GradeMax

Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

COMPUTER SCIENCE **9608/23**

Paper 2 Written Paper **May/June 2018**

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 2018 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.

© UCLES 2018

GradeMax

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.

© UCLES 2018

GradeMax

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>1(a)</td><td>
<table>
<tr><th>Description of data item</th><th>Suitable identifier name</th></tr>
<tr><td>The temperature inside the house</td><td>InsideTemperature</td></tr>
<tr><td>The temperature outside the house</td><td>OutsideTemperature</td></tr>
<tr><td>The wind speed</td><td>WindSpeed</td></tr>
<tr><td>Whether it was raining or not</td><td>WasRaining</td></tr>
</table>
The above are examples only.
Names must be meaningful and unambiguous

Items 1 and 2 must have suitable prefix/suffix (i.e. not just ‘temperature’)

Reject single letter names
</td><td>4</td></tr>
<tr><td>1(b)(i)</td><td>
<table>
<tr><th>Expression</th><th>Evaluates to</th></tr>
<tr><td>MID(MyName, 4, 4) & "ol"</td><td>"phenol"</td></tr>
<tr><td>QualityConfirmed AND (Factor >= 6.5)</td><td>TRUE</td></tr>
<tr><td>20 + ASC(Quality)</td><td>88</td></tr>
<tr><td>QualityConfirmed + 3</td><td>ERROR</td></tr>
<tr><td>MOD(Factor * 2, 9)</td><td>4</td></tr>
</table>
</td><td>5</td></tr>
<tr><td>1(b)(ii)</td><td>
<table>
<tr><th>Variable</th><th>Data type</th></tr>
<tr><td>QualityConfirmed</td><td>BOOLEAN</td></tr>
<tr><td>DayNumber</td><td>INTEGER</td></tr>
<tr><td>Factor</td><td>REAL</td></tr>
<tr><td>Quality</td><td>CHAR</td></tr>
<tr><td>MyName</td><td>STRING</td></tr>
</table>
One mark per answer
</td><td>5</td></tr>
</table>

© UCLES 2018

GradeMax

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>2(a)</td><td>Comments:
Explain the functionality of the code // Easier for other people to understand // Easier to maintain / debug / modify

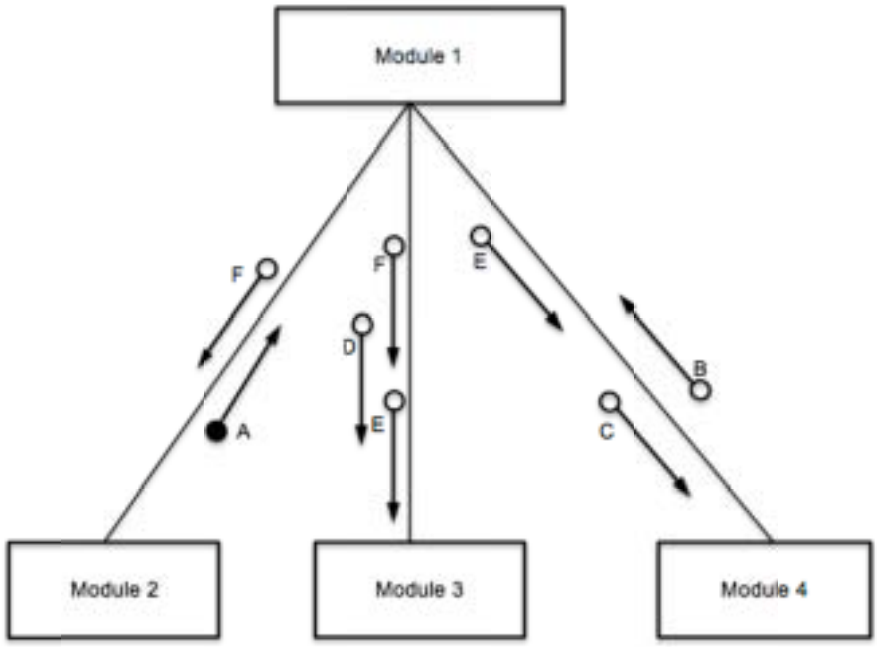
Indentation:
Easier to identify structure / blocks // identify blocks of code</td><td>2</td></tr>
<tr><td>2(b)</td><td><table><tr><th>Feature</th><th>Answer</th></tr><tr><td>A line number containing an example of an integer assignment statement</td><td>8, 9, 10,12, 17, 34, 45</td></tr><tr><td>A line number containing the start of a selection structure</td><td>14, 19, 23</td></tr><tr><td>A line number containing the end of a selection structure</td><td>28, 29, 30</td></tr><tr><td>The upper bound of the <code>Mark</code> array</td><td>100</td></tr><tr><td>The number of dimensions of the <code>Mark</code> array</td><td>1</td></tr><tr><td>The name for the type of loop structure used</td><td>‘post condition’</td></tr><tr><td>A line number containing an unnecessary assignment statement</td><td>10</td></tr><tr><td>The number of times that <code>OUTPUT</code> is called</td><td>100</td></tr><tr><td>The number of local variables</td><td>4</td></tr></table></td><td>9</td></tr>
<tr><td>2(c)(i)</td><td>Either:
• Mistake: function header specifies return of an <code>INTEGER</code> but line 37 returns a <code>STRING</code> // pseudocode returns <code>Grade</code> but should have returned <code>DGradeCount</code>
• Correction: <code>RETURN DGradeCount</code> (as per code pseudocode comment)

