



**GCSE (9–1)**

**Computer Science**

**J276/02:** Computational thinking, algorithms and programming

General Certificate of Secondary Education

**Mark Scheme for November 2020**

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 2020

**Annotations**

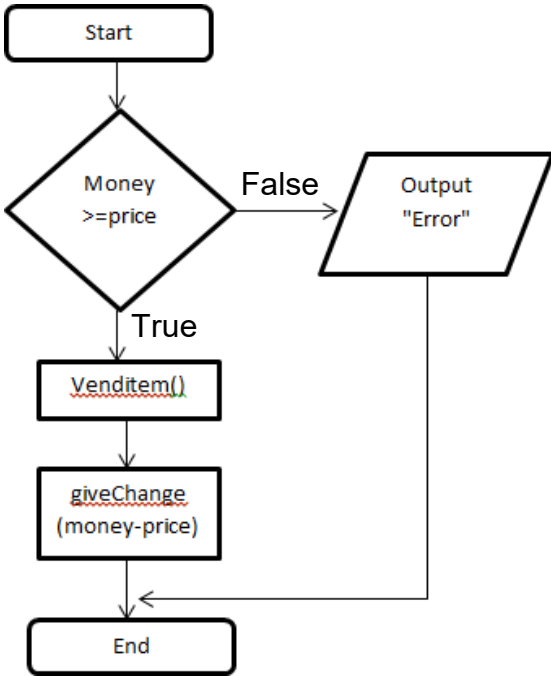
Annotation	Meaning
<b>SEEN</b>	Answer seen – this annotation <b>must</b> be used on all blank pages within an answer booklet (structured or unstructured) and on each page of an additional object where there is no candidate response.
	Omission mark
	Benefit of doubt
	Cross
	Follow through
	Not answered question
	Benefit of doubt not given
	Repeat
	Tick

Each question must include annotation. All marks credited on point-based marking must be given a tick.

Question			Answer	Mark	Guidance															
1	(a)		1 mark for each letter <table><tr><td>Decomposition</td><td>D</td></tr><tr><td>Abstraction</td><td>B</td></tr><tr><td>Input Sanitisation</td><td>A</td></tr><tr><td>Casting</td><td>F</td></tr></table>	Decomposition	D	Abstraction	B	Input Sanitisation	A	Casting	F	4  AO1 1a(4)	Accept answers that write the definition instead of the letter.							
Decomposition	D																			
Abstraction	B																			
Input Sanitisation	A																			
Casting	F																			
1	(b)	(i)	<ul style="list-style-type: none"><li><code>timer = 7.3</code></li></ul>	1  AO3 2b(1)	Ignore dim / define / etc and data types Do not allow use of string delimiters or other unsuitable data types. Allow other suitable assignment symbols (e.g. := ) Do not allow == for assignment. Do not penalise case. Spelling must be accurate															
1	(b)	(ii)	<ul style="list-style-type: none"><li>Real // Float</li></ul>	1  AO2 1b(1)	Allow decimal, single, double or equivalent															
2	(a)		<table><tr><td>Line</td><td>Program code</td><td>Output</td></tr><tr><td>08</td><td><code>print score</code></td><td><b>18</b></td></tr><tr><td>09</td><td><code>print "name"</code></td><td><b>name</b></td></tr><tr><td>10</td><td><code>print newscore(score,2)</code></td><td><b>37</b></td></tr><tr><td>11</td><td><code>print score</code></td><td><b>18</b></td></tr></table>	Line	Program code	Output	08	<code>print score</code>	<b>18</b>	09	<code>print "name"</code>	<b>name</b>	10	<code>print newscore(score,2)</code>	<b>37</b>	11	<code>print score</code>	<b>18</b>	4  AO2 1b(4)	
Line	Program code	Output																		
08	<code>print score</code>	<b>18</b>																		
09	<code>print "name"</code>	<b>name</b>																		
10	<code>print newscore(score,2)</code>	<b>37</b>																		
11	<code>print score</code>	<b>18</b>																		

