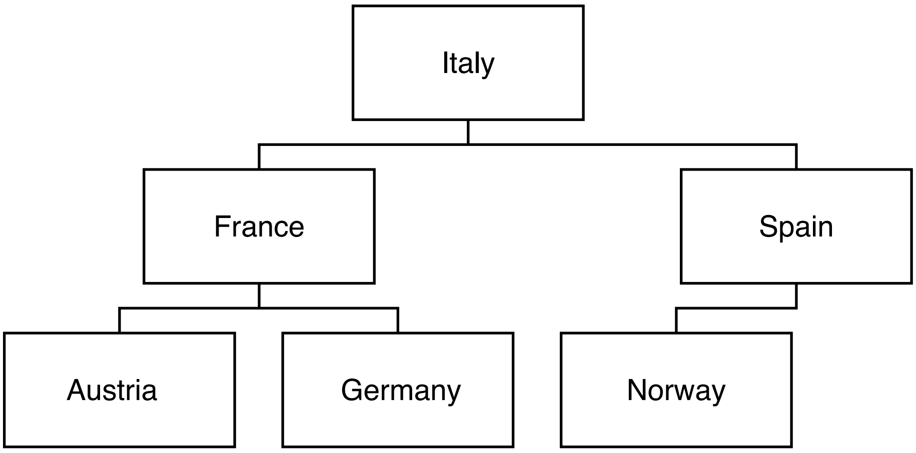
OCR is an exempt Charity

Section A

Answer **all** the questions.

1 A program stores entered data in a binary search tree.

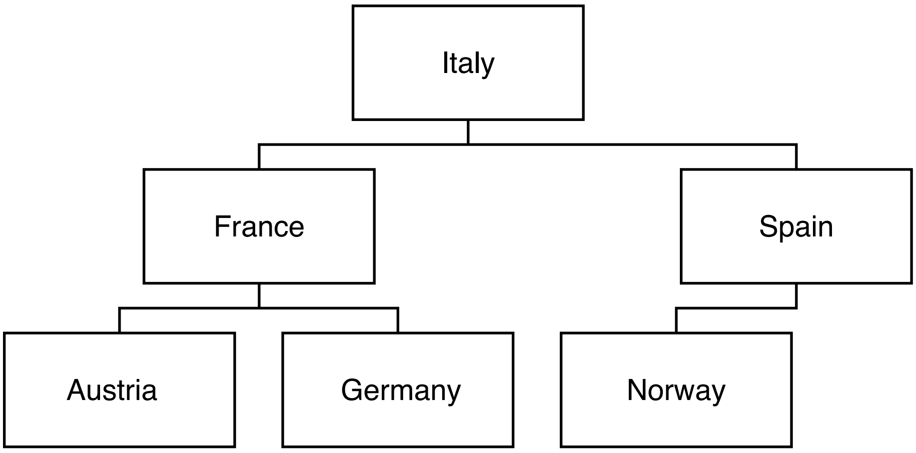
The current contents of the tree are shown:



(a) Complete the diagram to show the contents of the tree after the following data is added:

England, Scotland, Wales, Australia **[3]**

(b) Show the order of the nodes visited in a breadth first traversal on the following tree.



....

....

....

....

....

....

....

....**[3]**

© OCR 2018

(c) A pseudocode algorithm is written to search the tree to determine if the data item "Sweden" is in the tree.

The function `currentNode.left()` returns the node positioned to the left of `currentNode`.

The function `currentNode.right()` returns the node positioned to the right of `currentNode`.

```
function searchForData(currentNode:byVal, searchValue:byVal)

      thisNode = getData(..........................................................)

      if thisNode == ........................................................ then

           return ........................................................

      elseif thisNode < searchValue then

           if currentNode.left() != null then

                return (searchForData(currentNode.left(), searchValue))

           else

                return ........................................................

           endif

      else

           if ........................................................  != null then

               return (searchForData(currentNode.right(), searchValue))

           else

                return false

           endif

      endif

endfunction
```

(i) Complete the algorithm.

[5]

(ii) The algorithm needs to be used in different scenarios, with a range of different trees.

Identify **two** preconditions needed of a tree for this algorithm to work.

1 ....

2 ....

[2]

BLANK PAGE

PLEASE DO NOT WRITE ON THIS PAGE

2 A company merger is joining five e-commerce retailers under one company, OCRRetail. Each retailer has a different sales system and OCRRetail wants to develop one computer system that can be used by all the retailers.

Mary's software development company has been employed to analyse and design a solution for the company.

(a) **(i)** Two computational methods (techniques used to solve a problem using computational thinking) that Mary will use are problem recognition and decomposition.

State what is meant by problem recognition and decomposition.

Recognition ....

....

Decomposition ....

.... **[2]**

(ii) State **one** additional computational method.

.... **[1]**

(b) Mary plans to use data mining to generate information about OCRRetail's customers. Mary will use this information to benefit the company.

(i) Define the term 'data mining'.

....

.... **[1]**

(ii) Identify **two** pieces of information that data mining could provide OCRRetail about sales, and state how OCRRetail could make use of this information.

1 ....

....

....

....

2 ....

....

....

.... **[4]**

(c) Mary has developed the program and is considering using performance modelling before installing the system.

(i) Define the term ‘performance modelling’.

....

.... **[1]**

(ii) Identify **one** way performance modelling could be used to test the new system.

....

.... **[1]**

(d) Mary created the program as a series of sub-programs that can be reused.

Describe **one** benefit of Mary creating reusable program components.

....

....

....

.... **[2]**

© OCR 2018

3 A puzzle has multiple ways of reaching the end solution. Fig. 3 shows a graph that represents all possible routes to the solution. The starting point of the game is represented by A, the solution is represented by J. The other points in the graph are possible intermediary stages.

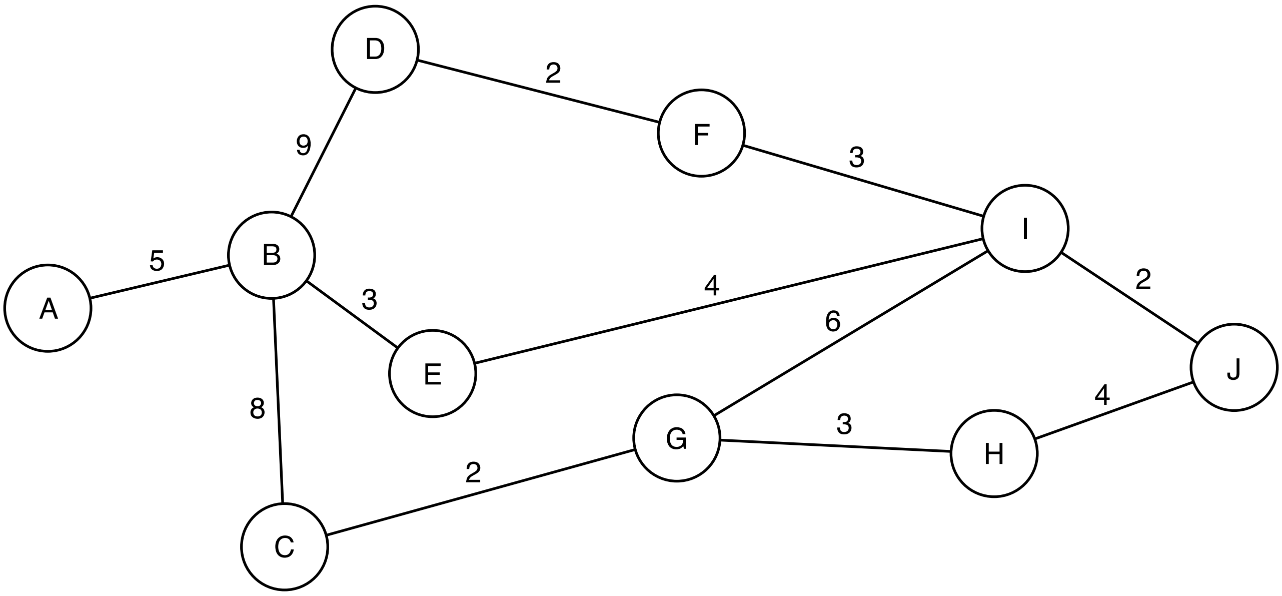


Fig. 3

(a) The graph in Fig. 3 is a visualisation of the problem.

(i) Identify **one** difference between a graph and a tree.

....

.... **[1]**

(ii) Explain how the graph is an abstraction of the problem.

....

....

....

.... **[2]**

(iii) Identify **two** advantages of using a visualisation such as the one shown in Fig. 3.

1 ....

....

2 ....

.... **[2]**

(b) Demonstrate how Dijkstra's algorithm would find the shortest path to the solution in Fig. 3.

....

....

....

....

....

....

....

....

....

....

....

.... **[7]**

(c)* The creator of the puzzle has been told that the A* algorithm is more efficient at finding the shortest path because it uses heuristics.

Compare the performance of Dijkstra's algorithm and the A* search algorithm, making reference to heuristics, to find the shortest path to the problem.

.... **[9]**

(d) A computer program version of the puzzle is to be developed. A programmer will use an IDE to debug the program during development.

Describe **three** features of an IDE that help debug the program.

1...

....

....

....

2...

....

....

....

3...

....

....

....

[6]

© OCR 2018

4 A recursive function, `generate`, is shown.

```
function generate(num1:byval)
    if num1 > 10 then
        return 10
    else
        return num1 + (generate(num1 + 1) DIV 2)
    endif
endfunction
```

(a) Trace the algorithm to show the value returned when `generate(7)` is called. Show each step of your working.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

(b) The parameter, `num1`, is passed by value.

Explain why the parameter was passed by value instead of by reference.

....

....

....

.... **[2]**

(c)* Parameters can be used to reduce the use of global variables.

Compare the use of parameters to global variables in recursive functions.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[9]**

© OCR 2018

(d) A student called Jason writes a recursive algorithm. The recursive algorithm uses more memory than if Jason had written it as an iterative algorithm.

Explain why the recursive algorithm uses more memory than the iterative algorithm.

....

....

....

.... **[2]**

5 A computer program stores data input on a stack named `dataItems`. The stack has two sub-programs to add and remove data items from the stack. The stack is implemented as a 1D array, `dataArray`.

Sub-program	**Description**
`push()`	The parameter is added to the top of the stack
`pop()`	The element at the top of the stack is removed

The current contents of `dataItems` are shown:

6
15
100
23

(a) Show the contents of the stack `dataItems` after each line of the following lines of code are run

```
01  push(13)
02  pop()
03  push(10)
04  push(20)
```

Line 01	Line 02	Line 03	Line 04
6			
15			
100			
23			

[4]

© OCR 2018

(b) The main program asks a user to push or pop an item from the stack. If the user chooses ‘push’, the data item is added to the stack. If the user chooses “pop”, the next item is removed from the stack, multiplied by 3 and output.

The main program is shown:

```
01 userAnswer = input("Would you like to push or pop an item?")
02 if userAnswer == "push" then
03     push(input("Enter data item"))
04 else
05     print(pop() * 3)
06 endif
```

(i) Before the sub-programs, `push()` and `pop()`, can add or remove items from the stack, a selection statement is used to decide if each action is possible.

Describe the decision that needs to be made in each sub-program and how this impacts the next process.

`push()` ....

....

....

....

`pop()` ....

....

....

....

