Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

COMPUTER SCIENCE **9608/21**

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

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

<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 of the patient</td><td>PatientTemperature</td></tr>
<tr><td>The temperature of the room</td><td>RoomTemperature</td></tr>
<tr><td>The patient identification number</td><td>PatientID</td></tr>
<tr><td>The name of the nurse taking the measurement</td><td>NurseName</td></tr>
</table>
The above are examples only.
Names must be meaningful and unambiguous

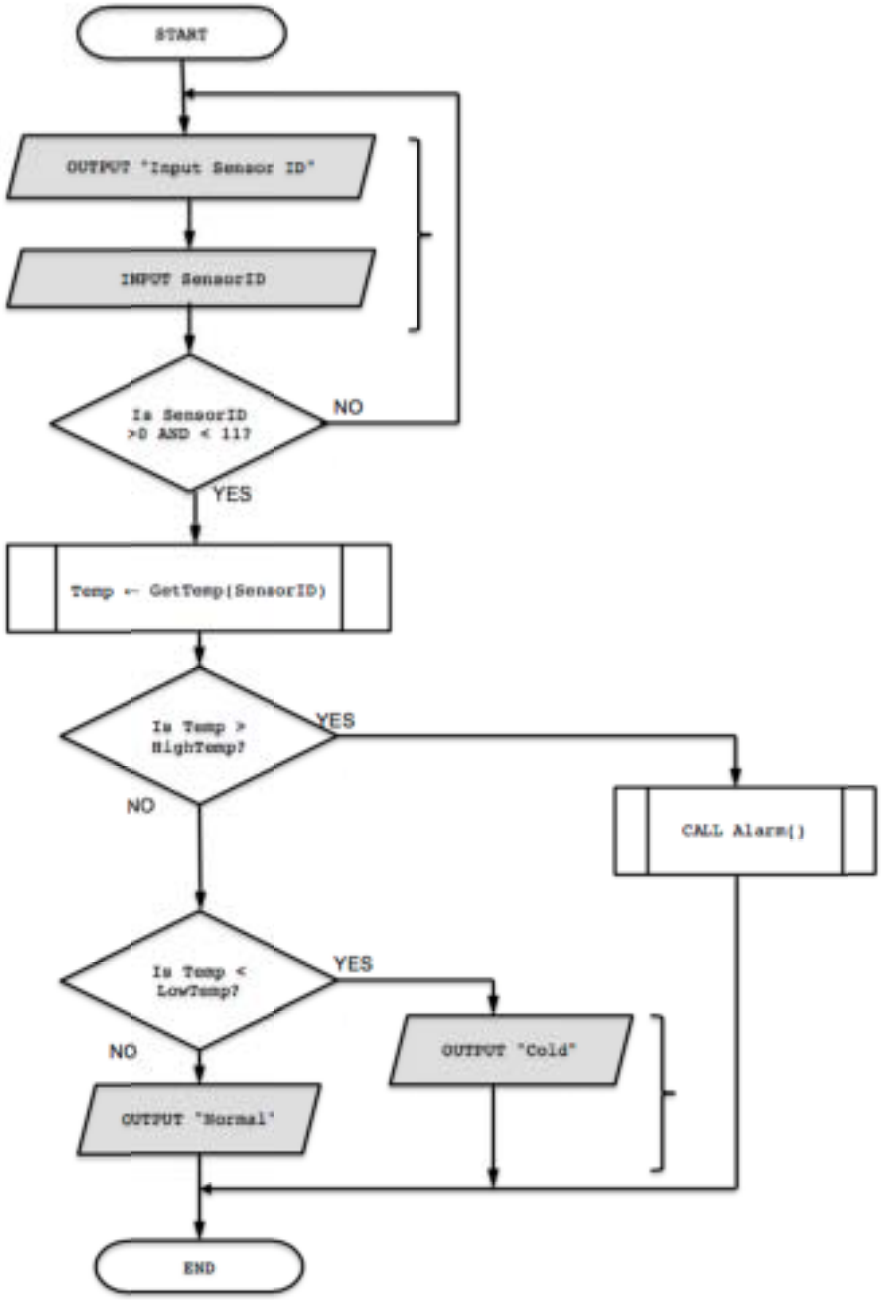
Items 1 and 2 must have suitable prefix / suffix (i.e. not just 'temp')
</td><td>4</td></tr>
<tr><td>1(b)(i)</td><td>
<table>
<tr><th>Expression</th><th>Evaluates to</th></tr>
<tr><td><code>"Mon" & MID(MyGreeting, 10, 2)</code></td><td>"Month"</td></tr>
<tr><td><code>AgeInYears + ASC(MyInitial)</code></td><td>94</td></tr>
<tr><td><code>INT(MyInitial)</code></td><td>ERROR</td></tr>
<tr><td><code>MOD(Weight * 2, 10)</code></td><td>1</td></tr>
<tr><td><code>Maried AND (NOT Children)</code></td><td>FALSE</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><code>MyGreeting</code></td><td>STRING</td></tr>
<tr><td><code>MyInitial</code></td><td>CHAR</td></tr>
<tr><td><code>AgeInYears</code></td><td>INTEGER</td></tr>
<tr><td><code>Weight</code></td><td>REAL</td></tr>
<tr><td><code>Married</code></td><td>BOOLEAN</td></tr>
</table>
One mark per answer

