Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

COMPUTER SCIENCE **9608/22**

Paper 2 Written Paper **May/June 2017**

MARK SCHEME

Maximum Mark: 75

Published

Question	Answer	Marks
1(a)		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)`		✓	✓

Mark as follows:

Row 1 as shown
Row 2 no marks if tick in Input column, otherwise 1 mark per tick
Row 3 as shown
Row 4 no marks if tick in Input column, otherwise 1 mark per tick

Question	Answer	Marks
1(b)(i)	• Integer / Real / Single / Double / Floating Point / Float • Boolean	2
1(b)(ii)		3

Expression	Evaluates to
`(FlagA AND FlagB) OR FlagC`	**TRUE**
`FlagA AND (FlagB OR FlagC)`	**TRUE**
`(NOT FlagA) OR (NOT FlagC)`	**FALSE**

1 mark per answer

Question	Answer	Marks
1(c)		4

```
MyCount ← 101

REPEAT
   OUTPUT MyCount
   MyCount ← MyCount + 2
UNTIL MyCount > 199
```

1 mark for each of the following:

- Counter initialisation
- Repeat … Until loop
- Method for choosing (correct range of) odd numbers
- Output all odd numbers in the range

Note: Counter variable name must be consistent

Question	Answer	Marks
2(a)	• to increase the level of detail of an algorithm / design... // breaking down a problem / module / task into smaller parts… • …from which the task may be programmed 1 mark per underlined phrase or equivalent	**2**
2(b)	1 mark for first 3 data types – String 1 mark for last data type – Boolean 1 mark for each description: `FileUserID` Stores (User) ID from file `FilePreferredName` Stores (preferred) name from file `IDFoundFlag` True if (User) ID found in file // False if (User) ID not found in file // If `SearchUserID` matches `FileUserID`	**5**
2(c)	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 1 mark per functional equivalent of each numbered statement.	**Max 8**

Question	Answer	Marks
3		**Max 11**

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

1 mark per underlined word / expression

Question	Answer	Marks
4(a)	• Functions • Procedures • Global / Local variables 1 mark per item	**Max 2**
4(b)	**Name of parameter passing method** / **Value output** / **Explanation**: (Call) by reference — 5 — • The <u>address of</u> the variable is passed. • <u>Original value is changed</u> when parameter changed in called module. (Call) by value — 4 — • A <u>copy of</u> the variable itself is passed. • <u>Original value not changed</u> when parameter changed in called module. Mark as follows: • 1 mark for each name **and** value • 1 mark per bullet in explanation	**6**

Table for 4(b):

Name of parameter passing method	Value output	Explanation
(Call) by reference	5	• The <u>address of</u> the variable is passed. • <u>Original value is changed</u> when parameter changed in called module.
(Call) by value	4	• A <u>copy of</u> the variable itself is passed. • <u>Original value not changed</u> when parameter changed in called module.

Question	Answer	Marks
5(a)(i)	• Any character <u>except</u> colon, space or any alpha-numeric • Reason: character is not in the login information strings	**2**
5(a)(ii)	`DECLARE` <u>`LogArray : ARRAY[1 : 20]`</u> `OF` <u>`STRING`</u> 1 mark per underline	**2**

Question	Answer	Marks

5(b)	Pseudocode solution included here for development and clarification of mark scheme.
	Programming language example solutions appear in the **Appendix**.
	
	`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`
	
	1 mark for each of the following:
	
	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)	Pseudocode solution included here for development and clarification of mark scheme.
	Programming language example solutions appear in the Appendix.
	
	`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`
		Max 9

Question	Answer	Marks
6(a)	1 mark for each of the following: 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)	**String1:** (for example, "ABC12XYZ") One mark for a valid string having: • Correct length (between 6 and 9 characters) • 3 capital letters followed by… • 2 numeric characters followed by… • between 1 and 4 capital letters **String2 to String5:** 1 mark for each string **and** explanation (testing different rules of the function) Test strings breaking **one different** rules: • 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**