[4]

(ii) The algorithm does not work when the user enters `"PUSH"` or `"Push"`. The algorithm needs to be changed in order to accept these inputs.

Identify the line number to be changed and state the change that should be made.

Line number ....

Change If userAnswer == “push” OR “PUSH” OR “Push” ....

....

[2]

(c) The stack is implemented as a 1D array, `dataArray`.

Describe how a 1D array can be set up and used to push and pop items as a stack.

....

....

....

....

....

.... **[3]**

(d) As an array, the data in `dataArray` is sorted and then searched for a specific value.

(i) The data in `dataArray` is sorted into ascending order using an insertion sort.

The current contents of `dataArray` are shown:

100	22	5	36	999	12

Show the steps of an insertion sort on the current contents of the array `dataArray`.

....

....

....

....

....

....

....

....

....

....

....

....

.... **[5]**

(ii) The array `dataArray` can now be searched using a binary search.

Describe the stages of a binary search on an array of size `n`.

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[7]**

(iii) The array has 50 items.

The function, `searchItem()`, performs a linear search for a data item.

```
function searchItem(dataItem)
    for count = 0 to 49
        if dataArray[count] == dataItem then
            return(count)
        endif
    next count
    return(-1)
endfunction
```

Rewrite the function using a while loop.

....

....

....

....

....

....

....

....

....

....

....

.... **[4]**

Section B

Answer **all** questions.

6 Kamran is writing a program to manipulate the data for a set of items.

For each item, the program needs to store:

- Item name (e.g. Box)
- Cost (e.g. 22.58)
- Date of arrival (e.g. 1/5/2018)
- Transferred (e.g. true)

The items are added to a queue for processing.

The queue is defined as a class, `itemQueue`.

`itemQueue`

`theItems[10] : Items`
`head : Integer`
`tail : Integer`
`numItems : Integer`

`constructor`
`enqueuer()`
`dequeuer()`
`setnumItems()`
`getnumItems()`

The `head` attribute points to the first element in the queue. The `tail` attribute points to the next available space in the queue. The `numItems` attribute states how many items are currently in the queue.

(a) The data about the items can be stored using either a record structure, or as objects of a class.

(i) Explain the similarities and differences between a record and a class.

....

....

....

....

....

.... **[3]**

(ii) Kamran chooses to use a record structure to store the data about the items.

Record structures may be declared using the following syntax:

```
recordStructure recordstructurename
    fieldname : datatype
    …
endRecordStructure
```

Complete the pseudocode to declare a record called `items`.

```
recordStructure ................................................................................

    itemName : .....................................................................

    ......................................................................: Currency

    ......................................................................: Date

    transferred : .....................................................................

endRecordStructure
```

[5]

(iii) New records may be created using the following syntax:

```
recordidentifier : recordstructurename
recordidentifier.fieldname = data
…
```

Write a programming statement to create a new item, using the identifier 'box1', with the item name "Box", the cost 22.58, date of arrival 1/5/2018 and transferred true.

....

....

....

....

....

....

....

.... **[3]**

(b) The array, `theItems`, stores the items in the queue. When the tail of the queue exceeds the last element in the array, it adds a new item to the first element if it is vacant.

For example, in the following queue, the next item to be added would be placed at index 0.

Index	0	1	2	3	4	5	6	7	8	9
Element				Data	Data	Data	Data	Data	Data	Data

(i) Define the term 'queue'.

....

....

....

.... **[2]**

(ii) The attributes in `itemQueue` are all declared as private.

Explain how a private attribute improves the integrity of the data.

....

....

....

.... **[2]**

(iii) The constructor method creates a new instance of `itemQueue` and sets the `head`, `tail` and `numItems` attributes to 0.

Write an algorithm, using pseudocode or program code, for the constructor including the initialisation for all attributes.

....

....

....

....

....

.... **[2]**

(iv) The `enqueue` method:

- takes as a parameter the item to insert in the queue
- checks if the queue is full
- reports an error and returns `false` if the queue is full
- does the following if the queue is not full:
    - o adds the item to the array at the tail position and adjusts the pointer(s)
    - o returns `true`

The attribute `numItems` stores the number of items currently in the queue.

Write an algorithm, using pseudocode or program code, for the `enqueue` method.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

(v) Write a programming statement to declare an instance of `itemQueue` called `myItems`.

....

.... **[1]**

(vi) Write a procedure, `insertItems()`, to ask the user to input the data for an item. The item is then added to the queue `myItems`. The user is continually asked to input data items until the queue is full.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[5]**

(vii) When the main program ends, the items and the queue no longer exist.

Describe how Kamran could amend the program to make sure the items and queue still exist and are used the next time the program is run.

....

....

....

....

....

.... **[2]**

(c)* Kamran wants to expand the program to allow it to handle up to 100,000,000 items and to allow him to search for data about items. Kamran is worried that the increase in the number of items will cause a decrease in the performance of the program. He decides to investigate the benefits of caching and concurrent processing.

Evaluate the use of caching and concurrent processing in this scenario and make a recommendation to Kamran.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[9]**

END OF QUESTION PAPER

PLEASE DO NOT WRITE ON THIS PAGE

Oxford Cambridge and RSA

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) 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.

OCR
Oxford Cambridge and RSA

GCE

Computer Science

Unit **H446/02:** Algorithms and programming

Advanced GCE

Mark Scheme for June 2018

OCR (Oxford Cambridge and RSA) is a leading UK awarding body, providing a wide range of qualifications to meet the needs of candidates of all ages and abilities. OCR qualifications include AS/A Levels, Diplomas, GCSEs, Cambridge Nationals, Cambridge Technicals, Functional Skills, Key Skills, Entry Level qualifications, NVQs and vocational qualifications in areas such as IT, business, languages, teaching/training, administration and secretarial skills.

It is also responsible for developing new specifications to meet national requirements and the needs of students and teachers. OCR is a not-for-profit organisation; any surplus made is invested back into the establishment to help towards the development of qualifications and support, which keep pace with the changing needs of today’s society.

This mark scheme is published as an aid to teachers and students, to indicate the requirements of the examination. It shows the basis on which marks were awarded by examiners. It does not indicate the details of the discussions which took place at an examiners’ meeting before marking commenced.

All examiners are instructed that alternative correct answers and unexpected approaches in candidates’ scripts must be given marks that fairly reflect the relevant knowledge and skills demonstrated.

Mark schemes should be read in conjunction with the published question papers and the report on the examination.

© OCR 2018

Annotations

Annotation	Meaning
^	Omission mark
BOD	Benefit of the doubt
✗	Incorrect point
FT	Follow through
NAQ	Not answered question
NBOD	No benefit of doubt given
REP	Repeat
✓	Correct point
TV	Too vague
BP	Blank Page – this annotation must be used on all blank pages within an answer booklet (structured or unstructured) and on each page of
L1	Level 1
L2	Level 2
L3	Level 3

Subject-specific Marking Instructions

INTRODUCTION

Your first task as an Examiner is to become thoroughly familiar with the material on which the examination depends. This material includes:

- the specification, especially the assessment objectives
- the question paper and its rubrics
- the mark scheme.

You should ensure that you have copies of these materials.

You should ensure also that you are familiar with the administrative procedures related to the marking process. These are set out in the OCR booklet **Instructions for Examiners**. If you are examining for the first time, please read carefully **Appendix 5 Introduction to Script Marking: Notes for New Examiners**.

Please ask for help or guidance whenever you need it. Your first point of contact is your Team Leader.

USING THE MARK SCHEME

Please study this Mark Scheme carefully. The Mark Scheme is an integral part of the process that begins with the setting of the question paper and ends with the awarding of grades. Question papers and Mark Schemes are developed in association with each other so that issues of differentiation and positive achievement can be addressed from the very start.

This Mark Scheme is a working document; it is not exhaustive; it does not provide ‘correct’ answers. The Mark Scheme can only provide ‘best guesses’ about how the question will work out, and it is subject to revision after we have looked at a wide range of scripts.

The Examiners’ Standardisation Meeting will ensure that the Mark Scheme covers the range of candidates’ responses to the questions, and that all Examiners understand and apply the Mark Scheme in the same way. The Mark Scheme will be discussed and amended at the meeting, and administrative procedures will be confirmed. Co-ordination scripts will be issued at the meeting to exemplify aspects of candidates’ responses and achievements; the co-ordination scripts then become part of this Mark Scheme.

Before the Standardisation Meeting, you should read and mark in pencil a number of scripts, in order to gain an impression of the range of responses and achievement that may be expected.

In your marking, you will encounter valid responses which are not covered by the Mark Scheme: these responses must be credited. You will encounter answers which fall outside the ‘target range’ of Bands for the paper which you are marking. Please mark these answers according to the marking criteria.

Please read carefully all the scripts in your allocation and make every effort to look positively for achievement throughout the ability range. Always be prepared to use the full range of marks.

LEVELS OF RESPONSE QUESTIONS:

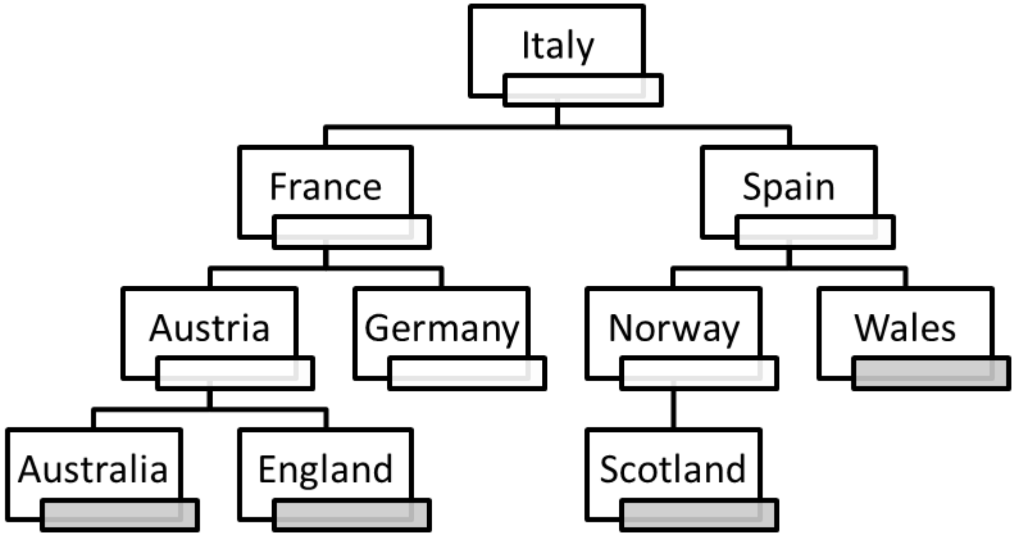
The indicative content indicates the expected parameters for candidates' answers, but be prepared to recognise and credit unexpected approaches where they show relevance.
Using 'best-fit', decide first which set of BAND DESCRIPTORS best describes the overall quality of the answer. Once the band is located, adjust the mark concentrating on features of the answer which make it stronger or weaker following the guidelines for refinement.

- **Highest mark:** If clear evidence of all the qualities in the band descriptors is shown, the HIGHEST Mark should be awarded.
- **Lowest mark:** If the answer shows the candidate to be borderline (i.e. they have achieved all the qualities of the bands below and show limited evidence of meeting the criteria of the band in question) the LOWEST mark should be awarded.
- **Middle mark:** This mark should be used for candidates who are secure in the band. They are not 'borderline' but they have only achieved some of the qualities in the band descriptors.