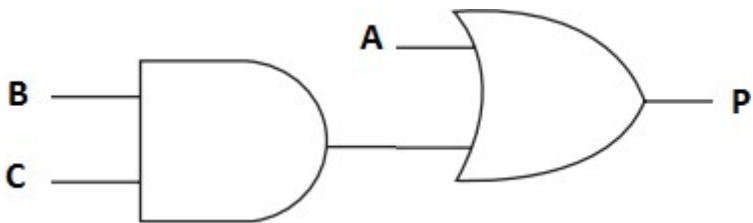
2	(b)	1 mark per bullet to max 2 <ul style="list-style-type: none"> <li>• <b>Easier/quicker</b> for humans to <b>write</b></li> <li>• <b>Easier/quicker</b> to <b>read / understand / remember</b></li> <li>• <b>Easier/quicker</b> to <b>maintain / debug / spot errors</b></li> <li>• ...because code is closer to English / uses English words</li> <li>• Less code to write</li> <li>• ...because one HLL instruction represents many assembly instructions</li> <li>• Portable (between processors) // will work with different types of computer</li> </ul>	2 AO1 1b(2)	Accept “human language” as English for BP4  “Easier to use” is too vague.
2	(c)	1 mark per bullet to max 2 <ul style="list-style-type: none"> <li>• Each character (in character set) has a <b>unique</b> (binary) number/value</li> <li>• Each character in the <b>string</b> is assigned its associated number/value</li> <li>• The (binary) value of each character is stored/combined (in order)</li> <li>• ... by example e.g. The binary value for D, then for r, then for u</li> <li>• Uses ASCII/Extended ASCII/Unicode</li> </ul>	2 AO2 1a(2)	

Question			Answer	Mark	Guidance															
3	(a)	(i)	1 mark per bullet to max 2 e.g. <ul style="list-style-type: none"><li>• Check the program meets the <b>user requirements</b></li><li>• Check the program works (as intended) // detect logic / syntax errors</li><li>• Check the program does not crash (under invalid entry) // check error messages are suitable</li><li>• ...allow these errors to be fixed</li><li>• ...make sure there are no problems when released</li><li>• Any suitable example <b>related to the vending machine</b> e.g. gives correct change</li></ul>	2  AO1 1b(2)	Allow two any suitable examples for two marks  BOD “find errors”, “find bugs” for BP2  “fix errors” by itself is one mark (BP4)															
3	(a)	(ii)	1 mark per bullet to max 2 <ul style="list-style-type: none"><li>• Iterative is during development // repeatedly testing <u>after/while making changes</u></li><li>• Final is when the development is (almost) complete // done after iterative testing</li></ul>	2  AO1 1b(2)	Do not accept just “repeatedly testing” for iterative  BOD “iterative testing tests modules/sections”															
3	(a)	(iii)	<table><tr><th>Code entered</th><th>Money inserted</th><th>Expected result</th></tr><tr><td></td><td></td><td></td></tr><tr><td>C2</td><td></td><td></td></tr><tr><td></td><td>£0.49 (or any value less than £0.50)</td><td></td></tr><tr><td></td><td></td><td>Invalid Selection (or any suitable error message)</td></tr></table>	Code entered	Money inserted	Expected result				C2				£0.49 (or any value less than £0.50)				Invalid Selection (or any suitable error message)	3  AO3 2b(3)	For £0.49 accept any value <£0.50. Must be a specific value, not a description.  Accept any suitable error message for invalid selection
Code entered	Money inserted	Expected result																		
C2																				
	£0.49 (or any value less than £0.50)																			
		Invalid Selection (or any suitable error message)																		

3	(b)	(i)	<ul style="list-style-type: none"> <li>money</li> <li>price</li> </ul>	1 AO1 1b(1)	Must be an identifier, not description. Ignore case.
3	(b)	(ii)	<ul style="list-style-type: none"> <li>one</li> </ul>	1 AO2 1b(1)	
3	(c)		<p>1 mark per bullet</p> <ul style="list-style-type: none"> <li>Checking if <code>money &gt;= price</code>...</li> <li>...decision (diamond shape) used</li> <li>...<code>venditem()</code> and <code>giveChange(money-price)</code> if <u>True/Yes</u></li> <li>...output an error if <u>False / No</u></li> <li>Terminator used to start and end the program and all paths terminated</li> </ul>	5 AO3 2b(5)	<p>Reasonable attempt at BP1 needed for credit BP2, 3 and 4</p> <p>Ignore other additional code.</p> <p>BP3 and BP4 must follow on from True/False // Yes/No decision to be credited.</p> <p>Subroutines names and parameters must be correct. Ignore missing brackets on <code>venditem</code>.</p>  <pre> graph TD     Start([Start]) --&gt; Decision{Money &gt;= price}     Decision -- False --&gt; Output[/Output "Error"/]     Decision -- True --&gt; Venditem[Venditem()]     Venditem --&gt; GiveChange[giveChange (money-price)]     GiveChange --&gt; End([End])     Output --&gt; End   </pre>

