



Cambridge International AS & A Level

COMPUTER SCIENCE**9618/21**

Paper 21 Fundamental Problem Solving & Programming Skills

May/June 2022

MARK SCHEME

Maximum Mark: 75

Published

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the May/June 2022 series for most Cambridge IGCSE, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

This document consists of **12** printed pages.

Generic Marking Principles

These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptors for a question. Each question paper and mark scheme will also comply with these marking principles.

GENERIC MARKING PRINCIPLE 1:

Marks must be awarded in line with:

- the specific content of the mark scheme or the generic level descriptors for the question
- the specific skills defined in the mark scheme or in the generic level descriptors for the question
- the standard of response required by a candidate as exemplified by the standardisation scripts.

GENERIC MARKING PRINCIPLE 2:

Marks awarded are always **whole marks** (not half marks, or other fractions).

GENERIC MARKING PRINCIPLE 3:

Marks must be awarded **positively**:

- marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate
- marks are awarded when candidates clearly demonstrate what they know and can do
- marks are not deducted for errors
- marks are not deducted for omissions
- answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous.

GENERIC MARKING PRINCIPLE 4:

Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.

GENERIC MARKING PRINCIPLE 5:

Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen).

GENERIC MARKING PRINCIPLE 6:

Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind.

Question	Answer	Marks															
1(a)	An algorithm	1															
1(b)(i)	<table border="1"> <thead> <tr> <th>Variable</th> <th>Use of variable</th> <th>Data type</th> </tr> </thead> <tbody> <tr> <td>Temp</td> <td>Stores the average temperature</td> <td>REAL</td> </tr> <tr> <td>PetName</td> <td>Stores the name of my pet</td> <td>STRING</td> </tr> <tr> <td>MyDOB</td> <td>To calculate how many days until my next birthday</td> <td>DATE</td> </tr> <tr> <td>LightOn</td> <td>Stores state of light; light is only on or off</td> <td>BOOLEAN</td> </tr> </tbody> </table> <p>One mark for each data type</p>	Variable	Use of variable	Data type	Temp	Stores the average temperature	REAL	PetName	Stores the name of my pet	STRING	MyDOB	To calculate how many days until my next birthday	DATE	LightOn	Stores state of light; light is only on or off	BOOLEAN	4
Variable	Use of variable	Data type															
Temp	Stores the average temperature	REAL															
PetName	Stores the name of my pet	STRING															
MyDOB	To calculate how many days until my next birthday	DATE															
LightOn	Stores state of light; light is only on or off	BOOLEAN															
1(b)(ii)	<p>One mark for variable name, and one for reason</p> <p>Variable: Temp</p> <p>Reason: Name does not indicate what the variable is used for</p>	2															
1(c)	<table border="1"> <thead> <tr> <th>Expression</th> <th>Evaluation</th> </tr> </thead> <tbody> <tr> <td>INT((31 / 3) + 1)</td> <td>11</td> </tr> <tr> <td>MID(TO_UPPER("Version"), 4, 2)</td> <td>"SI"</td> </tr> <tr> <td>TRUE AND (NOT FALSE)</td> <td>TRUE</td> </tr> <tr> <td>NUM_TO_STR(27 MOD 3)</td> <td>"0"</td> </tr> </tbody> </table> <p>One mark per row</p>	Expression	Evaluation	INT((31 / 3) + 1)	11	MID(TO_UPPER("Version"), 4, 2)	"SI"	TRUE AND (NOT FALSE)	TRUE	NUM_TO_STR(27 MOD 3)	"0"	4					
Expression	Evaluation																
INT((31 / 3) + 1)	11																
MID(TO_UPPER("Version"), 4, 2)	"SI"																
TRUE AND (NOT FALSE)	TRUE																
NUM_TO_STR(27 MOD 3)	"0"																