Be prepared to use the full range of marks. Do not reserve (e.g.) high Band 3 marks 'in case' something turns up of a quality you have not yet seen. If an answer gives clear evidence of the qualities described in the band descriptors, reward appropriately.

	AO1	AO2	AO3
High (thorough)	Precision in the use of question terminology. Knowledge shown is consistent and well-developed. Clear appreciation of the question from a range of different perspectives making extensive use of acquired knowledge and understanding.	Knowledge and understanding shown is consistently applied to context enabling a logical and sustained argument to develop. Examples used enhance rather than detract from response.	Concerted effort is made to consider all aspects of a system / problem or weigh up both sides to an argument before forming an overall conclusion. Judgements made are based on appropriate and concise arguments that have been developed in response resulting in them being both supported and realistic.
Middle (reasonable)	Awareness of the meaning of the terms in the question. Knowledge is sound and effectively demonstrated. Demands of question understood although at times opportunities to make use of acquired knowledge and understanding not always taken.	Knowledge and understanding applied to context. Whilst clear evidence that an argument builds and develops through response there are times when opportunities are missed to use an example or relate an aspect of knowledge or understanding to the context provided.	There is a reasonable attempt to reach a conclusion considering aspects of a system / problem or weighing up both sides of an argument. However the impact of the conclusion is often lessened by a lack of supported judgements which accompany it. This inability to build on and develop lines of argument as developed in the response can detract from the overall quality of the response.
Low (basic)	Confusion and inability to deconstruct terminology as used in the question. Knowledge partial and superficial. Focus on question narrow and often one-dimensional.	Inability to apply knowledge and understanding in any sustained way to context resulting in tenuous and unsupported statements being made. Examples if used are for the most part irrelevant and unsubstantiated.	Little or no attempt to prioritise or weigh up factors during course of answer. Conclusion is often dislocated from response and any judgements lack substance due in part to the basic level of argument that has been demonstrated throughout response.

	Assessment Objective
AO1	Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.
AO1.1	Demonstrate **knowledge** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO1.2	Demonstrate **understanding** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO2	Apply knowledge and understanding of the principles and concepts of computer science including to analyse problems in computational terms.
AO2.1	Apply knowledge and understanding of the principles and concepts of computer science.
AO2.2	Analyse problems in computational terms.
AO3	Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.
AO3.1	Design computer systems that solve problems.
AO3.2	Program computer systems that solve problems.
AO3.3	Evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

Question			Answer	Marks	Guidance
1	(a)		 Italy France Spain Austria Germany Norway Wales Australia England Scotland 1 mark for each of: - Scotland in correct place - Wales in correct place - Australia and England both in correct place	**3** **AO2.2 (3)**	
1	(b)		1 mark per bullet to max • Italy • France, Spain • Austria, Germany, Norway	**3** **AO1.1 (1)** **AO2.1 (1)** **AO2.2 (1)**	

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>1</td><td>(c)</td><td>(i)</td><td>1 mark per bullet to max 5

```
function searchForData(currentNode:byVal, searchValue:byVal)
      thisNode = getData(currentNode)
      if thisNode == searchValue then
         return true
      elseif thisNode < searchValue then
         if currentNode.left () != null then
             return (searchForData(currentNode.left (),
searchValue))
         else
             return false
         endif
     else
         if currentNode.right() != null then
         return (searchForData(currentNode.right (),
searchValue))
         else
             return false
         endif
     endif
endfunction
```
</td><td>5
AO2.2 (2)
AO3.2 (3)</td><td>The line <b><code>elseif thisNode < searchValue then</code></b> should have read <b><code>elseif thisNode > searchValue then</code></b>

If candidates attempt to correct the code and their answers are consistent with, and work with their amendment, such answers should be credited.</td></tr>
<tr><td>1</td><td>(c)</td><td>(ii)</td><td>• It's a binary tree
• It's ordered / sorted</td><td>2
AO2.2 (2)</td><td></td></tr>
</table>

Question			Answer	Marks	Guidance
2	(a)	(i)	Recognition • Identify there is a problem to be solved // what the problem is Decomposition • Splitting down a problem into sub-problems	**2** **AO1.1** **(2)**	
2	(a)	(ii)	e.g. • Divide and conquer • Abstraction	**1** **AO1.1** **(1)**	Accept other credible answers e.g.: Critical thinking, Modelling, Heuristics, Concurrency, Visualisation, Backtracking
2	(b)	(i)	• Turning large quantities of data into useful information / Finding patterns within large quantities of information	**1** **AO1.1 (1)**	Must refer to large quantities of data
2	(b)	(ii)	1 mark per identifying data, 1 for use e.g. • Identify customer trends • To identify items to sell/offers to send customers • Identify which stores are making the most profit • To identify what the other stores are doing well • Which items are not selling well • To replace them with other items	**4** **AO2.2** **(4)**	Accept any valid responses
2	(c)	(i)	Simulate/test the behaviour of the system before it is used	**1** **AO1.1 (1)**	
2	(c)	(ii)	e.g. • Testing it with a large number of simultaneous orders (stress testing) • Testing it with a large number of customers/items/orders	**1** **AO2.2 (1)**	
2	(d)		1 mark per bullet to max 2 e.g. • the components can be used in a future program… • they do not need to be rewritten / saves time	**2** **AO1.1 (1)** **AO2.1 (1)**	

Question			Answer	Marks	Guidance
			• they have already been tested… • …it will save time		
3	**(a)**	**(i)**	Any one from: - A graph has cycles - A graph can be directed/undirected - A tree has a hierarchy (e.g. Parent/Child)	**1** **AO1.2 (1)**	Allow any appropriate description e.g. graph can be weighted, tree has a root
3	**(a)**	**(ii)**	1 mark per bullet to max 2 • The puzzle is not shown in the diagram • The graph shows different sequences of sub problems in the puzzle that can be solved to get to the final solution • The puzzle does not have all states visible at once	**2** **AO1.2 (1)** **AO2.1 (1)**	Answers must be in context of the puzzle
3	**(a)**	**(iii)**	1 mark per bullet to max 2 e.g. • Visualisations benefit humans rather than computers • Visualisations present the information in a simpler form to understand • Visualisations can best explain complex situations	**2** **AO1.1 (1)** **AO2.1 (1)**	
3	**(b)**		1 mark per bullet • Mark A as the initial node and then visit B (5) • Node E (8) is then visited (chosen from C (13), D (14), E (8)) • Node I (12) is then visited after E • Node J (14) is then visited after I : • Visiting G (18) from I; • Visiting G (15) from C – overriding the previous value of 18 • solution A-B-E-I-J path length 14	**7** **AO1.2 (3)** **AO2.1 (2)** **AO2.2 (2)**	

Question			Answer	Marks	Guidance
3	(c)		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of Dijkstra's and A*; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of Dijkstra's and A*; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of Dijkstra's and A* with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and	**9** **AO1.1 (2)** **AO1.2 (2)** **AO2.1 (2)** **AO3.3 (3)**	**AO1: Knowledge and Understanding** **Indicative content** • Heuristic helps produce a solution in a faster time • A* uses estimated distance from final node • Dijkstra uses a weight/distance • A* chooses which path to take next based on lowest current distance travelled **AO2: Application** • Description of how A* will differ from Dijkstra, e.g. taking the shorter route A-B-E-I before exploring nodes from D and E • Description of the different number of comparisons that would be needed in this problem • A* doesn't need to find all possible solutions (saves time) **AO3: Evaluation** Candidates will need to evaluate the benefits and drawbacks of each algorithm • Small-scale problem • Quick to find a solution using either method • Difference in programming complexity is minimal • Don't know if this problem needs to scale • Most efficient route needed

Question			Answer	Marks	Guidance
			understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		
3	**(d)**		1 mark per bullet to max 4 e.g. • Underlines syntax errors dynamically • Can be corrected before running // saves times • Watch window • View how variables change during running of the program • Break points • Stop the program at set points to check the values of variables • Error message list • Tells you where errors are and suggests corrections • Step-mode • Executes program one statement at a time to watch variable values and program pathways • Traces • Print-outs of variable values for each statement	**6** **AO1.1 (3)** **AO1.2 (3)**	

Question			Answer	Marks	Guidance
			execution within a program • Crash-dump/post-mortem routine • Shows the state of variables where an error occurs • Stack contents • Shows sequencing through procedures/modules • Cross-referencers • Identifies where variables/constants are used in a program to avoid duplications		

4	**(a)**		1 mark per bullet for working to max 6
			• generate(7)
			return 7 + (generate(8) DIV 2)
			• generate(8)
			return 8 + (generate(9) DIV 2)
			generate(9)
			return 9 + (generate(10) DIV 2)
			generate(10)
			return 10 + (generate(11) DIV 2)
			• generate(11)
			return 10
			• Rewinding: return 10 + (10 DIV 2) = 10 + 5 = 15
			• return 9 + (15 DIV 2) = 9 + 7 = 16
			return 8 + (16 DIV 2) = 8 + 8 = 16
			• return 7 + (16 DIV 2) = 7 + 8 = 15
				6
				AO1.2 (1)
				AO2.2 (5)	

Question			Answer	Marks	Guidance
4	(b)		• If the value is sent by value, num1 will not be overridden / it is a copy of the parameter that is used (1) and this will produce the correct output (1) • if the parameter had been passed by reference it would not produce the correct result (1) as num1 would be overridden / because it is a pointer to the address of the variable (1)	**2** **AO2.1 (1)** **AO2.2 (1)**	
4	(c)		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of parameters and global variables; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of parameters and global variables; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and*	**9** **AO1.1 (2)** **AO1.2 (2)** **AO2.1 (2)** **AO3.3 (3)**	**AO1: Knowledge and Understanding** **Indicative content** • Parameter allows a value to be sent to a sub-program • Global variables can be accessed throughout the scope of the program • Local variables can only be accessed within the scope of the sub-program it's defined within – a parameter becomes a local variable in the function **AO2: Application** • If global, equivalent of by reference -value would be over-ridden • Global variable takes more memory than a local variable/parameter • In recursion, each call produces a new local variable for num1 **AO3: Evaluation** Candidates will need to evaluate the benefits and drawbacks of each algorithm • Global would require altering the algorithm as the value would be over-ridden on each call • Global would mean that memory space is kept throughout the running of the program, not just the sub-program • Parameter enables memory to be reallocated • Many more memory spaces needed for parameter

Question			Answer	Marks	Guidance
			supported by some evidence. **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of parameters and global variables with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		in recursion, 1 for each call

