
COMPUTER SCIENCE

9608/23

Paper 2 Written Paper

October/November 2019

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 October/November 2019 series for most Cambridge IGCSE™, Cambridge International A and AS Level components and some Cambridge O Level components.

This document consists of **13** 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)(i)	One mark for each (different) data type and one mark for a corresponding example value Acceptable types: Integer, Real, String, Char, Boolean, Date,	6								
1(a)(ii)	Declaration	1								
1(b)	Two from (max 2): <ul style="list-style-type: none">A list of identifier / variable namesExplanations/descriptions (of what they are used for)Data types	2								
1(c)(i)	One mark for each stage (Input, Output) One mark for each correct example <table><tr><th>Stage</th><th>Example statement</th></tr><tr><td>Input</td><td><u>Next = Console.ReadLine()</u></td></tr><tr><td>Process</td><td><u>x = INT(y/3)</u></td></tr><tr><td>Output</td><td><u>Console.WriteLine("Goodbye")</u></td></tr></table>	Stage	Example statement	Input	<u>Next = Console.ReadLine()</u>	Process	<u>x = INT(y/3)</u>	Output	<u>Console.WriteLine("Goodbye")</u>	5
Stage	Example statement									
Input	<u>Next = Console.ReadLine()</u>									
Process	<u>x = INT(y/3)</u>									
Output	<u>Console.WriteLine("Goodbye")</u>									
1(c)(ii)	One mark for statement in program code that includes (at least) two 'stages' Example correct answers: <ul style="list-style-type: none">Next = LEN(Console.Input())Console.WriteLine(Name & Address)Console.WriteLine(Console.ReadLine() & " is what you entered")	1								
1(d)	Three from the following (max 3): <ul style="list-style-type: none">Blank linesCapitalisation of KeywordsSensible variable namesUse of (library/built-in) functionsCommentsPrettyPrint / keywords coloured	3								
1(e)	White-box	1								

Question	Answer	Marks												
2(a)(i)	<ul style="list-style-type: none">Count-controlled // FOR loopUsed when the number of iterations is known / fixed	2												
2(a)(ii)	<pre>REPEAT CALL AlarmReset() Status1 ← GetStatus(Sys_A) Status2 ← GetStatus(Sys_B) UNTIL (Status1 = TRUE AND Status2 = TRUE)</pre> <p>One mark for each of:</p> <ul style="list-style-type: none">1 REPEAT ... UNTIL2 Call to AlarmReset()3 Assignment of Status1 and Status24 correct logical test	4												
2(b)	<table><tr><th>Feature</th><th>Answer</th></tr><tr><td>The symbol used to indicate an assignment</td><td>:=</td></tr><tr><td>The line numbers for the start and end of a count-controlled loop</td><td>180/190 and 230</td></tr><tr><td>The step value of the count-controlled loop</td><td>2</td></tr><tr><td>The character that indicates a comment</td><td>%</td></tr><tr><td>The name of a function</td><td>Mult // Read</td></tr></table>	Feature	Answer	The symbol used to indicate an assignment	:=	The line numbers for the start and end of a count-controlled loop	180/190 and 230	The step value of the count-controlled loop	2	The character that indicates a comment	%	The name of a function	Mult // Read	5
Feature	Answer													
The symbol used to indicate an assignment	:=													
The line numbers for the start and end of a count-controlled loop	180/190 and 230													
The step value of the count-controlled loop	2													
The character that indicates a comment	%													
The name of a function	Mult // Read													
2(c)	Compiler / Interpreter	1												

Question	Answer	Marks
3	<pre> sequenceDiagram participant CP as ChangePassword() participant GP as GetPassword() participant UF as UpdateFile() CP->>GP: AccountID GP-->>CP: OldPassword CP->>UF: AccountID UF-->>CP: NewPassword UF-->>CP: </pre> <p>One mark for each of the following:</p> <ul style="list-style-type: none"> all three boxes correctly labelled parameters in to <code>GetPassword()</code> value back from <code>GetPassword()</code> parameters in to <code>UpdateFile()</code> BOOLEAN value back from <code>UpdateFile()</code> 	5

Question	Answer	Marks
4(a)	<p>One mark for each point.</p> <p>Valid string must contain:</p> <ul style="list-style-type: none"> at least one '.' characters one '@' character more than 5 other characters. 	3

