
COMPUTER SCIENCE

9608/22

Paper 2 Written Paper

May/June 2017

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 will not enter into discussions about these mark schemes.

Cambridge is publishing the mark schemes for the May/June 2017 series for most Cambridge IGCSE[®], Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

© IGCSE is a registered trademark.

This document consists of **8** printed pages.

Question	Answer					Marks																									
1(a)	<table border="1" data-bbox="252 248 1378 546"> <thead> <tr> <th data-bbox="252 248 352 300">Item</th> <th data-bbox="357 248 1043 300">Statement</th> <th data-bbox="1048 248 1133 300">Input</th> <th data-bbox="1137 248 1262 300">Process</th> <th data-bbox="1267 248 1378 300">Output</th> </tr> </thead> <tbody> <tr> <td data-bbox="252 306 352 358">1</td> <td data-bbox="357 306 1043 358">SomeChars = "Hello World"</td> <td data-bbox="1048 306 1133 358"></td> <td data-bbox="1137 306 1262 358">✓</td> <td data-bbox="1267 306 1378 358"></td> </tr> <tr> <td data-bbox="252 365 352 416">2</td> <td data-bbox="357 365 1043 416">OUTPUT RIGHT(String1,5)</td> <td data-bbox="1048 365 1133 416"></td> <td data-bbox="1137 365 1262 416">✓</td> <td data-bbox="1267 365 1378 416">✓</td> </tr> <tr> <td data-bbox="252 423 352 474">3</td> <td data-bbox="357 423 1043 474">READFILE (MyFile, String2)</td> <td data-bbox="1048 423 1133 474">✓</td> <td data-bbox="1137 423 1262 474"></td> <td data-bbox="1267 423 1378 474"></td> </tr> <tr> <td data-bbox="252 481 352 533">4</td> <td data-bbox="357 481 1043 533">WRITEFILE (MyFile, "Data is " & String2)</td> <td data-bbox="1048 481 1133 533"></td> <td data-bbox="1137 481 1262 533">✓</td> <td data-bbox="1267 481 1378 533">✓</td> </tr> </tbody> </table> <p data-bbox="252 577 464 613">Mark as follows:</p> <p data-bbox="252 645 472 680">Row 1 as shown</p> <p data-bbox="252 680 1094 716">Row 2 no marks if tick in Input column, otherwise 1 mark per tick</p> <p data-bbox="252 716 472 752">Row 3 as shown</p> <p data-bbox="252 752 1094 788">Row 4 no marks if tick in Input column, otherwise 1 mark per tick</p>					Item	Statement	Input	Process	Output	1	SomeChars = "Hello World"		✓		2	OUTPUT RIGHT(String1,5)		✓	✓	3	READFILE (MyFile, String2)	✓			4	WRITEFILE (MyFile, "Data is " & String2)		✓	✓	6
Item	Statement	Input	Process	Output																											
1	SomeChars = "Hello World"		✓																												
2	OUTPUT RIGHT(String1,5)		✓	✓																											
3	READFILE (MyFile, String2)	✓																													
4	WRITEFILE (MyFile, "Data is " & String2)		✓	✓																											
1(b)(i)	<ul data-bbox="252 815 1007 884" style="list-style-type: none"> • Integer / Real / Single / Double / Floating Point / Float • Boolean 					2																									
1(b)(ii)	<table border="1" data-bbox="252 916 1091 1200"> <thead> <tr> <th data-bbox="252 916 807 967">Expression</th> <th data-bbox="812 916 1091 967">Evaluates to</th> </tr> </thead> <tbody> <tr> <td data-bbox="252 974 807 1048">(FlagA AND FlagB) OR FlagC</td> <td data-bbox="812 974 1091 1048">TRUE</td> </tr> <tr> <td data-bbox="252 1055 807 1128">FlagA AND (FlagB OR FlagC)</td> <td data-bbox="812 1055 1091 1128">TRUE</td> </tr> <tr> <td data-bbox="252 1135 807 1200">(NOT FlagA) OR (NOT FlagC)</td> <td data-bbox="812 1135 1091 1200">FALSE</td> </tr> </tbody> </table> <p data-bbox="252 1234 499 1270">1 mark per answer</p>					Expression	Evaluates to	(FlagA AND FlagB) OR FlagC	TRUE	FlagA AND (FlagB OR FlagC)	TRUE	(NOT FlagA) OR (NOT FlagC)	FALSE	3																	
Expression	Evaluates to																														
(FlagA AND FlagB) OR FlagC	TRUE																														
FlagA AND (FlagB OR FlagC)	TRUE																														
(NOT FlagA) OR (NOT FlagC)	FALSE																														
1(c)	<pre data-bbox="252 1301 687 1509">MyCount ← 101 REPEAT OUTPUT MyCount MyCount ← MyCount + 2 UNTIL MyCount > 199</pre> <p data-bbox="252 1541 671 1576">1 mark for each of the following:</p> <ul data-bbox="252 1608 991 1749" style="list-style-type: none"> • Counter initialisation • Repeat ... Until loop • Method for choosing (correct range of) odd numbers • Output all odd numbers in the range <p data-bbox="252 1780 887 1816">Note: Counter variable name must be consistent</p>					4																									