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>4</td><td>(d)</td><td></td><td>1 mark per bullet
• Each recursive call stores the current state on the stack // creates new variables
• Iteration reuses the same variables</td><td>2
AO1.2 (1)
AO2.1 (1)</td><td></td></tr>
<tr><td>5</td><td>(a)</td><td></td><td>1 mark for each correct stack
<table>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>20</td></tr>
<tr><td>13</td><td></td><td>10</td><td>10</td></tr>
<tr><td>6</td><td>6</td><td>6</td><td>6</td></tr>
<tr><td>15</td><td>15</td><td>15</td><td>15</td></tr>
<tr><td>100</td><td>100</td><td>100</td><td>100</td></tr>
<tr><td>23</td><td>23</td><td>23</td><td>23</td></tr>
</table></td><td><b>4
AO1.2 (2)
AO2.2 (2)</b></td><td></td></tr>
<tr><td>5</td><td>(b)</td><td>(i)</td><td>1 mark per bullet, max 2 for insert, max 2 for remove
push
• Check if the stack is full (pointer = array.length/array.length+1)
• If it is not – insert the item
• If it is – return/error that the stack is full

pop
• Check if the stack is empty (pointer = 0/1)
• If it is – return/error that the stack is empty
• If it is not – return the item</td><td>4
AO1.2 (2)
AO2.2 (2)</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>5</td><td>(b)</td><td>(ii)</td><td>1 mark per line, 1 for change
• line 02
• Include an OR with variations (e.g. <code>userAnswer = "PUSH" OR userAnswer = "Push"</code> etc.)/Convert input to uppercase/lowercase and just compare to equivalent</td><td>2
AO2.2 (2)</td><td></td></tr>
<tr><td>5</td><td>(c)</td><td></td><td>1 mark per bullet to max 3
• Array size defined
• A stack pointer is used to point to the top of the stack
• When an item is pushed the stack pointer is incremented
• When an item is popped the stack pointer is decremented</td><td>3
AO1.2 (1)
AO2.1 (1)
AO2.2 (1)</td><td></td></tr>
<tr><td>5</td><td>(d)</td><td>(i)</td><td>1 mark per row (after first row)
<table>
<tr><td>100</td><td>22</td><td>5</td><td>36</td><td>999</td><td>12</td><td></td></tr>
<tr><td>22</td><td>100</td><td>5</td><td>36</td><td>999</td><td>12</td><td>1 mark</td></tr>
<tr><td>5</td><td>22</td><td>100</td><td>36</td><td>999</td><td>12</td><td>1 mark</td></tr>
<tr><td>5</td><td>22</td><td>36</td><td>100</td><td>999</td><td>12</td><td>1 mark</td></tr>
<tr><td>5</td><td>22</td><td>36</td><td>100</td><td>999</td><td>12</td><td>1 mark</td></tr>
<tr><td>5</td><td>12</td><td>22</td><td>36</td><td>100</td><td>999</td><td>1 mark</td></tr>
</table></td><td><b>5
AO2.2 (5)</b></td><td></td></tr>
<tr><td>5</td><td>(d)</td><td>(ii)</td><td>1 mark per bullet to max 7
• Repeat
• Calculating an array midpoint…
• …by adding the array lower bound to the array upper bound, dividing by 2 and rounding
• Compare array midpoint with value to search for…
• …if equal set found flag to true
• …if array midpoint < value to search for, change lowerbound to equal midpoint + 1
• …if array midpoint > value to search for, change upperbound to equal midpoint – 1
• Until lowerbound is greater than or equal to upperbound
• Return/output found flag</td><td>7
AO1.1 (2)
AO1.2 (3)
AO2.1 (1)
AO2.2 (1)</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>5</td><td>(d)</td><td>(iii)</td><td>1 mark per bullet
<ul>
<li>Setting variable to start at 0</li>
<li>Suitable while structure (endwhile or clear indentation)</li>
<li>looping 50 times</li>
<li>Incrementing the variable within the loop</li>
</ul>
e.g. 1
<pre>
function searchItem(dataItem)
    count = 0
    while count < 50
        if dataArray(count) == dataItem then
            return(count)
        endif
        count = count + 1
    endwhile
    return(-1)
endfunction
</pre>
e.g. 2
<pre>
function searchItem(dataItem)
    count = 0
    while count < 50 and dataArray[count]!=dataItem
        count = count + 1
    endwhile
    if count==50
        count=-1
    endif
    return(count)
endfunction
</pre>
</td><td>4
AO1.2 (1)
AO3.1 (1)
AO3.2 (2)</td><td></td></tr>
</table>

Question			Answer	Marks	Guidance
6	**(a)**	**(i)**	1 mark per bullet to max 3 • Record is a data structure… • …A class is a template for making data structures (objects) • Class also has methods (which describes functionality) • Both store data of different types • Which can be accessed by their names • But classes can make them accessible via methods • Both can have multiple ‘instances’ • Class can include visibility of properties / private	3 AO1.2 (3)	

6	**(a)**	**(ii)**	1 mark per space
			
			`recordStructure` **`items`**
			    `itemName :` **`String`**
			    `cost` `: Currency`
			    `dateArrival` `: Date`
			    `transferred :` **`Boolean`**
			`endRecordStructure`
				5
				AO2.2 (2)
				AO3.2 (3)	

6	**(a)**	**(iii)**	1 mark per bullet to max 3
			• Declaring box1 as an item
			• Using Box1. (or equivalent) for each variable
			• Setting each variable (matching 6aii) correctly
			
			e.g.
			`Box1 : Items`
			`Box1.itemName = “Box”`
			`Box1.cost = 22.58`
			`Box1.dateArrival = “1/5/2018”`
			`Box1.transfered = True`
				3
				AO2.2 (2)
				AO3.2 (1)
					Ensure variable names for cost and dateArrival are consistent with variable names given in a(ii)

6	**(b)**	**(i)**	1 mark per bullet to max 2 • A data structure • FIFO (first in first out)	2 AO1.1 (2)	

Question			Answer	Marks	Guidance
6	**(b)**	**(ii)**	1 mark per bullet to max 2 • Properties (are encapsulated) and can only be accessed through their methods • Enforce validation through the method // inappropriate data can be caught before entered • Cannot be changed/accessed accidentally	2 AO1.2 (2)	

6	**(b)**	**(iii)**	1 mark per bullet to max
			• Constructor method/`new`
			• Setting `head` and `tail` to 0 within constructor method
			
			e.g.
			`public procedure new()`
			`  head = 0`
			`  tail = 0`
			`  numItems = 0`
			`endprocedure`
				2
				AO2.2 (1)
				AO3.2 (1)	

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>6</td><td>(b)</td><td>(iv)</td><td>1 mark per bullet to max 6
<ul>
<li>Function declaration, taking item as a parameter</li>
<li>Checking if the queue is full…</li>
<li>…outputting/reporting error and returning <code>false</code></li>
<li>Adding the item to the <code>tail</code> position</li>
<li>Correctly updating the <code>tail</code> pointer (either before or after addition)</li>
<li>Incrementing <code>numItems</code> and returning <code>true</code> if successful</li>
</ul>
e.g.
<pre>
public function enqueue(newItem : items) : boolean
  if numItems = 10 then
    print("Error: The queue is full")
    return false
  else
    theItems[tail] = newItem
    if tail = 9 then
      tail = 0
    else
      tail += 1
    endif
    numItems += 1
    return true
  endif
endprocedure
</pre></td><td>6
AO2.2 (3)
AO3.1 (1)
AO3.2 (2)</td><td></td></tr>
<tr><td>6</td><td>(b)</td><td>(v)</td><td>e.g.
<pre>
myItems = (new) itemQueue()
</pre></td><td>1
AO2.1 (1)</td><td>Allow follow through if they have parameters in 6(b)(iii)</td></tr>
</table>

Question			Answer	Marks	Guidance

6	(b)	(vi)	1 mark per bullet to max 5
			• Procedure declaration for `insertItems`
			• Asking for input of data items for a new item …..
			• …using record structure correctly
			• Use of `myItems.enqueue`
			• Looping while the queue is not full
			
			e.g.
			`procedure insertItems()`
			
			`  newItem : Items`
			`  itemCount = myItems.getnumItems()`
			
			`  while itemCount < 10`
			`    newItem.itemName = input("Enter the item name")`
			`    newItem.cost = input("Enter the item cost")`
			`    newItem.dateArrival = input("Enter the date of arrival")`
			`    newItem.transferred = input("Has it been transferred?")`
			`    myItems.enqueue(newItem)`
			`    itemCount = itemCount + 1`
			`  endwhile`
			
			`  myItems.setnumItems(itemCount)`
			
			`endprocedure`
				5
				AO2.2 (2)
				AO3.1 (1)
				AO3.2 (2)	

6	(b)	(vii)	1 mark per bullet to max 2 • Store the items and queue to an external file (when the program closes) • Load the items and queue from the file when it starts	2 AO2.1 (1) AO2.2 (1)	

Question			Answer	Marks	Guidance
6	(c)		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of caching and concurrent processing; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of caching and concurrent processing; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of caching and concurrent processing with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to	**9** **AO1.1 (2)** **AO1.2 (2)** **AO2.1 (2)** **AO3.3 (3)**	**AO1: Knowledge and Understanding** **Indicative content** Caching • Previously used data is stored in a location… • that can be quickly accessed … • to speed up retrieval if needed in future Concurrent Processing • several processes work simultaneously to solve a problem **AO2: Application** Caching • search for previously searched for data items in a faster secondary storage device/RAM • Speed up access for that item …Relies on same item being searched for multiple times …Kamran needs to decide how feasible this is based on the number of item Concurrent • Computer would have multiple processors… • Each searching part of the data structure at one time… • This would be limited by bottlenecks such as accessing the storage device • The n processors could potentially mean an increase of up to 1/n of time…realistically speed increase is likely to be less than that • Only useful if using linear search // binary search cannot be performed concurrently **AO3: Evaluation** Candidates will need to evaluate the benefits and drawbacks of caching and concurrent processing Allow any point of view (caching / concurrent / both) as long as argument is presented suitably.

Question			Answer	Marks	Guidance
			apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		

OCR (Oxford Cambridge and RSA Examinations)
The Triangle Building
Shaftesbury Road
Cambridge
CB2 8EA

OCR Customer Contact Centre

Education and Learning
Telephone: 01223 553998
Facsimile: 01223 552627
Email: general.qualifications@ocr.org.uk

www.ocr.org.uk

For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored

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 (Oxford Cambridge and RSA Examinations)
Head office
Telephone: 01223 552552
Facsimile: 01223 552553

© OCR 2018

OCR

Oxford Cambridge and RSA

A Level Computer Science

H446/02 Algorithms and Programming

Thursday 22 June 2017 – Morning

Time allowed: 2 hours 30 minutes

Do not use:

- A calculator

First name	
Last name	

Centre number						Candidate number				

INSTRUCTIONS

- Use black ink.
- Complete the boxes above 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).
- Do **not** write in the barcodes.

INFORMATION

- The total mark for this paper is **140**.
- The marks for each question are shown in brackets **[ ]**.
- Quality of extended responses will be assessed in questions marked with an asterisk (*).
- This document consists of **28** pages.

© OCR 2017 [601/4911/5]

OCR is an exempt Charity

Answer **all** the questions.

Section A

1 A programmer needs to sort an array of numeric data using an insertion sort.

(a) (i) The following, incomplete, algorithm performs an insertion sort.

Complete the algorithm.

```
procedure sortit(dataArray, lastIndex)
  for x = 1 to lastIndex
    currentData = dataArray[......................]
    position = x
    while (position > 0 AND dataArray[position-1] > currentData)
      dataArray[position]  = dataArray[..........................]
      position = position - 1
    endwhile

    dataArray[position] = ............................
  next x
endprocedure
```

[3]

(ii) Show how an insertion sort would sort the following data:

6	**1**	**15**	**12**	**5**	**6**	**9**

....

....

....

....

....

....

....

....

....

.... **[6]**

(b) **(i)** Using Big-O notation state the best case complexity of insertion sort.

.... **[1]**

(ii) Explain what your answer to part **(b)(i)** means.