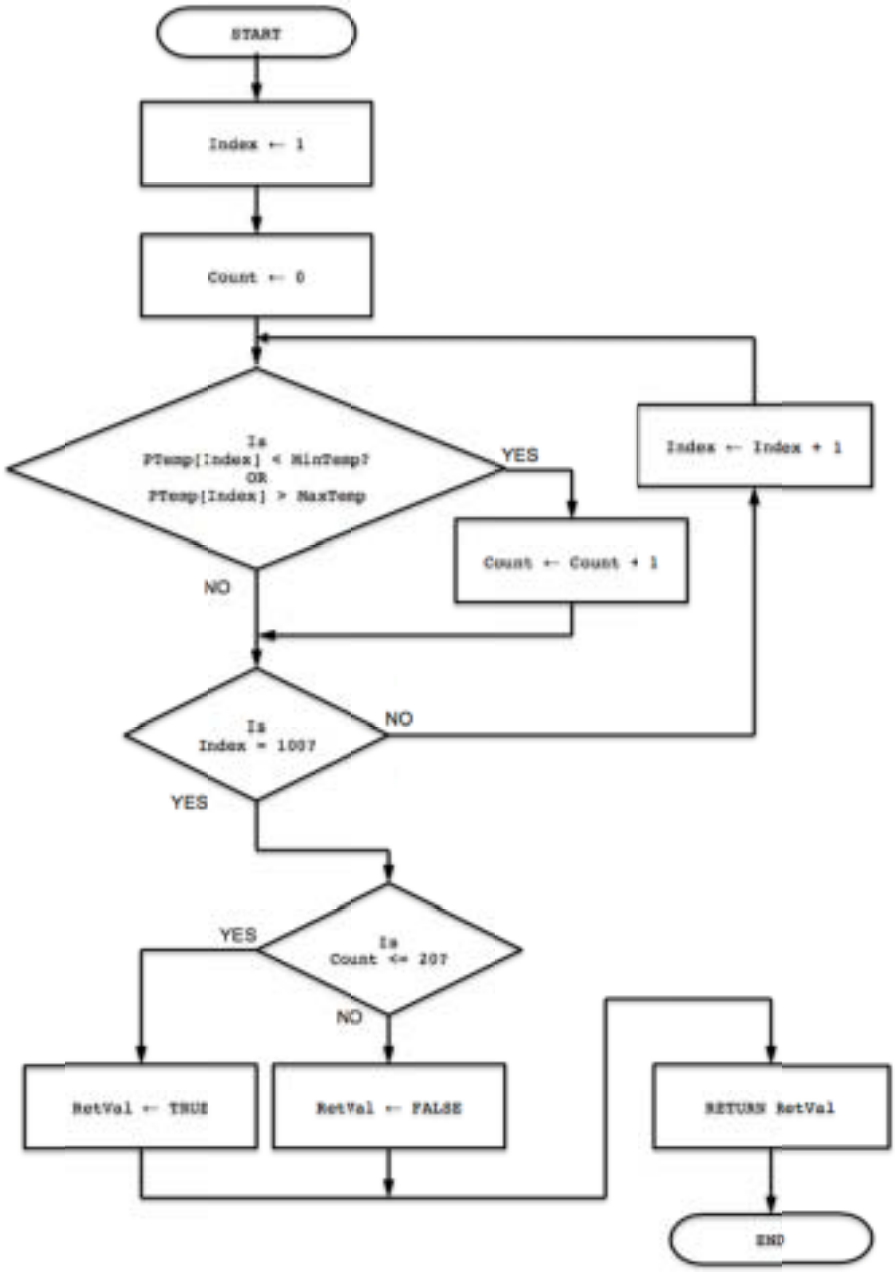
Or:
• Mistake: Statement on line 32 uses ‘&’ operator which concatenates <code>STRING</code>s, but variable n is an <code>INTEGER</code>
• Correction: Convert n to a <code>STRING</code> before concatenating</td><td>2</td></tr>
</table>