Question	Answer	Marks
2(a)	<ul style="list-style-type: none"> • to <u>increase the level of detail of an algorithm / design...</u> // breaking down a <u>problem / module / task</u> into smaller parts... • ...from which the task may be <u>programmed</u> <p>1 mark per underlined phrase or equivalent</p>	2
2(b)	<p>1 mark for first 3 data types – String 1 mark for last data type – Boolean</p> <p>1 mark for each description:</p> <pre> FileUserID Stores (User) ID from <u>file</u> FilePreferredName Stores (preferred) name from <u>file</u> IDFoundFlag True if (User) ID found in <u>file</u> // False if (User) ID not found in <u>file</u> // If SearchUserID matches FileUserID </pre>	5
2(c)	<ol style="list-style-type: none"> 1. LOOP through the file until EOF()... 2. OR SearchUserId is found 3. READ text line from UserNames.txt file in a loop 4. EXTRACT FileUserID in a loop 5. IF SearchUserId matches FileUserID THEN in a loop 6. SET FilePreferredName to the name from the file 7. Check if User ID found not in a loop 8. OUTPUT appropriate message for both conditions <p>1 mark per functional equivalent of each numbered statement.</p>	Max 8

Question	Answer	Marks
3	<pre> FUNCTION ExCamel (<u>InString</u>: STRING) RETURNS <u>STRING</u> DECLARE <u>NextChar</u> : <u>CHAR</u> DECLARE <u>OutString</u> : STRING DECLARE n : INTEGER <u>OutString</u> ← "" // initialise the return string // loop through InString to produce OutString FOR n ← 1 TO <u>LENGTH(InString)</u> // from first to last <u>NextChar</u> ← <u>MID(InString, n, 1)</u> // get next character IF <u>NextChar</u> >= 'A' AND <u>NextChar</u> <= 'Z' // check if upper case // <u>NextChar</u> = UCASE(<u>NextChar</u>) THEN IF n > 1 // if not first character THEN <u>OutString</u> ← <u>OutString & " "</u> // add space to OutString ENDIF <u>NextChar</u> ← <u>LCASE(NextChar)</u> // make NextChar lower case ENDIF <u>OutString</u> ← <u>OutString & NextChar</u> // add Nextchar to OutString ENDFOR RETURN <u>OutString</u> // return value ENDFUNCTION </pre> <p>1 mark per underlined word / expression</p>	Max 11

Question	Answer	Marks									
4(a)	<ul style="list-style-type: none"> • Functions • Procedures • Global / Local variables <p>1 mark per item</p>	Max 2									
4(b)	<table border="1"> <thead> <tr> <th>Name of parameter passing method</th> <th>Value output</th> <th>Explanation</th> </tr> </thead> <tbody> <tr> <td>(Call) by reference</td> <td>5</td> <td> <ul style="list-style-type: none"> • The <u>address</u> of the variable is passed. • <u>Original value is changed</u> when parameter changed in called module. </td> </tr> <tr> <td>(Call) by value</td> <td>4</td> <td> <ul style="list-style-type: none"> • A <u>copy</u> of the variable itself is passed. • <u>Original value not changed</u> when parameter changed in called module. </td> </tr> </tbody> </table> <p>Mark as follows:</p> <ul style="list-style-type: none"> • 1 mark for each name and value • 1 mark per bullet in explanation 	Name of parameter passing method	Value output	Explanation	(Call) by reference	5	<ul style="list-style-type: none"> • The <u>address</u> of the variable is passed. • <u>Original value is changed</u> when parameter changed in called module. 	(Call) by value	4	<ul style="list-style-type: none"> • A <u>copy</u> of the variable itself is passed. • <u>Original value not changed</u> when parameter changed in called module. 	6
Name of parameter passing method	Value output	Explanation									
(Call) by reference	5	<ul style="list-style-type: none"> • The <u>address</u> of the variable is passed. • <u>Original value is changed</u> when parameter changed in called module. 									
(Call) by value	4	<ul style="list-style-type: none"> • A <u>copy</u> of the variable itself is passed. • <u>Original value not changed</u> when parameter changed in called module. 									