....

....

....

....

....

.... **[3]**

© OCR 2017

(c*) The number of data items in the array is continually increasing.

Insertion sort has a worst case time complexity of O(n^2) and space complexity of O(1).

An alternative sorting algorithm that could be used is bubble sort which also has a worst case time complexity of O(n^2) and space complexity of O(1).

Briefly outline how the bubble sort algorithm works. Discuss the relationship between the complexities and the two sorting algorithms and justify which of the two algorithms is best suited to sorting the array. **[9]**

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

© OCR 2017

....

....

....

....

....

....

....

....

....

....

....

....

....

BLANK PAGE

PLEASE DO NOT WRITE ON THIS PAGE

2 A programmer is developing an ordering system for a fast food restaurant. When a member of staff inputs an order, it is added to a linked list for completion by the chefs.

(a) Explain why a linked list is being used for the ordering system.

....

....

....

.... **[2]**

(b) Each element in a linked list has:

- a pointer, `nodeNo`, which gives the number of that node
- the order number, `orderNo`
- a pointer, `next`, that points to the next node in the list

Fig. 2.1 shows the current contents of the linked list, `orders`.

nodeNo	orderNo	next
0	154	1
1	157	2
2	155	3
3	156	∅

Fig. 2.1

∅ represents a null pointer.

(i) Order 158 has been made, and needs adding to the end of the linked list.

Add the order, 158, to the linked list as shown in Fig. 2.1. Show the contents of the linked list in the following table.

nodeNo	orderNo	next

[2]

(ii) Order 159 has been made. This order has a high priority and needs to be the second order in the linked list.
Add the order, 159, to the original linked list as shown in Fig. 2.1. Show the contents of the linked list in the following table.

nodeNo	orderNo	next

[3]

(c) The linked list is implemented using a 2D array, `theOrders`:

- Row 0 stores `orderNo`
- Row 1 stores `next`

The data now stored in `theOrders` is shown in Fig. 2.2.

184	186	185	187
1	2	3	

Fig. 2.2

`theOrders[1,0]` would return `1`

The following algorithm is written:

```
procedure x()
    finished = false
    count = 0
    while NOT(finished)
        if theOrders[1,count] == null then
            finished = true
        else
            output = theOrders[0,count]
            print(output)
            count = theOrders[1,count]
        endif
    endwhile
    output = theOrders[0,count]
    print(output)

endprocedure
```

(i) Outline why `nodeNo` does not need to be stored in the array.

....

.... **[1]**

(ii) Complete the trace table for procedure `x`, for the data shown in Fig. 2.2.

finished	count	output

[3]

(iii) Describe the purpose of procedure `x`.

....

....

....

.... **[2]**

© OCR 2017

(iv) A new order, 190, is to be added to `theOrders`. It needs to be the third element in the list.
The current contents of the array are repeated here for reference:

184	186	185	187		
1	2	3			

Describe how the new order, 190, can be added to the array, so the linked list is read in the correct order, without rearranging the array elements.

....

....

....

....

....

....

....

.... **[4]**

(d) The user needs to be able to search for, and find, a specific order number.

State an appropriate search algorithm that could be used, and justify your choice against an alternative Search algorithm.

Appropriate Search Algorithm..

Justification...

....

....

....

....

[3]

(e) The programmer is writing the program using an IDE.

Identify **three** features of an IDE that the programmer would use when writing the code and describe how the features benefit the programmer.

1...

....

....

....

2...

....

....

....

3...

....

....

....

[6]

(f*) The programmer is considering using concurrent programming.

Discuss how concurrent programming can be applied to the food ordering system and the benefits and limitations of doing so. **[9]**

....

....

....

....

....

....

....

....

....

....

© OCR 2017

3 An encryption routine reads a line of text from a file, reverses the order of the characters in the string and subtracts 10 from the ASCII value of each letter, then saves the new string into the same file.

The program is split into sub-procedures. Three sub-procedures are described as follows:

- Read string from file
- Push each character of the string onto a stack
- Read and encrypt each character message

(a) **(i)** Identify **one** further sub-procedure that could be used in the program.

.... **[1]**

(ii) Describe **two** advantages of splitting the problem into sub-procedures.

1 ....

....

....

....

2 ....

....

....

.... **[4]**

(b) A function, `readMessage`:

- takes the file name as a parameter
- reads and returns the line of text

Complete the pseudocode algorithm for `readMessage`:

```
function ..............................(fileName)
      messageFile = openRead(..............................)
      message = messageFile.readLine()
      messageFile. ..............................
      return ..............................
endfunction
```

[4]

(c) A function, `push`, can be used to add a character to a stack. For example:

```
theStack.push("H")
```

places the character `H` onto the stack, `theStack`.

A procedure, `pushToStack`, takes a string as a parameter and pushes each character of the message onto the stack, `messageStack`.

Complete the procedure below.

Add comments to explain how your code works.

```
procedure pushToStack(message)
```

....

....

....

....

....

....

....

....

```
endprocedure
```

[5]

(d) Describe the steps that the program would have to take in order to encrypt the characters stored in the stack, and save them in a single variable.

....

....

....

....

....

....

....

....

.... **[5]**

BLANK PAGE

PLEASE DO NOT WRITE ON THIS PAGE

© OCR 2017

4 A data structure is shown below in Fig. 4.1.

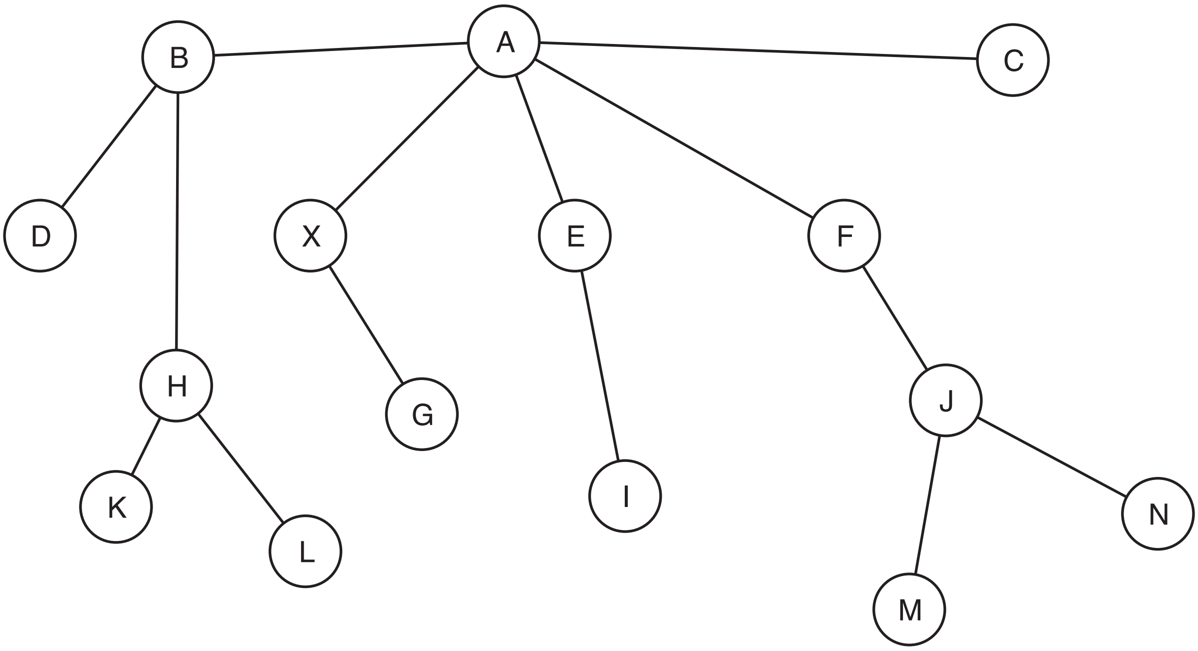


Fig. 4.1

(a) Identify the data structure shown in Fig. 4.1.

.... **[1]**

(b) The programmer is considering using a depth-first (post-order) traversal, or a breadth-first traversal to find the path between node A and node X.

(i) Explain the difference between a depth-first (post-order) and breadth-first traversal.

....

....

....

....

....

....

....

....

.... **[4]**

(ii) Show how a depth-first (post-order) traversal would find the path between node A and node X for the structure shown in Fig. 4.1.

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[6]**

(iii) Explain how you used backtracking in your answer to part **(b)(ii)**.

....

....

....

....

....

.... **[3]**

5 A recursive function, `calculate`, is shown below:

```
01 function calculate(num1, num2)
02    if num1 == num2 then
03        return num1
04    elseif num1 < num2 then
05        return calculate(num1, (num2-num1))
06    else
07        return calculate(num2, (num1-num2))
08    endif
09 endfunction
```

(a) Identify the lines where recursion is used.

.... **[1]**

(b) Trace the algorithm, showing the steps and result when the following line is run:

```
print(calculate(4,10))
```

[5]

© OCR 2017

(c) Re-write the function so it uses iteration instead of recursion.

....

....

....

....

....

....

....

....

....

....

....

....

.... **[4]**

Section B

Answer **all** questions.

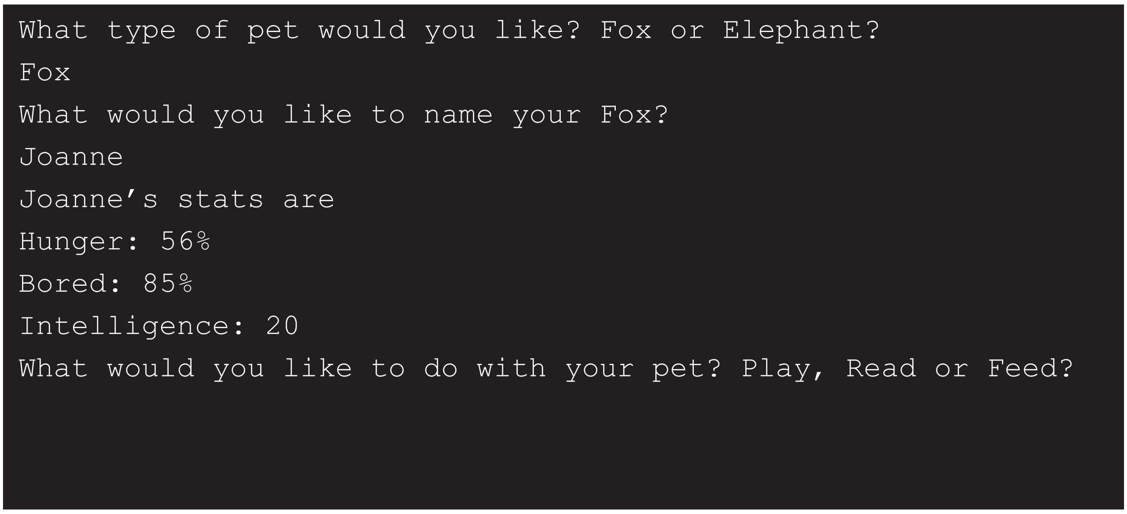
6 A software developer is creating a Virtual Pet game.

The user can choose the type of animal they would like as their pet, give it a name and then they are responsible for caring for that animal. The user will need to feed, play with, and educate their pet.

The aim is to keep the animal alive and happy, for example if the animal is not fed over a set period of time then the pet will die.