© UCLES 2018

Question	Answer	Marks

2(c)(ii)	`CASE OF ThisMark`
	`  > 74:  Grade ← "Distinction"`
	`  DGradeCount ← DGradeCount + 1`
	`  60 TO 74:  Grade ← "Merit"`
	`  40 TO 59:  Grade ← "Pass"`
	`  OTHERWISE  Grade ← "Fail"`
	`ENDCASE`
	
	One mark for each of:
	
	1 `CASE OF ThisMark ... ENDCASE`
	2 Three grade ranges with corresponding assignment of `Grade`
	3 `DGradeCount` increment within `CASE` clause
	4 `OTHERWISE` / fourth grade range with correct assignment of `Grade`
		4

Question	Answer	Marks
3(a)	Parameters Accept arguments	**1**
3(b)	 Mark as follows: • One mark for all four modules • One mark for each set of interface parameters	**4**

Question	Answer	Marks
4	 This is one possible solution – selection structure may differ One mark for: 1 `START` **and** `END // STOP` 2 Initialisation of an `Index` variable **and** initialisation of a `Count` variable 3 Decision box / boxes to check temperature within acceptable range 4 Correct increment of `Count` variable 5 Decision box comparing Index to 100 6 Correct increment of `Index` 7 Decision box comparing `Count` > 20 8 Assigning both `TRUE` **and** `FALSE` 9 Returning the Boolean value For solutions where Boolean variable not used: 8 Return `TRUE` 9 Return `FALSE`	**9**

GradeMax

Question	Answer	Marks
5(a)(i)	1	**1**
5(a)(ii)	Information is saved after the program ends // after the computer is switched off	**1**
5(b)	Two from these examples: • Indentation • Colour-coding of keywords /comments • Expansion / collapsing of complex data structures	**Max 2**

© UCLES 2018

GradeMax

Question	Answer	Marks

5(c)	'Pseudocode' solution included here for development and clarification of mark scheme.
	Programming language solutions appear in the Appendix.
	
	`FUNCTION GetAverageScore(MembershipNumber : STRING)`
	`  RETURNS INTEGER`
	
	`  DECLARE FileData, FileMembershipNumber : STRING`
	`  DECLARE NumberOfScores, TotalScore, AverageScore :`
	`  INTEGER`
	`  OPENFILE "ScoreDetails.txt" FOR READ`
	`  NumberOfScores ← 0`
	`  TotalScore ← 0`
	
	`  WHILE NOT EOF("ScoreDetails.txt")`
	`  READFILE("ScoreDetails.txt", FileData)`
	`  FileMembershipNumber ← LEFT(FileData, 4)`
	`  IF FileMembershipNumber = MembershipNumber`
	`  THEN`
	`  NumberOfScores ← NumberOfScores + 1`
	`  TotalScore ← TotalScore +`
	`  INT(RIGHT(FileData, 2))`
	`  ENDIF`
	`  ENDWHILE`
	
	`  AverageScore ← INT(TotalScore / NumberOfScores)`
	
	`  CLOSEFILE("ScoreDetails.txt")`
	
	`  RETURN(AverageScore)`
	
	`ENDFUNCTION`
	
	1 mark for each of the following:
	
	1 Function heading **and** ending including Input **and** return parameter
	2 Declare variables to store `NumberOfScores` **and** `TotalScore` as INTEGERs (commented in Python) (variable names may be different)
	3 Initialisation of `NumberOfScores` **and** `TotalScore` to 0
	4 Open file in `READ` mode
	5 Loop until `EOF()`
	6 Read a line from the file **in a loop**
	7 Use of substring function to extract at least one data item
	8 Compare the membership number
	9 Convert score to an integer
	10 Increment `NumberOfScores` **and** sum `TotalScore`
	11 Calculate the average outside the loop
	12 Close the file
	13 Return the parameter
		Max 10

© UCLES 2018

GradeMax

Question	Answer	Marks
6(a)	Subscript / index	**1**

6(b)	`FUNCTION Clip(MaxVal : INTEGER) RETURNS BOOLEAN`
	`  DECLARE i : INTEGER`
	`  DECLARE j : INTEGER`
	`  DECLARE ClipFlag : BOOLEAN`
	
	`  ClipFlag ← FALSE`
	
	`  FOR i ← 1 TO 8`
	`  FOR j ← 1 TO 8`
	`  IF Picture[i, j] > MaxVal`
	`  THEN`
	`  Picture[i, j] ← MaxVal`
	`  ClipFlag ← TRUE`
	`  ENDIF`
	`  ENDFOR`
	`  ENDFOR`
	
	`  RETURN ClipFlag`
	
	`ENDFUNCTION`
	
	1 mark for each of the following:
	
	1 Correct Function heading (must have `MaxVal` and return a `BOOLEAN`) **and** ending
	2 Declare **and** initialise local variable for return `BOOLEAN` to `FALSE` / other mechanism to record pixel being clipped
	3 Declare local variables for loop counters
	4 Nested loops with correct number of iterations
	5 Accessing correct element from `Picture` array
	6 Comparing element with `MaxVal`
	7 Changing value of element if necessary
	8 Setting flag to `TRUE` / other mechanism if element is changed
	9 Returning `BOOLEAN` **after loop** (following conversion if other mechanism used)
		9

© UCLES 2018

GradeMax

Question	Answer	Marks

7	'Pseudocode' solution included here for development and clarification of mark scheme.
	Programming language solutions appear in the Appendix.
	