Question	Answer	Marks
5(a)(i)	<ul style="list-style-type: none"> • Any character <u>except</u> colon, space or any alpha-numeric • Reason: character is not in the login information strings 	2
5(a)(ii)	<pre>DECLARE <u>LogArray</u> : ARRAY[1 : 20] OF <u>STRING</u></pre> <p>1 mark per underline</p>	2

Question	Answer	Marks
5(b)	<p>Pseudocode solution included here for development and clarification of mark scheme. Programming language example solutions appear in the Appendix.</p> <pre> PROCEDURE LogEvents() DECLARE FileData : STRING DECLARE ArrayIndex : INTEGER OPENFILE "LoginFile.txt" FOR APPEND FOR ArrayIndex ← 1 TO 20 // IF LogArray[ArrayIndex] <> "****" THEN FileData ← LogArray[ArrayIndex] WRITEFILE ("LoginFile.txt", FileData) ENDIF ENDFOR CLOSEFILE("LoginFile.txt") ENDPROCEDURE </pre> <p>1 mark for each of the following:</p> <ol style="list-style-type: none"> 1. Procedure heading and ending 2. Declare ArrayIndex as integer // commented in python 3. Open file 'LoginFile' for append 4. Correct loop 5. extract data from array in a loop 6. check for unused element in a loop 7. write data to file in a loop 8. Close the file outside the loop 	8

Question	Answer	Marks
6(a)	<p>Pseudocode solution included here for development and clarification of mark scheme. Programming language example solutions appear in the Appendix.</p> <pre> FUNCTION ValidateRegistration(Registration : STRING) RETURNS BOOLEAN DECLARE UCaseChar, NumChar : INTEGER DECLARE NextChar : CHAR DECLARE ReturnFlag : BOOLEAN DECLARE n : INTEGER ReturnFlag ← TRUE ValidateRegistration ← True IF LEN(Registration) < 6 OR LEN(Registration) > 9 //check length THEN ReturnFlag ← False ELSE FOR n ← 1 TO 3 //check for 3 upper case alpha NextChar ← MID(Registration, n, 1) IF NextChar < 'A' AND NextChar > 'Z' THEN ReturnFlag ← False ENDIF ENDFOR FOR n ← 4 TO 5 //check for 2 numeric NextChar ← MID(Registration, n, 1) IF NextChar < '0' AND NextChar > '9' THEN ReturnFlag ← False ENDIF ENDFOR FOR n ← 6 TO LEN(Registration) //check remaining characters NextChar ← MID(Registration, n, 1) IF NextChar < 'A' AND NextChar > 'Z' THEN ReturnFlag ← False ENDIF ENDFOR ENDIF RETURN (ReturnFlag) ENDFUNCTION </pre>	Max 9

Question	Answer	Marks
6(a)	<p>1 mark for each of the following:</p> <ol style="list-style-type: none"> 1. Correct Function heading and ending 2. Check for correct length 3. Extract first three characters 4. Check first three characters are capitals 5. Extract characters four and five 6. Check characters four and five are numeric 7. Extract remaining characters 8. Check remaining characters are capitals 9. Combine all four tests results into a single Boolean value 10. Return a Boolean value 	
6(b)	<p>String1: (for example, "ABC12XYZ")</p> <p>One mark for a valid string having:</p> <ul style="list-style-type: none"> • Correct length (between 6 and 9 characters) • 3 capital letters followed by... • 2 numeric characters followed by... • between 1 and 4 capital letters <p>String2 to String5:</p> <p>1 mark for each string and explanation (testing different rules of the function)</p> <p>Test strings breaking one different rules:</p> <ul style="list-style-type: none"> • Incorrect length • With incorrect number of capital letters at the start • With non-numeric characters in positions 4 and 5 • With incorrect number of capital letters at the end • Containing an invalid character (not alpha-numeric) 	5