Question	Answer	Marks																																																																																																														
4(b)(i)	<div>One mark for each area as outlined:</div> <table><tr><th>Index</th><th>NextChar</th><th>NumDots</th><th>NumA's</th><th>NumOthers</th></tr><tr><td></td><td></td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>'J'</td><td></td><td></td><td>1</td></tr><tr><td>2</td><td>'i'</td><td></td><td></td><td>2</td></tr><tr><td>3</td><td>'m'</td><td></td><td></td><td>3</td></tr><tr><td>4</td><td>'.'</td><td>1</td><td></td><td></td></tr><tr><td>5</td><td>'9'</td><td></td><td></td><td>4</td></tr><tr><td>6</td><td>'9'</td><td></td><td></td><td>5</td></tr><tr><td>7</td><td>'@'</td><td></td><td>1</td><td></td></tr><tr><td>8</td><td>'s'</td><td></td><td></td><td>6</td></tr><tr><td>9</td><td>'k'</td><td></td><td></td><td>7</td></tr><tr><td>10</td><td>'a'</td><td></td><td></td><td>8</td></tr><tr><td>11</td><td>'i'</td><td></td><td></td><td>9</td></tr><tr><td>12</td><td>'l'</td><td></td><td></td><td>10</td></tr><tr><td>13</td><td>'.'</td><td>2</td><td></td><td></td></tr><tr><td>14</td><td>'c'</td><td></td><td></td><td>11</td></tr><tr><td>15</td><td>'o'</td><td></td><td></td><td>12</td></tr><tr><td>16</td><td>'m'</td><td></td><td></td><td>13</td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr></table>	Index	NextChar	NumDots	NumA's	NumOthers			0	0	0	1	'J'			1	2	'i'			2	3	'm'			3	4	'.'	1			5	'9'			4	6	'9'			5	7	'@'		1		8	's'			6	9	'k'			7	10	'a'			8	11	'i'			9	12	'l'			10	13	'.'	2			14	'c'			11	15	'o'			12	16	'm'			13																					5
Index	NextChar	NumDots	NumA's	NumOthers																																																																																																												
		0	0	0																																																																																																												
1	'J'			1																																																																																																												
2	'i'			2																																																																																																												
3	'm'			3																																																																																																												
4	'.'	1																																																																																																														
5	'9'			4																																																																																																												
6	'9'			5																																																																																																												
7	'@'		1																																																																																																													
8	's'			6																																																																																																												
9	'k'			7																																																																																																												
10	'a'			8																																																																																																												
11	'i'			9																																																																																																												
12	'l'			10																																																																																																												
13	'.'	2																																																																																																														
14	'c'			11																																																																																																												
15	'o'			12																																																																																																												
16	'm'			13																																																																																																												
4(b)(ii)	TRUE	1																																																																																																														

Question	Answer	Marks
4(c)	<p>One mark for string and one mark for correct explanation.</p> <p>Same for second answer providing it results in a different path through the algorithm.</p> <p>Correct answers may be:</p> <ul style="list-style-type: none"> • without the correct number of '.' • without the correct number of '@' • without the correct number of 'other characters' 	4

Question	Answer	Marks
5	<pre> FUNCTION Abbreviate(Name : STRING) RETURNS STRING DECLARE NewString : STRING DECLARE NextChar : CHAR DECLARE Index : INTEGER DECLARE Space : BOOLEAN CONSTANT SPACECHAR = ' ' Space ← TRUE NewString ← "" FOR Index ← 1 TO LENGTH(Name) NextChar ← MID(Name, Index, 1) IF Space = TRUE THEN NewString ← NewString & NextChar // first char of next word Space ← FALSE ELSE IF NextChar = SPACECHAR THEN Space ← TRUE ENDIF ENDIF ENDFOR RETURN NewString ENDFUNCTION </pre> <p>1 mark for each of the following (max 8):</p> <ol style="list-style-type: none"> 1 Function header, ending and return parameters 2 Declare and Initialise <code>NewString</code> to either "" or first character of name 3 FOR loop picking out all characters from <code>Name</code>: 4 extract an individual character in a loop 5 check for space character in a loop 6 concatenate the next character to <code>NewString</code> in a loop 7 Return <code>NewString</code> 8 Accommodate a string with trailing space 	8

Question	Answer	Marks
6(a)(i)	<p>One mark per underlined section:</p> <pre> <u>DECLARE Result : ARRAY</u> <u>[0:99, 0:1]</u> <u>OF STRING</u> </pre>	3