- The game tells the user how hungry or bored the animal is as a percentage (%) and the animal's intelligence is ranked as a number between 0 and 150 (inclusive).
- Hunger and boredom increase by 1% with every tick of a timer.
- When the feed option is selected, hunger is reduced to 0.
- When the play option is selected, bored is reduced to 0.
- When the read option is selected, the intelligence is increased by 0.6% of its current value.

An example of the game is shown:



```
What type of pet would you like? Fox or Elephant?
Fox
What would you like to name your Fox?
Joanne
Joanne's stats are
Hunger: 56%
Bored: 85%
Intelligence: 20
What would you like to do with your pet? Play, Read or Feed?
```

Fig. 1.1

(a) Identify **three** inputs that the user will have to enter to start, and/or play the game.

1..

2..

3..

[3]

© OCR 2017

(b) The developer is using decomposition to design the game.

(i) Describe the process of decomposition.

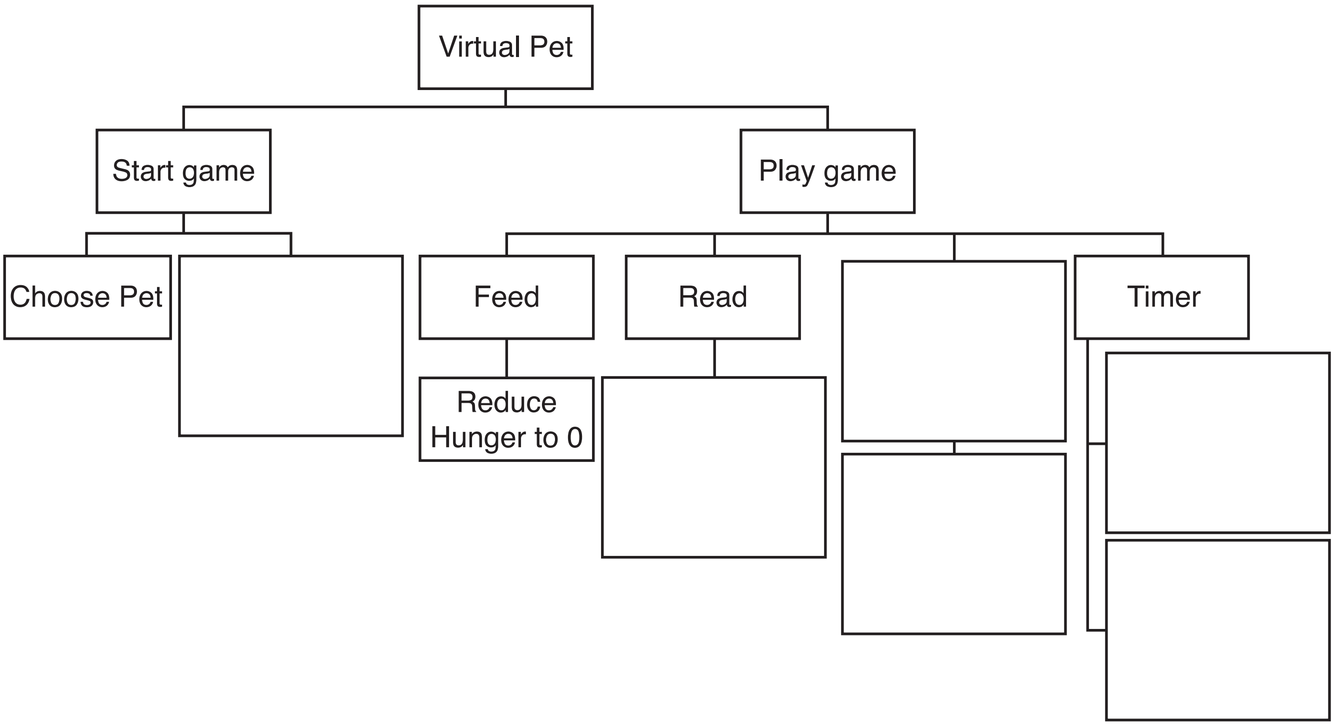
....

....

....

.... **[2]**

(ii) The developer has produced the following structure diagram for the game:



Complete the structure diagram for the Virtual Pet game by filling in the empty boxes.

[6]

(c) The developer needs to write procedures for the options play and read. Each of the options changes its corresponding value, and outputs the results to the screen.

(i) Write a procedure, using pseudocode, to reset `bored` and output the new value in an appropriate message.

....

....

....

....

....

.... **[3]**

(ii) Write a procedure, using pseudocode, to increase `intelligence` by 0.6% and output the new intelligence in an appropriate message.

....

....

....

....

....

.... **[3]**

(d) The developer is extending the game to allow users to have multiple pets of different types. The developer has written a class, `Pet`.

The attributes and methods in the class are described in the table:

Identifier	**Attribute/Method**	**Description**
`petName`	Attribute	Stores the pet's name
`bored`	Attribute	Stores the % bored
`hunger`	Attribute	Stores the % hunger
`intelligence`	Attribute	Stores the intelligence
`type`	Attribute	Stores the type of animal
`new`	Method	Creates a new instance of `pet`
`feed`	Method	Reduces `hunger` to 0 and outputs `hunger`
`play`	Method	Reduces `bored` to 0 and outputs `bored`
`read`	Method	Increases `intelligence` by a set value
`outputGreeting`	Method	Outputs a message to the user

Part of the class declaration is given:

```
class Pet
    private petName
    private bored
    private hunger
    private intelligence
    private type
…
…
```

(i) After a user enters the pet name, and chooses a type, the constructor method of `Pet` is called to create a new instance. The method needs to set `petName`, as well as `hunger`, `bored` and `intelligence` to starting values of 0.

Write, using pseudocode, the constructor method for this class.

....

....

....

....

....

....

....

....

.... **[4]**

(ii) Write a line of code that creates a new instance of `Pet` for a Tiger called "Springy".

.... **[2]**

(iii) The method `outputGreeting` for the superclass is written as follows:

```
public procedure outputGreeting()
    print("Hello, I'm " + petName + ", I'm a " + type)
endprocedure
```

A class is needed for Tiger. The class needs to:

- inherit the methods and attributes from `pet`
- in the constructor, set `type` to Tiger, `intelligence` to 10, `hunger` to 50 and `bored` to 10
- extend the method `outputGreeting`, by outputting an additional line that says “I like to eat meat and roar”

Write, using pseudocode, the class `Tiger`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... **[5]**

© OCR 2017

(e*) The developer made use of abstraction when creating the Virtual Pet game.

Discuss the need for and purpose of abstraction and how abstraction will be used in the development of the game. **[9]**

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

© OCR 2017

(f) The developer is storing the user's pets in a 1-dimensional array. At each timer interval, the array is searched, using a linear search, to check if any pets' hunger or bored values are greater than 90%. If they are, an alert is displayed to the user.

(i) State the complexity of searching the pets in Big-O notation.

.... **[1]**

(ii) A given computer takes 4 milliseconds (ms) to search an array of 20 pets. Calculate an estimate of how long the computer will take to search an array of 100 pets.

Show your working.

....

....

.... **[2]**

END OF QUESTION PAPER

© OCR 2017

PLEASE DO NOT WRITE ON THIS PAGE

Oxford Cambridge and RSA

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) 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.

© OCR 2017

GCE

Computer Science

Unit **H446A/02:** Algorithms and programming

Advanced GCE

Mark Scheme for June 2017

OCR (Oxford Cambridge and RSA) is a leading UK awarding body, providing a wide range of qualifications to meet the needs of candidates of all ages and abilities. OCR qualifications include AS/A Levels, Diplomas, GCSEs, Cambridge Nationals, Cambridge Technicals, Functional Skills, Key Skills, Entry Level qualifications, NVQs and vocational qualifications in areas such as IT, business, languages, teaching/training, administration and secretarial skills.

It is also responsible for developing new specifications to meet national requirements and the needs of students and teachers. OCR is a not-for-profit organisation; any surplus made is invested back into the establishment to help towards the development of qualifications and support, which keep pace with the changing needs of today's society.

This mark scheme is published as an aid to teachers and students, to indicate the requirements of the examination. It shows the basis on which marks were awarded by examiners. It does not indicate the details of the discussions which took place at an examiners' meeting before marking commenced.

All examiners are instructed that alternative correct answers and unexpected approaches in candidates' scripts must be given marks that fairly reflect the relevant knowledge and skills demonstrated.

Mark schemes should be read in conjunction with the published question papers and the report on the examination.

OCR will not enter into any discussion or correspondence in connection with this mark scheme.

© OCR 2017

Annotations

Annotation	Meaning
^	Omission mark
BOD	Benefit of the doubt
✗	Incorrect point
FT	Follow through
NAQ	Not answered question
NBOD	No benefit of doubt given
REP	Repeat
✓	Correct point
TV	Too vague
BP	Blank Page – this annotation must be used on all blank pages within an answer booklet (structured or unstructured) and on each page of an additional object where there is no candidate response.
L1	Level 1
L2	Level 2
L3	Level 3

Subject-specific Marking Instructions

INTRODUCTION

Your first task as an Examiner is to become thoroughly familiar with the material on which the examination depends. This material includes:

- the specification, especially the assessment objectives
- the question paper and its rubrics
- the mark scheme.

You should ensure that you have copies of these materials.

You should ensure also that you are familiar with the administrative procedures related to the marking process. These are set out in the OCR booklet **Instructions for Examiners**. If you are examining for the first time, please read carefully **Appendix 5 Introduction to Script Marking: Notes for New Examiners**.

Please ask for help or guidance whenever you need it. Your first point of contact is your Team Leader.

USING THE MARK SCHEME

Please study this Mark Scheme carefully. The Mark Scheme is an integral part of the process that begins with the setting of the question paper and ends with the awarding of grades. Question papers and Mark Schemes are developed in association with each other so that issues of differentiation and positive achievement can be addressed from the very start.

This Mark Scheme is a working document; it is not exhaustive; it does not provide ‘correct’ answers. The Mark Scheme can only provide ‘best guesses’ about how the question will work out, and it is subject to revision after we have looked at a wide range of scripts.

The Examiners’ Standardisation Meeting will ensure that the Mark Scheme covers the range of candidates’ responses to the questions, and that all Examiners understand and apply the Mark Scheme in the same way. The Mark Scheme will be discussed and amended at the meeting, and administrative procedures will be confirmed. Co-ordination scripts will be issued at the meeting to exemplify aspects of candidates’ responses and achievements; the co-ordination scripts then become part of this Mark Scheme.

Before the Standardisation Meeting, you should read and mark in pencil a number of scripts, in order to gain an impression of the range of responses and achievement that may be expected.

In your marking, you will encounter valid responses which are not covered by the Mark Scheme: these responses must be credited. You will encounter answers which fall outside the ‘target range’ of Bands for the paper which you are marking. Please mark these answers according to the marking criteria.

Please read carefully all the scripts in your allocation and make every effort to look positively for achievement throughout the ability range. Always be prepared to use the full range of marks.

LEVELS OF RESPONSE QUESTIONS:

The indicative content indicates the expected parameters for candidates' answers, but be prepared to recognise and credit unexpected approaches where they show relevance.
Using 'best-fit', decide first which set of BAND DESCRIPTORS best describes the overall quality of the answer. Once the band is located, adjust the mark concentrating on features of the answer which make it stronger or weaker following the guidelines for refinement.