	`FUNCTION IsFactor(Num1: INTEGER, Num2: INTEGER)`
	`  RETURNS BOOLEAN`
	`  IF Num2 <> 0`
	`  THEN`
	`  IF MOD(Num1, Num2) = 0`
	`  THEN`
	`  RETURN TRUE`
	`  ENDIF`
	`  ENDIF`
	
	`  RETURN FALSE`
	
	`ENDFUNCTION`
	
	1 mark for each of the following:
	
	1 Correct Function heading (including parameters) and ending
	2 Check that `Num2` is not zero
	3 Mechanism to ensure no call to `MOD` (or equivalent) if `Num2` is zero
	4 Use of `MOD` function or alternative
	5 Check value of remainder
	6 Return Boolean value
		6

*** End of Mark Scheme – program code example solutions follow ***

© UCLES 2018

GradeMax

Appendix

Program Code Example Solutions

Q5(c): Visual Basic

```
Function GetAverageScore(ByVal MembershipNumber As String) As Integer

    Dim FileData As String
    Dim FileMembershipNumber As String
    Dim NumberOfScores As Integer
    Dim TotalScore As Integer
    Dim AverageScore As Integer
    Dim ObjReader As IO.StreamReader

    ObjReader = New IO.StreamReader("ScoreDetails.txt")
    NumberOfScores = 0
    TotalScore = 0

    Do While ObjReader.Peek <> -1
        FileData = ObjReader.ReadLine()
        FileMembershipNumber = LEFT(FileData, 4)
        If FileMembershipNumber = MembershipNumber Then
            NumberOfScores = NumberOfScores + 1
            TotalScore = TotalScore + INT(RIGHT(FileData, 2))
        End If
    Loop

    AverageScore = INT(TotalScore / NumberOfScores)
    ObjReader.Close()
    Return (AverageScore)

End Function
```

Q5(c): Pascal

```
function GetAverageScore(MembershipNumber : string):integer;

 var
    FileData, FileMembershipNumber: string;
    NumberOfScores, TotalScore, AverageScore : integer;
    ScoreFile : textFile;
 begin
    NumberOfScores := 0;
    TotalScore := 0;
    assignFile(ScoreFile, 'ScoreDetails.txt');
    reset(ScoreFile);
    while not eof(ScoreFile) do
    begin
        readln(ScoreFile, FileData);
        FileMembershipNumber := copy(FileData, 1, 4);
        if FileMembershipNumber = MembershipNumber then
        begin
            NumberOfScores := NumberOfScores + 1
            TotalScore := TotalScore + StrToInt(RightStr(FileData, 2));
         end;
    end;
```

© UCLES 2018

```
    AverageScore := StrToInt(TotalScore / NumberOfScores);
    GetAverageScore := AverageScore;
    CloseFile (ScoreFile);
 end;
```

Q5(c): Python

```
# FileData AS STRING
# FileMembershipNumber AS STRING
# NumberOfScores AS INTEGER
# TotalScore AS INTEGER
# AverageScore AS INTEGER


def GetAverageScore(MembershipNumber):
    FileHandle = open("ScoreDetails.txt", "r")
    NumberOfScores = 0
    TotalScore = 0
    FileData  = FileHandle.readline()
    while len(FileData) > 0:
        FileMembershipNumber = FileData[0:4]
        if FileMembershipNumber == MembershipNumber:
            NumberOfScores = NumberOfScores + 1
            TotalScore = TotalScore + int(FileData[-2])
            FileData  = FileHandle.readline()
    AverageScore = int(TotalScore / NumberOfScores)
    Return (AverageScore)
    FileHandle.close()
```

© UCLES 2018

GradeMax

Q7: Visual Basic

```
Function IsFactor(Num1 As Integer, Num2 As Integer) As Boolean

    If Num2 <> 0 Then
        If Num1 Mod Num2 = 0 Then
            Return True
        End If
    End if
    Return False

End Function
```

Q7: Pascal

```
function IsFactor(Num1,Num2 : integer) : boolean;


 begin
    if Num2 <> 0 then
    begin
        if Num1 MOD Num2 = 0 then
            Return True;
    end;
    Return False;

 end;
```

Q7: Python

```
def IsFactor (Num1, Num2):
    if Num2 != 0:
        if Num1 % Num2 == 0:
            Return True
    Return False
```

© UCLES 2018