Question			Answer	Mark	Guidance
3	d	i	1 mark per bullet to max 2 <ul style="list-style-type: none"> <li>• Indentation // whitespace</li> <li>• Appropriately named variables / identifiers</li> <li>• Modularisation / use of subroutines</li> </ul>	2  AO2 1b(2)	
3	d	ii	<ul style="list-style-type: none"> <li>• Comments</li> <li>• Use of constants</li> </ul>	1  AO2 1b(1)	
3	e		<ul style="list-style-type: none"> <li>• <code>SELECT ItemCode // *</code></li> <li>• <code>FROM ITEMS</code></li> <li>• <code>WHERE</code></li> <li>• <code>...Stock &lt; 10</code></li> </ul>	4  AO3 2b(4)	Accept other fields shown in addition to <code>ItemCode</code>  Accept <code>Stock &lt;=9 / etc.</code>  Ignore case. Spelling of fields and table must be correct.  If <code>WHERE</code> missing, <code>Stock &lt; 10</code> must be after <code>FROM</code> clause.
3	f		1 mark per bullet <ul style="list-style-type: none"> <li>• Input from user</li> <li>• Check IF input value is “on”...</li> <li>• ... if so, assign 1 to <code>statevalue</code></li> <li>• Correct assignment of 2 for “off” and 3 for “suspended” with correct state and IF</li> <li>• Correct logical check (else) to output “invalid state” <u>if no state set</u></li> </ul>	5  AO3 2b(5)	Accept alternative error messages. Variable names must not include obvious spaces.  BP3 dependent on BP2. BP2 and BP4 must be a logical comparison using IF and not just the CASE statement. NE to simply replace CASE with IF.  Penalise each error once then apply FT.  e.g. <pre> newstate = input("Enter the new state : ") if newstate == "on" then     statevalue = 1 elseif newstate = "off" then     statevalue = 2 elseif newstate = "suspended"     statevalue = 3 else     print("Invalid state") endif           </pre>



4	(a)		<ul style="list-style-type: none"> <li>E 3</li> </ul>	2 AO1 1b(2)	1 mark per digit (mark right to left) Max 1 if any additional leading values
4	(b)		<ul style="list-style-type: none"> <li>0110 1001 <u>must be 8 bits</u></li> </ul>	2 AO1 1b(2)	1 mark per nibble (mark right to left). Max 1 if any additional leading values
4	(c)		1 mark per bullet to max 2 <ul style="list-style-type: none"> <li>Easier/quicker to communicate / enter / write / read / remember</li> <li>Less chance of input errors // easier to spot errors</li> <li>They are smaller / shorter</li> <li>Easy to convert between binary and Hexadecimal</li> </ul>	2 AO1 1b(2)	Mark response as a whole.  Do not accept answers simply describing what hexadecimal is.  “easier to understand” or “easier to use” on its own is NE  BP3 (smaller) must refer to size when written down, NOT size when stored which is unaffected)
4	(d)	(i)	<ul style="list-style-type: none"> <li>3</li> </ul>	1 AO1 1b(1)	CAO
4	(d)	(ii)	<ul style="list-style-type: none"> <li>1</li> </ul>	1 AO1 1b(1)	CAO
4	e		<ul style="list-style-type: none"> <li>00001111</li> </ul>	1 AO1 1b(1)	Ignore missing or additional leading zeros
4	f	i	1 mark per bullet point <ul style="list-style-type: none"> <li>B AND C</li> <li>OR gate with two inputs, one of which is A</li> <li>...correct connection of these two gates with no additional gates / connections</li> </ul>	3 AO1 1b(3)	Shape must be accurate  

4	f	ii	<div>1 mark per bullet point</div> <ul style="list-style-type: none"><li>• Correct completion of A and B inputs as <b>1 1</b></li><li>• 0 output for 01 input</li><li>• 0 output for 10 input</li><li>• 0 output for 11 input</li></ul>	4  AO1 1b(1) AO2 1b(3)	CAO <table><tr><th>A</th><th>B</th><th>P</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	P	0	0	1	0	1	0	1	0	0	1	1	0
A	B	P																		
0	0	1																		
0	1	0																		
1	0	0																		
1	1	0																		