Alternative appropriate data types acceptable
</td><td>5</td></tr>
</table>

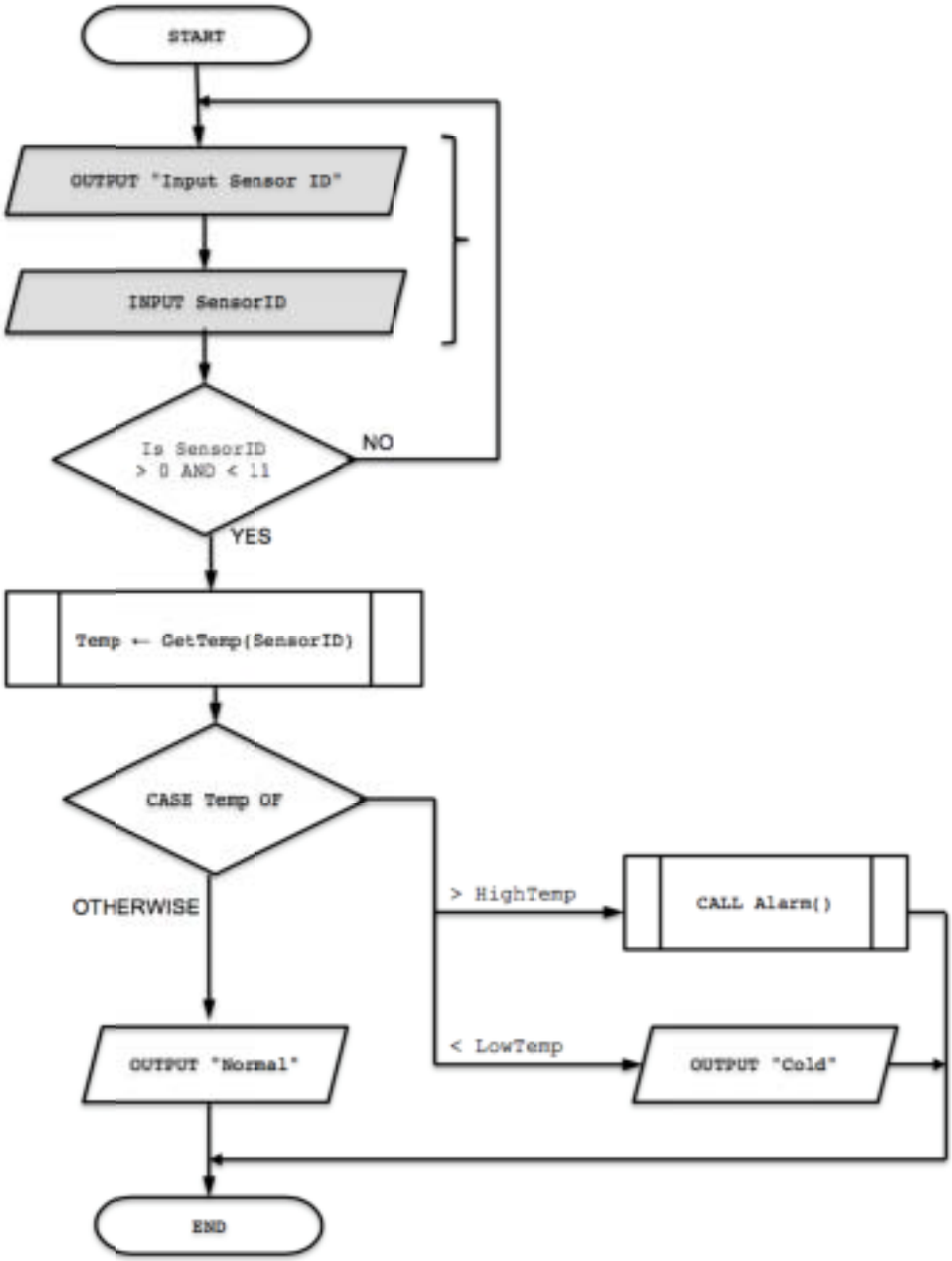
Question	Answer	Marks
2(a)(i)	• Indentation • Blank lines / white space • Capitalisation of keywords • Meaningful identifier names	4
2(a)(ii)	• Comments	1

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>2(b)</td><td>
<table>
<tr><th>Feature</th><th>Answer</th></tr>
<tr><td>A line containing an example of an assignment statement</td><td>08,12,13,19</td></tr>
<tr><td>A line containing the start of a repetition block</td><td>10</td></tr>
<tr><td>A line containing the end of a repetition block</td><td>23</td></tr>
<tr><td>The line containing the start of a selection statement</td><td>15</td></tr>
<tr><td>The number of parameters of the MID function</td><td>3</td></tr>
<tr><td>The boolean operator used</td><td>AND</td></tr>
<tr><td>The number of local variables</td><td>4</td></tr>
<tr><td>The number of function calls from within <code>StringClean()</code> resulting from the call:
<code>NewString ← StringClean("Me")</code></td><td>5</td></tr>
<tr><td>The number of a line containing an unneccessary statement</td><td>06</td></tr>
</table>
</td><td>9</td></tr>
</table>