Question	Answer	Marks
6(a)(ii)	<p>'Pseudocode' solution included here for development and clarification of mark scheme. Programming language example solutions appear in the Appendix.</p> <pre> FUNCTION FindBooksBy(SearchAuthor : STRING) RETURNS INTEGER DECLARE Title : STRING DECLARE Author : STRING DECLARE Isbn : STRING DECLARE Location : STRING DECLARE Count : INTEGER Count ← 0 OPENFILE "Library.txt" FOR READ WHILE NOT EOF ("Library.txt") READFILE "Library.txt", Title READFILE "Library.txt", ThisAuthor READFILE "Library.txt", ISBN READFILE "Library.txt", Location IF SearchAuthor = ThisAuthor THEN Result[Count, 0] ← Title Result[Count, 1] ← Location Count ← Count + 1 ENDIF ENDWHILE CLOSEFILE("Library.txt") RETURN Count ENDFUNCTION </pre> <p>One mark for each of the following:</p> <ol style="list-style-type: none"> Function heading (and ending) including parameters Declaration of variables used Open file for reading (Allow Library or Library.txt) WHILE loop checking for EOF(): Read all information 'fields', in the correct order, in a loop If the author matches write Title and Location to Result array in a loop And increment array index in a loop // number found Close file and RETURN Count 	8

Question	Answer	Marks
6(b)	<pre> PROCEDURE DisplayResults (Author:STRING, Count:INTEGER) DECLARE Index, GLen : INTEGER DECLARE Gap : STRING Gap ← " " // 25 spaces IF Count = 0 THEN OUTPUT "Search found no books by: " & Author ELSE OUTPUT "Books written by: " & Author OUTPUT "Title" & LEFT(Gap, 20) & "Location" FOR Index ← 1 TO Count GLen ← 25 - LENGTH(Result[Index, 0]) OUTPUT Result[Index, 0] & LEFT(Gap, GLen) & Result[Index, 1] ENDFOR OUTPUT "Number of titles found: " & NUM_TO_STRING(Count) ENDIF ENDPROCEDURE </pre> <p>One mark for each of the following (max 7):</p> <ol style="list-style-type: none"> 1 Procedure heading and ending including parameters 2 Declaration of local INTEGER variable for use as index 3 Test if count = 0 and if so output suitable message including Author for no books found otherwise output the two header strings (exact format not important) 4 A FOR loop for Count times 5 ... output two array elements from Result array in a loop 6 Final output statement 7 A reasonable attempt at calculating the number of spaces required to align 'Location' column: 8 Alignment correct 	7

Program Code Example Solutions

Q6 (a) (ii): Visual Basic

```
FUNCTION FindBooksBy(ByVal SearchAuthor As String) As Integer

    Dim Title As String
    Dim Author As String
    Dim Isbn As String
    Dim Location As String
    Dim Count As Integer

    Count = 0

    FileOpen(1, "Library.txt", OpenMode.Input)

    While Not EOF(1)
        Title = LineInput(1)
        ThisAuthor = LineInput(1)
        Isbn = LineInput(1)
        Location = LineInput(1)

        If SearchAuthor = ThisAuthor Then
            Result(Count, 0) = Title
            Result(Count, 1) = Location
            Count = Count + 1
        End If
    End While

    FileClose(1)

    Return Count
END FUNCTION
```

Q6 (a) (ii): Pascal

```
function FindBooksBy(SearchAuthor : string) : integer;

var
  Title : string;
  Author : string;
  Isbn : string;
  Location : string;
  Count : integer;
  MyFile : text;

begin
  Count := 0;
  assign(MyFile, 'Library.txt');
  reset(MyFile);

  while not EOF(MyFile) do
    begin
      readln(MyFile, Title);
      readln(MyFile, ThisAuthor);
      readln(MyFile, Isbn);
      readln(MyFile, Location);

      if SearchAuthor = ThisAuthor then
        begin
          Result[Count, 0] := Title;
          Result[Count, 1] := Location;
          Count := Count + 1;
        end;
    end;

  close(MyFile);
  FindBooksBy := Count;

end;
```

Q6 (a) (ii): Python

```
def FindBooksBy(SearchAuthor):  
  
    ## Title : STRING  
    ## Author : STRING  
    ## Isbn : STRING  
    ## Location : STRING  
    ## Count : INTEGER  
  
    Count = 0  
    MyFile = open("Library.txt", 'r')  
    Title = MyFile.readline()  
  
    while Title != "":  
        ThisAuthor = MyFile.readline()  
        Isbn= MyFile.readline()  
        Location = MyFile.readline()  
        if SearchAuthor == ThisAuthor.strip():  
            Result[Count][0] = Title  
            Result[Count][1] = Location  
            Count = Count + 1  
        Title = MyFile.readline()  
  
    MyFile.close()  
    return(Count)
```