Question			Answer	Mark	Guidance
5	a		<ul style="list-style-type: none"> <li>Number of pixels (in an image)</li> <li>Height <u>and</u> width (of an image)</li> </ul>	1  AO2 1b(1)	Accept pixels per inch / mm / unit area (density)
5	b		<ul style="list-style-type: none"> <li>90 (pixels in an image) // 15 x 6 (pixels in image)</li> <li>Multiply pixels x bits per pixel</li> <li>...2 bits required per pixel (because 3 colours)</li> <li><b>180</b> bits overall answer</li> </ul>	4  AO1 1b(2) AO1 1b(2)	Must clearly show multiplication for 3 <sup>rd</sup> BP
5	c		<ul style="list-style-type: none"> <li>Reduce number of pixels / resolution</li> <li>Reduce number of colours</li> <li>Use lossy compression</li> <li>Use lossless compression</li> </ul>	2  AO2 1a(2)	Accept descriptive answers linked to given logo (e.g "change to black and white only") "Make image smaller" is NE  Allow compression by itself for one answer.
5	d	i	<ul style="list-style-type: none"> <li>Data <u>about</u> data / the image/file // properties of the file</li> </ul>	1  AO1 1b(2)	Do not accept examples without a definition.
5	d	li	e.g. <ul style="list-style-type: none"> <li>height</li> <li>width</li> <li>colour depth</li> <li>resolution</li> <li>geolocation</li> <li>date/time <b>created/last edited</b> // timestamp</li> <li>file type</li> <li>author details</li> </ul>	1  AO1 1a(2)	Accept any sensible data that could be stored alongside an image.  Do not accept filename
6	(a)		<ul style="list-style-type: none"> <li>Access "<u>Rob</u>" / <u>studentnames[0]...</u></li> <li>...does <b>not</b> equal "Anna" // not desired item // move on</li> <li>Access "<u>Anna</u>" / <u>studentnames[1]</u></li> <li>...does equal "Anna" // stop // item found</li> </ul>	4  AO2 1b(4)	Answer must refer to this array, not a generic description of linear search. "Access first item" is NE for BP1 or BP3. Must refer to this scenario.  Max 1 for "Compare ' <b>Anna</b> ' to <b>each item</b> in list" if nothing else credited.

6	(b)		<ul style="list-style-type: none"> <li>• Anna inserted before Rob as first two elements...</li> <li>• ...Huw correctly inserted into sorted list...</li> <li>• ...Emma correctly inserted into sorted list ...</li> <li>• ...Patrice correctly inserted into sorted list ...</li> <li>• ...Iqbal correctly inserted into sorted list and <b>no further changes</b> made.</li> </ul>	5 AO2 1b(5)	<table border="1"> <tr> <td>Rob</td><td>Anna</td><td>Huw</td><td>Emma</td><td>Patrice</td><td>Iqbal</td></tr> <tr> <td>Anna</td><td>Rob</td><td>Huw</td><td>Emma</td><td>Patrice</td><td>Iqbal</td></tr> <tr> <td>Anna</td><td>Huw</td><td>Rob</td><td>Emma</td><td>Patrice</td><td>Iqbal</td></tr> <tr> <td>Anna</td><td>Emma</td><td>Huw</td><td>Rob</td><td>Patrice</td><td>Iqbal</td></tr> <tr> <td>Anna</td><td>Emma</td><td>Huw</td><td>Patrice</td><td>Rob</td><td>Iqbal</td></tr> <tr> <td>Anna</td><td>Emma</td><td>Huw</td><td>Iqbal</td><td>Patrice</td><td>Rob</td></tr> </table> <p>Sorted list highlighted</p>	Rob	Anna	Huw	Emma	Patrice	Iqbal	Anna	Rob	Huw	Emma	Patrice	Iqbal	Anna	Huw	Rob	Emma	Patrice	Iqbal	Anna	Emma	Huw	Rob	Patrice	Iqbal	Anna	Emma	Huw	Patrice	Rob	Iqbal	Anna	Emma	Huw	Iqbal	Patrice	Rob
Rob	Anna	Huw	Emma	Patrice	Iqbal																																				
Anna	Rob	Huw	Emma	Patrice	Iqbal																																				
Anna	Huw	Rob	Emma	Patrice	Iqbal																																				
Anna	Emma	Huw	Rob	Patrice	Iqbal																																				
Anna	Emma	Huw	Patrice	Rob	Iqbal																																				
Anna	Emma	Huw	Iqbal	Patrice	Rob																																				
6	(c)		<ul style="list-style-type: none"> <li>• Use of iteration (any use) ...</li> <li>• ...loops for each item in array // loops 6 times</li> <li>• ...to print out each item in <code>studentnames</code></li> <li>• ...input attendance</li> <li>• Add up/calculate students present <b>and</b> absent</li> <li>• ...Outputs present <b>and</b> absent (in suitable message)</li> </ul>	6 AO3 2b(6)	<p>BP 2 and 3 may be met together with suitable input statement. Both dependent on attempt at iteration.</p> <p>BP5 not dependent on correct previous parts.</p> <p>BP6 needs reasonable attempt at totalling present and absent figures.</p> <p>Ignore non-initialisation of counter variables.</p> <p>Flowcharts are acceptable but must show <b>how</b> to solve the problem, not simply repeat the question.</p> <p><u>Example algorithm</u></p> <pre> present=0 absent=0 for i = 0 to (studentnames.length) -1     print(studentnames[i])     attendance=input("absent or present?")     if attendance=="present" then         present=present+1     else         absent=absent+1     endif next i print ("Present students: " + present) print ("Absent students: " + absent) </pre>																																				

**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](mailto:general.qualifications@ocr.org.uk)

[www.ocr.org.uk](http://www.ocr.org.uk)

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