Question	Answer	Marks								
2(a)	<p>One mark per row</p> <table border="1"> <thead> <tr> <th></th> <th>Answer</th> </tr> </thead> <tbody> <tr> <td>The number of different inputs</td> <td>3</td> </tr> <tr> <td>The number of different outputs</td> <td>3</td> </tr> <tr> <td>The single input value that could result in S4</td> <td>Button-Y</td> </tr> </tbody> </table>		Answer	The number of different inputs	3	The number of different outputs	3	The single input value that could result in S4	Button-Y	3
	Answer									
The number of different inputs	3									
The number of different outputs	3									
The single input value that could result in S4	Button-Y									

Question	Answer	Marks															
2(b)	<p>One mark per row</p> <p>Example answer</p> <table border="1"> <thead> <tr> <th>Input</th> <th>Output</th> <th>Next state</th> </tr> </thead> <tbody> <tr> <td>Button-Y</td> <td>none</td> <td>S3</td> </tr> <tr> <td>Button-Y</td> <td>none</td> <td>S4</td> </tr> <tr> <td>Button-Z</td> <td>Output-B</td> <td>S2</td> </tr> <tr> <td>Button-Z</td> <td>none</td> <td>S1</td> </tr> </tbody> </table> <p>Note: Accept other valid answers</p>	Input	Output	Next state	Button-Y	none	S3	Button-Y	none	S4	Button-Z	Output-B	S2	Button-Z	none	S1	4
Input	Output	Next state															
Button-Y	none	S3															
Button-Y	none	S4															
Button-Z	Output-B	S2															
Button-Z	none	S1															

Question	Answer	Marks
3(a)	<p>One mark per description of appropriate sub-problem for given scenario.</p> <p>Examples include:</p> <ul style="list-style-type: none"> Allows the user to search for films being shown // input name of film they want to see Allows the user to search for available seats Calculate cost of booking Book a given number of seats for a particular screening 	3
3(b)	Function	1

Question	Answer	Marks						
4(a)	<p>One mark per row</p> <table border="1"> <thead> <tr> <th colspan="2">Answer</th> </tr> </thead> <tbody> <tr> <td>The value that has been on the stack for the longest time.</td> <td>'H'</td> </tr> <tr> <td>The memory location pointed to by <code>TopOfStack</code> if three POP operations are performed.</td> <td>206</td> </tr> </tbody> </table>	Answer		The value that has been on the stack for the longest time.	'H'	The memory location pointed to by <code>TopOfStack</code> if three POP operations are performed.	206	2
Answer								
The value that has been on the stack for the longest time.	'H'							
The memory location pointed to by <code>TopOfStack</code> if three POP operations are performed.	206							

Question	Answer	Marks																														
4(b)	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="2">Stack</th> <th>Pointer</th> </tr> <tr> <th>Memory location</th> <th>Value</th> <th></th> </tr> </thead> <tbody> <tr> <td>200</td> <td></td> <td></td> </tr> <tr> <td>201</td> <td>'D'</td> <td>← TopOfStack</td> </tr> <tr> <td>202</td> <td>'C'</td> <td></td> </tr> <tr> <td>203</td> <td>'A'</td> <td></td> </tr> <tr> <td>204</td> <td>'X'</td> <td></td> </tr> <tr> <td>205</td> <td>'Z'</td> <td></td> </tr> <tr> <td>206</td> <td>'N'</td> <td></td> </tr> <tr> <td>207</td> <td>'P'</td> <td></td> </tr> </tbody> </table> <p>One mark for:</p> <ol style="list-style-type: none"> 1 TopOfStack pointing to 'D' 2 Value 'D' in 201 3 Values 'C' & 'A' in 202 and 203 4 Values 'X' to 'P' unchanged (204 to 207) 	Stack		Pointer	Memory location	Value		200			201	'D'	← TopOfStack	202	'C'		203	'A'		204	'X'		205	'Z'		206	'N'		207	'P'		4
Stack		Pointer																														
Memory location	Value																															
200																																
201	'D'	← TopOfStack																														
202	'C'																															
203	'A'																															
204	'X'																															
205	'Z'																															
206	'N'																															
207	'P'																															

Question	Answer	Marks
5	<p>One mark per point to Max 6</p> <ol style="list-style-type: none"> 1 Open file in read mode 2 Set up a conditional loop, repeating until the value is found or the EOF () is reached 3 Read a line from the file in a loop 4 Extract Field 2 5 Description of how Field 2 could be extracted e.g. using substring function and lengths of Field 1 and Field 2 6 Compare extracted field with search value 7 If search value found, extract Field 1 and Field 3 and output them 8 Close the file after loop has finished 	6