- **Highest mark:** If clear evidence of all the qualities in the band descriptors is shown, the HIGHEST Mark should be awarded.
- **Lowest mark:** If the answer shows the candidate to be borderline (i.e. they have achieved all the qualities of the bands below and show limited evidence of meeting the criteria of the band in question) the LOWEST mark should be awarded.
- **Middle mark:** This mark should be used for candidates who are secure in the band. They are not 'borderline' but they have only achieved some of the qualities in the band descriptors.

Be prepared to use the full range of marks. Do not reserve (e.g.) high Band 3 marks 'in case' something turns up of a quality you have not yet seen. If an answer gives clear evidence of the qualities described in the band descriptors, reward appropriately.

	AO1	**AO2**	**AO3**
High (thorough)	Precision in the use of question terminology. Knowledge shown is consistent and well-developed. Clear appreciation of the question from a range of different perspectives making extensive use of acquired knowledge and understanding.	Knowledge and understanding shown is consistently applied to context enabling a logical and sustained argument to develop. Examples used enhance rather than detract from response.	Concerted effort is made to consider all aspects of a system / problem or weigh up both sides to an argument before forming an overall conclusion. Judgements made are based on appropriate and concise arguments that have been developed in response resulting in them being both supported and realistic.
Middle (reasonable)	Awareness of the meaning of the terms in the question. Knowledge is sound and effectively demonstrated. Demands of question understood although at times opportunities to make use of acquired knowledge and understanding not always taken.	Knowledge and understanding applied to context. Whilst clear evidence that an argument builds and develops through response there are times when opportunities are missed to use an example or relate an aspect of knowledge or understanding to the context provided.	There is a reasonable attempt to reach a conclusion considering aspects of a system / problem or weighing up both sides of an argument. However the impact of the conclusion is often lessened by a lack of supported judgements which accompany it. This inability to build on and develop lines of argument as developed in the response can detract from the overall quality of the response.
Low (basic)	Confusion and inability to deconstruct terminology as used in the question. Knowledge partial and superficial. Focus on question narrow and often one-dimensional.	Inability to apply knowledge and understanding in any sustained way to context resulting in tenuous and unsupported statements being made. Examples if used are for the most part irrelevant and unsubstantiated.	Little or no attempt to prioritise or weigh up factors during course of answer. Conclusion is often dislocated from response and any judgements lack substance due in part to the basic level of argument that has been demonstrated throughout response.

	Assessment Objective
AO1	Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.
AO1.1	Demonstrate **knowledge** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO1.2	Demonstrate **understanding** of the principles and concepts of abstraction, logic, algorithms, data representation or other as appropriate.
AO2	Apply knowledge and understanding of the principles and concepts of computer science including to analyse problems in computational terms.
AO2.1	Apply knowledge and understanding of the principles and concepts of computer science.
AO2.2	Analyse problems in computational terms.
AO3	Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.
AO3.1	Design computer systems that solve problems.
AO3.2	Program computer systems that solve problems.
AO3.3	Evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

Question			Answer	Marks	Guidance
1	a	i	1 mark for each correct item in **bold**	**3** **AO1.1** **(3)**	answers must be in the correct case as given e.g. **currentData**

```
procedure sortit(dataArray, lastIndex)
   for x = 1 to lastIndex
      currentData = dataArray[x]
      position = x
      while (position > 0 AND dataArray[x - 1] > currentData)
         dataArray[position]  = dataArray[position-1]
         position = position - 1
      endwhile
      dataArray[position] = currentData

   next x
endprocedure
```

Question			Answer	Marks	Guidance
1	a	ii	1 mark for contents of each row in table	**6** **AO2.1** **(6)**	… each row is dependent upon the preceding row being correct

6	**1**	**15**	**12**	**5**	**6**	**9**		
1	**6**	**15**	**12**	**5**	**6**	**9**	6 is the sorted list 1 is the compared to sorted list 1 is put in place in sorted list	1
1	**6**	**15**	**12**	**5**	**6**	**9**	15 is compared 15 is in place in sorted list	1
1	**6**	**12**	**15**	**5**	**6**	**9**	12 is compared 12 is in place in sorted list	1
1	**5**	**6**	**12**	**15**	**6**	**9**	5 is compared 5 is in place in sorted list	1
1	**5**	**6**	**6**	**12**	**15**	**9**	6 is compared 6 is in place in sorted list	1
1	**5**	**6**	**6**	**9**	**12**	**15**	9 is compared and put in place	1

1	**b**	**i**	O(n)	**1** **AO1.1** **(1)**	
1	**b**	**ii**	1 mark per bullet to max 3 • The best case is for a sorted list (O(n)) • As the number of elements increases • … the number of steps increases in a <u>linear</u> fashion	**3** **AO1.2** **(3)**	B(ii) dependent upon b(i) being correct i.e. answers for O(n) only Accept appropriate graph for bullet points 2 and 3
1	**c**		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of how bubble sort works and Big O complexity; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of of how bubble sort works and Big O complexity; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level**	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** • Description of bubble sort: ◊ Starting at the beginning of the list Items are swapped with their neighbour if they are out of order. ◊ Each pair of neighbours is checked in order. ◊ When a swap is made a flag is set. ◊ If at the end of the list the flag has been set the flag is unset and the algorithm starts from the beginning of the list again. ◊ When the algorithm gets to the end of the list and the flag is unset the list is sorted and the

			(1-3 marks) The candidate demonstrates a **basic** knowledge of of how bubble sort works and Big O complexity with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		algorithm finishes. • O(n^2) denotes as the data size increases the time the list takes to sort increases in a quadratic manner. • O(1) denotes the space used is constant **AO2: Application** • As data set gets bigger, bubble sort's time gets larger at an increasing rate.. • Complexity doesn't denote the actual time but the order with which the time/space grows. • O(1) space complexity means no matter how big the data set becomes the amount of space (extra to the data itself) remains the same. • O(n^2) time complexity means as n increases time increases by n^2 / if n doubles the time taken is squared. • Bubble sort can be tweaked with improvements (e.g. checking one less item per iteration and alternating

					sorting directions). • These optimisations don't change the complexity. IT will run a little quicker on smaller sets but time taken increases rapidly with data size. • When choosing an algorithm we may also want to take into account the average and best case scenarios. (in this case they are also the same for both algorithms.) **AO3: Evaluation** • The algorithms may have the same time complexity but this does not mean they take the same time to execute on the same data set. • Insertion sort generally performs quicker than bubble sort and is therefore preferable. (Neither scale well however.) • Both algorithms have a space complexity of O(1). This is because both algorithms are in-place (i.e. all sorting takes place

					within the actual data). • Both have a time complexity of O(n^2) as a consequence of their nested loops. (NB last two points are only likely to appear in the very highest mark answers.)

2	**a**		1 mark per bullet, to max 2, e.g. • Orders can be processed in the order they are in the queue • Orders can be inserted at any place in the list e.g. high priority item inserted earlier in the list • Orders can be deleted from any position in the list once they are complete • List is dynamic… • … to allow orders to be added / deleted	**2** **AO1.2** **(1)** **AO2.1** **(1)**	
2	**b**	**i**	1 mark per bullet • `nodeNo` and `next` columns are both correct • `orderNo` column is correct `nodeNo` `orderNo` `next`	**2** **AO1.2** **(2)**	

<table>
<tr><td></td><td></td><td></td><td><table><tr><td>∅</td><td>154</td><td>1</td></tr><tr><td>1</td><td>157</td><td>2</td></tr><tr><td>2</td><td>155</td><td>3</td></tr><tr><td>3</td><td>156</td><td>4</td></tr><tr><td>4</td><td>158</td><td>∅</td></tr></table></td><td></td><td></td></tr>
<tr><td><b>2</b></td><td><b>b</b></td><td><b>ii</b></td><td>1 mark per correct column
<table><tr><th><code>nodeNo</code></th><th><code>orderNo</code></th><th><code>next</code></th></tr><tr><td>∅</td><td>154</td><td>4</td></tr><tr><td>1</td><td>157</td><td>2</td></tr><tr><td>2</td><td>155</td><td>3</td></tr><tr><td>3</td><td>156</td><td>∅</td></tr><tr><td>4</td><td>159</td><td>1</td></tr></table></td><td><b>3
AO1.2
(3)</b></td><td></td></tr>
<tr><td><b>2</b></td><td><b>c</b></td><td><b>i</b></td><td>• The index/subscript of the array acts as the nodeNo</td><td><b>1
AO1.2
(1)</b></td><td></td></tr>
</table>

<table>
<tr><td><b>2</b></td><td><b>c</b></td><td><b>ii</b></td><td>1 mark for each correctly completed column
<table><tr><th>Finished</th><th>Count</th><th>output</th></tr><tr><td>False</td><td>0</td><td>184</td></tr><tr><td>(False)</td><td>1</td><td>186</td></tr><tr><td>(False)</td><td>2</td><td>185</td></tr><tr><td>True</td><td>3</td><td>187</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table></td><td><b>3
AO1.2
(1)
AO2.2
(2)</b></td><td></td></tr>
<tr><td><b>2</b></td><td><b>c</b></td><td><b>iii</b></td><td>1 mark per bullet to max 2
• Output the order numbers ...
• ...in the order they are in the linked list</td><td><b>2
AO2.2
(2)</b></td><td></td></tr>
<tr><td><b>2</b></td><td><b>c</b></td><td><b>iv</b></td><td>1 mark per bullet to max
• Order 190 is added to the end
• Pointers are updated
• 186 will point to 4
• 190 will point to 2
OR
<table><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>Data</td><td>184</td><td>186</td><td>185</td><td>187</td><td>190</td></tr><tr><td>Pointer</td><td>1</td><td>4</td><td>3</td><td></td><td>2</td></tr></table></td><td><b>4
AO1.2
(2)
AO2.1
(2)</b></td><td>If a diagram is given then the mark for updating the pointers is implicit</td></tr>
</table>

2	d		Algorithm, max 1 • linear Justification, 1 mark per bullet to max 2 • Items do not have to be in a specific order • Binary needs items in order	**3** **AO1.1** **(1)** **AO2.1** **(2)**	No marks for justification if linear has not been identified

2	e		1 mark for feature, 1 for benefit. Max 2 per feature. e.g. • Auto-complete • Can view identifiers/avoid spelling mistakes • Colour coding text/syntax highlighting • Can identify features quickly/use to check code is correct • Stepping • Run one line at a time and check result • Breakpoints • Stop the code at a set point to check value of variable(s) • Variable watch/watch window	**6** **AO1.1** **(3)** **AO1.2** **(3)**	Question states when writing the code, therefore use of compiler/producing .exe etc. are not awarded marks Accept any suitable features e.g. traces, crash dump, stack contents, cross-references, line numbers, auto-indent

			• Check values of variables and how they change during the execution • Error diagnostics • Locate and report errors/give detail on errors		
2	f		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of concurrent programming; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of concurrent programming; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a **basic** knowledge of concurrent programming with limited understanding shown; the material is basic and contains some	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** • Processes are happening at the same time/at overlapping times • Only 1 process can actually happen at a time on a single core processor, concurrent tries to simulate multiple processes • One process may need to start before a second has finished • Individual processes are threads, each thread has a life line **AO2: Application** • Multiple orders can be made and added to the list at the same time • Programming will need to allow multiple threads to manipulate a single list • Will allow those reading and writing

			inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		to manipulate at the same time • Locking will need implementing – more complex programming **AO3: Evaluation** • Will allow for multiple orders at the same time – as it would happen in real life • Access to the linked list will need to be limited so it cannot be accessed/overwritten by two threads trying to do different operations • Not all of the process will be paralleliseable. X processors does not mean it will run in 1/xth of the time of one processor.

