ComputerScience4U.com

OCR
Oxford Cambridge and RSA

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.

ComputerScience4U.com

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.

ComputerScience4U.com

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.

ComputerScience4U.com

	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.

ComputerScience4U.com

	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.

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>1</td><td>a</td><td>i</td><td>1 mark for each correct item in bold

<pre>
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
</pre></td><td>3
AO1.1
(3)</td><td>answers must be in the correct case as given
e.g. currentData</td></tr>
<tr><td>1</td><td>a</td><td>ii</td><td>1 mark for contents of each row in table

<table>
<tr><td>6</td><td>1</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td></td><td></td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>6 is the sorted list
1 is the compared to sorted list
1 is put in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>15 is compared
15 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>12</td><td>15</td><td>5</td><td>6</td><td>9</td><td>12 is compared
12 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>12</td><td>15</td><td>6</td><td>9</td><td>5 is compared
5 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>12</td><td>15</td><td>9</td><td>6 is compared
6 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>9</td><td>12</td><td>15</td><td>9 is compared and put in place</td><td>1</td></tr>
</table></td><td>6
AO2.1
(6)</td><td>… each row is dependent upon the preceding row being correct</td></tr>
</table>

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

ComputerScience4U.com

					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

ComputerScience4U.com

					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)**	

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

<table>
<tr>
<td>2</td><td>c</td><td>ii</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>3
AO1.2
(1)
AO2.2
(2)</td>
<td></td>
</tr>
<tr>
<td>2</td><td>c</td><td>iii</td>
<td>1 mark per bullet to max 2
• Output the order numbers ...
• ...in the order they are in the linked list</td>
<td>2
AO2.2
(2)</td>
<td></td>
</tr>
<tr>
<td>2</td><td>c</td><td>iv</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>4
AO1.2
(2)
AO2.1
(2)</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 <u>linear</u> 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

ComputerScience4U.com

			• 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

ComputerScience4U.com

			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)**	

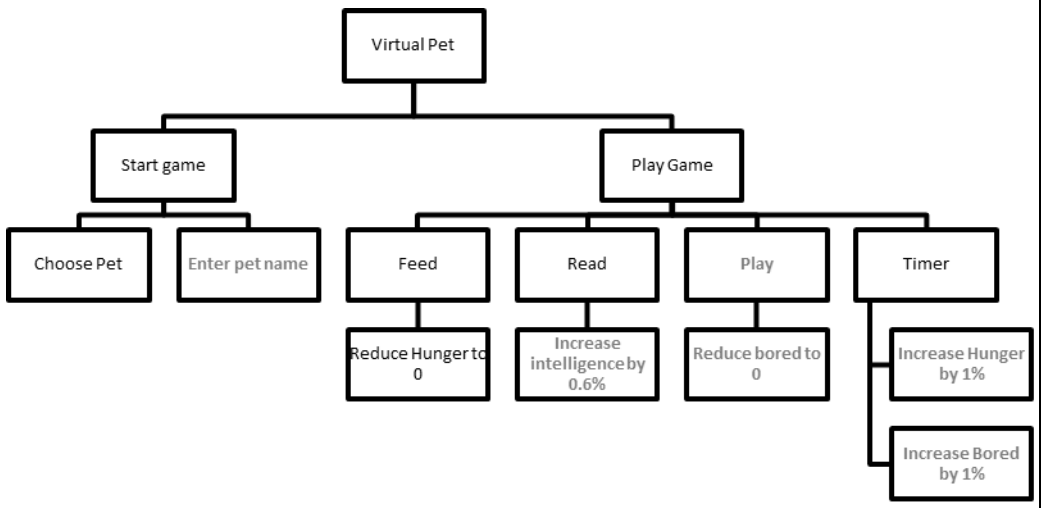
			• This repeats until a new node can be visited, or all nodes have been visited		

5	**a**		05 and 07	**1 AO2.1 (1)**	
5	**b**		1 mark for each highlighted element • 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	**5 AO2.1 (5)**	Allow trace table or any sensible equivalent.

			• `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
			o Resetting bored to 0
			o 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
			o Correct calculation
			o 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

ComputerScience4U.com

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. o Environment is not shown o Movements reduced/removed o Other factors that can be done/affect the 'pet' are removed o 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

© OCR 2017

PART OF THE CAMBRIDGE ASSESSMENT GROUP

LLOYD'S REGISTER QUALITY ASSURANCE · ISO9001 · UKAS MANAGEMENT SYSTEMS 001