Question	Answer	Marks
6(a)	<p>Simple Solution:</p> <pre> DECLARE ThisInt, Count : INTEGER Count ← 0 FOR ThisInt ← 100 TO 200 IF ThisInt MOD 10 = 7 THEN OUTPUT ThisInt Count ← Count + 1 ENDIF NEXT ThisInt OUTPUT Count </pre> <p>Mark as follows:</p> <ol style="list-style-type: none"> 1 Declare loop variable and counter as integers, counter initialised 2 Loop 100 to 200, no step defined 3 Test value in a loop 4 Output selected value and incrementing a counter in a loop 5 Output the counter, following a reasonable attempt, after the loop <p>Alternative Solution:</p> <pre> DECLARE ThisInt, Count : INTEGER Count ← 0 FOR ThisInt ← 107 TO 197 STEP 10 OUTPUT ThisInt Count ← Count + 1 NEXT ThisInt OUTPUT Count </pre> <p>Mark as follows:</p> <ol style="list-style-type: none"> 1 Declare loop variable and counter as integers, , counter initialised 2 Loop (107 to 197) 3 STEP 10 or explicit increment if conditional loop used 4 Output each value and incrementing a counter in a loop 5 Output the counter, following a reasonable attempt, after the loop 	5

Question	Answer	Marks
6(b)	<pre>IF MySwitch = 1 THEN ThisChar ← 'a' ELSE IF MySwitch = 2 THEN ThisChar ← 'y' ELSE IF MySwitch = 3 THEN ThisChar ← '7' ELSE ThisChar ← '*' ENDIF ENDIF ENDIF</pre> <p>Mark as follows:</p> <ol style="list-style-type: none">1. ANY test of MySwitch = 1, 2 or 32. All three comparisons and corresponding assignments3. OTHERWISE, or initial assignment of default value4. Completely correct IF...THEN...ELSE...ENDIF syntax	4

Question	Answer	Marks
7(a)	<pre> FUNCTION IsPalindrome(InString : STRING) RETURNS BOOLEAN DECLARE IsPal : BOOLEAN DECLARE Index, Num : INTEGER DECLARE CharA, CharB : CHAR IsPal ← TRUE Index ← 1 Num ← INT(LENGTH(InString) / 2) WHILE Index <= Num AND IsPal = TRUE CharA ← MID(InString, Index, 1) CharB ← MID(Instring, LENGTH(Instring) - Index + 1, 1) IF UCASE(CharA) <> UCASE(CharB) THEN IsPal ← FALSE // RETURN FALSE ENDIF Index ← Index + 1 ENDWHILE RETURN IsPal // RETURN TRUE ENDFUNCTION </pre> <p>Mark as follows:</p> <ol style="list-style-type: none"> 1 Functions header including parameter, ending and return type 2 Calculation of number of pairs to match (length or half length) 3 Loop for half or whole string 4 ...Extracting characters to compare // create reverse string 5 Convert characters to same case 6 Check for_mismatch of characters inside loop / test for mismatch after loop for reversed string 7 Returning Boolean in both cases 	7