3	**a**	**i**	Save string in file	**1** **AO2.2** **(1)**	
3	**a**	**ii**	1 mark per bullet, max 2 per advantage to max 4 e.g. • Procedures can be re-used • No need to reprogram/saves time • Program can be split between programmers • Can specialise in their area • Speed up completion time • As multiple procedures worked on concurrently • Easy to test/debug • As each module can be tested on its own then combined.	**4** **AO1.2** **(4)**	Allow any appropriate advantages

3	**b**		1 mark each
			`function readMessage(fileName)`
			`   messageFile = openRead(fileName)`
			`   message = messageFile.readLine()`
			`   messageFile.close()`
			`      return message`
			`endfunction`
				4
				AO2.1
				(4)
					We are not testing pseudocode knowledge – answers that work but do not match the pseudo code given should still be credited full marks.
					`readMessage` and **`fileName`** and **`message`** are case sensitive

3	**c**		1 mark per bullet to max 5
			• Use of appropriate loop
			• Correct end condition (length of `message`)
			• Correct use of `.push` with `messageStack`
			• Accessing `substring` (or equivalent) correctly
			• Appropriate comment(s)
			`procedure pushToStack(message)`
			`   for x = 0 to message.length() //loop through each`
			`                                  //letter`
			`       messageStack.push(message.substring(x,1)) //take`
			`              //each character and push onto stack`
			`   next x  //move to next letter`
			`endprocedure`
				5
				AO2.1
				(2)
				AO3.2
				(3)	

3	**d**		1 mark per bullet to max 5 • Pop element from stack • Convert to ASCII value • Subtract 10 from ASCII value • Convert back to character • Append/concatenate with variable	**5** **AO1.2** **(2)** **AO2.2** **(3)**	Accept pseudocode equivalent.

4	**a**		Tree // Graph (undirected)	**1** **AO1.2** **(1)**	Do not accept binary tree
4	**b**	**i**	1 mark per bullet to max 4 • Depth-first goes to left child node when it can... • If there is no left child it goes to the right child • when there are no child nodes the algorithm 'visits' it' and backtracks to the parent node. • Breadth-first visits all nodes connected directly to start node... • Then visits all nodes directly connected to each of those nodes (and then all nodes directly connected to those nodes and so on…) • Depth-first uses a stack • Breadth-first uses a queue	**4** **AO1.2** **(4)**	
4	**b**	**ii**	1 mark per node in correct order D→K→L→H→B→G→(X)	**6** **AO2.1** **(6)**	
4	**b**	**iii**	Max 3 e.g. • When a node does not have any node to visit e.g. D • The algorithm goes back to the previous visited node e.g. B • To check for further nodes to visit e.g. H	**3** **AO1.2** **(2)** **AO2.1** **(1)**	

			- This repeats until a new node can be visited, or all nodes have been visited		

<table>
<tr><td>5</td><td>a</td><td></td><td>05 and 07</td><td>1
AO2.1
(1)</td><td></td></tr>
<tr><td>5</td><td>b</td><td></td><td>1 mark for each highlighted element
<pre>
• calculate(4,10)

     if 4 == 10 FALSE
     elseif 4 < 10 TRUE
        return calculate(4, (10-4))
        return calculate(4, 6)
             • if 4 == 6 FALSE

                elseif 4 < 6 TRUE
                    return calculate(4, 6-4)
                    return calculate(4, 2)
                      • if 4 == 2 FALSE

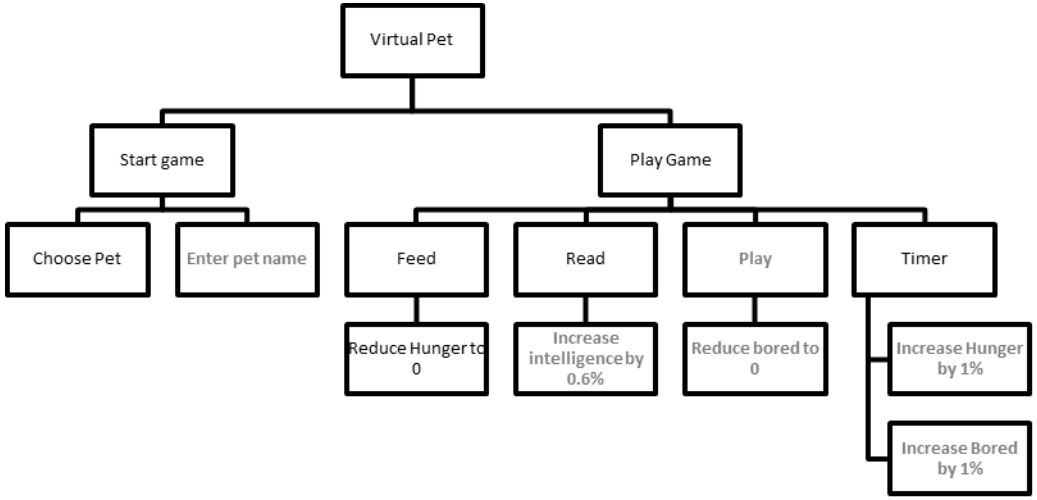
                         elseif 4 < 2 FALSE
                         else
                             return calculate(2, 4-2)
                             return calculate(2,2)
                                   • if 2 == 2 TRUE

                                      return 2
                             return 2
                    return 2
       return 2
</pre></td><td>5
AO2.1
(5)</td><td>Allow trace table or any sensible equivalent.</td></tr>
</table>

			• `output(2)`		

5	**c**		1 mark per bullet to max 4
			• Suitable loop with correct condition
			• In IF: Overwriting num2 with num2 – num1
			• In ELSE: Overwriting num1 with num2...
			• ... Overwriting num2 with num1-num2 correctly (using a temp variable)
			
			e.g.
			`while num1 != num2`
			`    if num1 < num2 then`
			`        num2 = num2 - num1`
			`    else`
			`        temp = num1 - num2`
			`        num1 = num2`
			`        num2 = temp`
			`    endif`
			`endwhile`
				4
				AO2.1
				(1)
				AO2.2
				(1)
				AO3.2
				(2)
					Alternatively swapping values by:
					`temp = num1`
					`num1 = num2`
					`num2 = temp - num2`

6	a		1 mark per input to max 3 • Choice of pet • Pet name • Feed • Play • Read	**3 AO2.1 (3)**	Allow any reasonable input to this system
6	b	i	1 mark per bullet to max 2 • Splitting a problem down • Into its component parts/sub-procedures/modules	**2 AO1.1 (2)**	
6	b	ii	1 mark per box	**6 AO2.2 (6)**	Calculations must be correct

			 Virtual Pet Start game Play Game Choose Pet Enter pet name Feed Read Play Timer Reduce Hunger to 0 Increase intelligence by 0.6% Reduce bored to 0 Increase Hunger by 1% Increase Bored by 1%		

6	c	i	1 mark per bullet to max 3
			- Defining procedure play
			  ○ Resetting bored to 0
			  ○ Outputting result
			
			e.g.
			`procedure play()`
			`    bored = 0`
			`    print("bored: " + bored + "%")`
			`endprocedure`
				3
				AO3.2
				(3)	

6	c	ii	1 mark per bullet to max 3
			
			- Defining procedure read
			  ○ Correct calculation
			  ○ Outputting result
			
			e.g.
			`procedure read()`
			`    intelligence = intelligence * 1.006`
			`    print("intelligence: " + intelligence)`
			`endprocedure`
				3
				AO2.2
				(1)
				AO3.2
				(2)	

6	d	i	1 mark per bullet to max 4 - Correct declaration, appropriate name (e.g. `new`) - Taking `name` and `theType` as a parameter - Setting `petName` to parameter	**4** **AO2.2** **(1)** **AO3.2** **(3)**	

			• Setting `bored`, `hunger` and `intelligence` to 0
			e.g.
			`public procedure new(name, theType)`
			`    petName = name`
			`    bored = 0`
			`    hunger = 0`
			`    intelligence = 0`
			`    type = theType`
			`endprocedure`		

6	d	ii	1 mark per bullet to max 2 • `myPet`/appropriate = `new pet` • `Springy` and `Tiger`, in `""`, in same order as constructor declaration e.g. `myPet = new pet("Springy", "Tiger")`	**2** **AO2.1** **(2)**	
6	d	iii	1 mark per bullet to max • Class declaration including inherit (or equivalent e.g. Tiger extends Pet, Tiger::Pet, Tiger(Pet)) • Constructor procedure (new) with all attributes present • bored = 10, hunger = 50, intelligence = 10, type = “Tiger” • outputGreeting procedure • Outputting original and new messages correctly e.g. `class Tiger inherits Pet`	**5** **AO2.2** **(2)** **AO3.2** **(3)**	

			`    public procedure new(name)`
			`        petName = name`
			`        bored = 10`
			`        hunger = 50`
			`        intelligence = 10`
			`        type = "Tiger"`
			`    endprocedure`
			
			`    public procedure outputGreeting()`
			`        print("Hello, I'm " + petName + ", I'm a " + type)`
			`        print("I like to eat meat and roar")`
			`    endprocedure`
			`endclass`
					Accept
					`super.outputGreeting()`
					In place of first `print` statement

6	e		**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of abstraction; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of abstraction; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a **basic** knowledge of abstraction with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.*	**9** **AO1.1** **(2)** **AO1.2** **(2)** **AO2.1** **(2)** **AO3.3** **(3)**	**AO1: Knowledge and Understanding** **Indicative content** • Removal of unnecessary elements • Uses symbols to represent elements of the problem • Increase chance of creating the program successfully • Reduces programming time and factors that can detract from the program **AO2: Application** • Examples of use in this system e.g. ○ Environment is not shown ○ Movements reduced/removed ○ Other factors that can be done/affect the 'pet' are removed ○ Time may not represented as minutes, seconds **AO3: Evaluation** • Reduces complexity of programming • Requires less computational power, so the game can be played on lower spec

			0 marks No attempt to answer the question or response is not worthy of credit.		devices e.g. phones • Focus is on the core aspects of the program rather than the extras • Too much abstraction can detract from the appeal of the game, may be too simplistic/not realistic enough, may not have enough scope to engage users
6	f	i	<u>O(n)</u>	**1** **AO1.1** **(1)**	
6	f	ii	1 mark per bullet to max 2 • 20(ms) • … showing working	**2** **AO1.2** **(1)** **AO2.1** **(1)**	

OCR (Oxford Cambridge and RSA Examinations)
1 Hills Road
Cambridge
CB1 2EU

OCR Customer Contact Centre

Education and Learning
Telephone: 01223 553998
Facsimile: 01223 552627
Email: general.qualifications@ocr.org.uk

www.ocr.org.uk

For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored

Oxford Cambridge and RSA Examinations
is a Company Limited by Guarantee
Registered in England
Registered Office; 1 Hills Road, Cambridge, CB1 2EU
Registered Company Number: 3484466
OCR is an exempt Charity

OCR (Oxford Cambridge and RSA Examinations)
Head office
Telephone: 01223 552552
Facsimile: 01223 552553

PART OF THE
CAMBRIDGE ASSESSMENT
GROUP

© OCR 2017