Question	Answer	Marks																
7(b)	<table border="1" data-bbox="320 248 1310 817"> <thead> <tr> <th data-bbox="320 248 435 315">Label</th> <th data-bbox="435 248 1310 315">Text</th> </tr> </thead> <tbody> <tr> <td data-bbox="320 315 435 383">A</td> <td data-bbox="435 315 1310 383">Set OutString to " "</td> </tr> <tr> <td data-bbox="320 383 435 450">B</td> <td data-bbox="435 383 1310 450">Is Index > LENGTH(InString)?</td> </tr> <tr> <td data-bbox="320 450 435 517">C</td> <td data-bbox="435 450 1310 517">Is MID(InString, Index, 1) = " "?</td> </tr> <tr> <td data-bbox="320 517 435 618">D</td> <td data-bbox="435 517 1310 618">Set OutString to OutString & MID(InString, Index, 1)</td> </tr> <tr> <td data-bbox="320 618 435 685">E</td> <td data-bbox="435 618 1310 685">Set Index to Index + 1</td> </tr> <tr> <td data-bbox="320 685 435 752">F</td> <td data-bbox="435 685 1310 752">YES</td> </tr> <tr> <td data-bbox="320 752 435 817">G</td> <td data-bbox="435 752 1310 817">NO</td> </tr> </tbody> </table> <p data-bbox="304 853 528 887">Mark for each of:</p> <ul data-bbox="304 889 496 1025" style="list-style-type: none"> • B • D • C • ...F and G <p data-bbox="304 1061 1238 1095">Note: The mark for F and G is dependent on a reasonable attempt at C</p>	Label	Text	A	Set OutString to " "	B	Is Index > LENGTH(InString)?	C	Is MID(InString, Index, 1) = " "?	D	Set OutString to OutString & MID(InString, Index, 1)	E	Set Index to Index + 1	F	YES	G	NO	4
Label	Text																	
A	Set OutString to " "																	
B	Is Index > LENGTH(InString)?																	
C	Is MID(InString, Index, 1) = " "?																	
D	Set OutString to OutString & MID(InString, Index, 1)																	
E	Set Index to Index + 1																	
F	YES																	
G	NO																	

Question	Answer	Marks
8(a)	<pre> FUNCTION RandomChar() RETURNS CHAR DECLARE ThisRange : INTEGER DECLARE ThisChar : CHAR //First select the range ThisRange ← INT(RAND(3)) + 1 // 1 to 3 CASE OF ThisRange 1: ThisChar ← CHR(INT(RAND(26) + 65)) // 65 to 90: 'A' to 'Z' ThisChar ← LCASE(ThisChar) // 'a' to 'z' 2: ThisChar ← CHR(INT(RAND(26) + 65)) // 65 to 90: A to Z 3: ThisChar ← NUM_TO_STR(INT(RAND(10)) // '0' to '9' ENDCASE RETURN ThisChar ENDFUNCTION </pre> <p>Mark as follows:</p> <ol style="list-style-type: none"> 1 Generation of any integer random number 2 Randomly decide which of the three ranges to select 3 Selection structure based on range 4 One alphanumeric character range correct 5 All alphanumeric character ranges correct 6 Return <code>ThisChar</code>, following a reasonable attempt to generate a character in each range 	6

Question	Answer	Marks
8(b)	<pre> FUNCTION FindPassword(Name: STRING) RETURNS STRING DECLARE Index : INTEGER DECLARE Password : STRING Password ← "" Index ← 1 WHILE Password = "" AND Index <= 500 IF Secret[Index, 1] = Name THEN Password ← Decrypt(Secret[Index, 2]) ELSE Index ← Index + 1 ENDIF ENDWHILE IF Password = "" THEN OUTPUT "Domain name not found" ENDIF RETURN Password ENDFUNCTION </pre> <p>Mark as follows:</p> <ol style="list-style-type: none"> 1 Declare all local variables used, attempted solution has to be reasonable 2 Conditional loop while not found and not end of array 3 Compare value of element in column 1 with parameter passed into function 4 ...and use Decrypt () with element in column 2 as parameter 5 ...use the return value of Decrypt () 6 Output warning message if parameter not found 7 Return STRING value 	7
8(c)	<p>One mark for the name, one for the description</p> <p>Name:</p> <ul style="list-style-type: none"> • Stub testing <p>Description:</p> <ul style="list-style-type: none"> • A simple module is written to replace each of the modules. • The simple module will return an expected value // will output a message to show they have been called 	3
8(d)	<p>Accept one example of a valid password to Max 2</p> <p>One mark for each password example that breaks one of the rules due to:</p> <ul style="list-style-type: none"> • Length too long // length too short • Invalid character • Incorrect grouping (including number of hyphens) • Duplicated characters 	2

Question	Answer	Marks
8(e)	One mark for each part: <ul style="list-style-type: none">• Generate a random integer divisible by 3• Split range into 1/3 and set as numeric• Else alphabetic character	3