© UCLES 2018

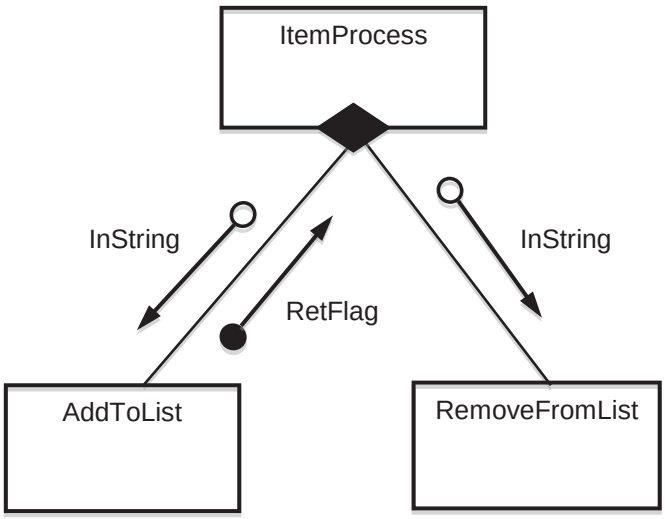
Question	Answer	Marks
3	 One mark for: 1 START and END / STOP 2 prompt and input of `SensorID` (allow alt name) 3 decision box checking that `SensorID` is between 1 & 10 4 calling `GetTemp` with `SensorID` as parameter 5 decision box comparing `Temp` > `HighTemp` 6 calling `Alarm()` 7 decision box comparing `Temp` < `LowTemp` 8 both output messages	**8**

Question	Answer	Marks
3	ALTERNATIVE SOLUTION USING 'CASE'	**8**



Mark as follows:

1 One mark for START and END / STOP
2 One mark for prompt and input of `SensorID` (allow alt name)
3 One mark for decision box checking that `SensorID` is between 1 & 10
4 One mark for calling `GetTemp` with `SensorID` as parameter
5 One mark for decision box `CASE Temp`
6 One mark for calling `Alarm()`
7 One mark for two correct `CASE` conditions
8 One mark for both output messages

Question	Answer	Marks
4(a)	Features: • The hierarchy of modules • The parameters that are passed between modules // the interface between the modules • The sequence of module execution One mark per item	**3**
4(b)	 Mark as follows: One mark for top box One mark for **both** lower boxes One mark for diamond 'decision' symbol One mark for each parameter (3 parameters)	**6**

Question	Answer	Marks
5(a)	Details are saved after the program ends // after the computer is switched off	**1**
5(b)	Two from the following examples: • Context-sensitive help • Syntax checking (on entry) • Automatic indentation • Type checking (Parameter checking) • PrettyPrinting • Highlight structure blocks (e.g. selection, iteration) • Highlight any undeclared variables • Highlight any unassigned variables	**Max 2**

© UCLES 2018

Question	Answer	Marks

5(c)	'Pseudocode' solution included here for development and clarification of mark scheme.
	Programming language solutions appear in the Appendix.
	
	`PROCEDURE AddNewScores()`
	
	`  DECLARE FileData : STRING`
	`  DECLARE ScoreDate : STRING`
	`  DECLARE MembershipNumber : STRING`
	`  DECLARE Score : STRING`
	
	`  OUTPUT "Input the date for the scores"`
	`  INPUT ScoreDate`
	
	`  OPENFILE "ScoreDetails.txt" FOR APPEND`
	
	`  OUTPUT "Input the Membership number"`
	`  INPUT MembershipNumber`
	
	`  WHILE NOT MembershipNumber = ""`
	`  OUPUT "Input the score"`
	`  INPUT Score`
	`  WHILE (INT(SCORE) < 50) OR (INT(SCORE) > 99)`
	`  OUTPUT "Input a valid score from 50 to 99"`
	`  INPUT Score`
	`  ENDWHILE`
	`  FileData = MembershipNumber & ScoreDate & Score`
	`  WRITEFILE "ScoreDetails.txt", FileData`
	`  OUTPUT "Input the Membership number"`
	`  INPUT MembershipNumber`
	
	`  ENDWHILE`
	
	`  CLOSEFILE("ScoreDetails.txt")`
	
	`ENDPROCEDURE`
	
	1 mark for each of the following:
	
	1 Declare `MembershipNumber` as `STRING` and `Score` as `INTEGER` / `STRING` (commented in Python)
	2 Prompt and Input of `ScoreDate`
	3 Open `ScoreDetails.txt` in APPEND mode
	4 Prompt and Input of `MembershipNumber` **in a loop**
	5 (Outer) loop terminated when `MembershipNumber` = ""
	6 Input `Score` **and** loop until valid
	7 Form the text string from the three variables
	8 Write the text string to the file
	9 Close the file
		9

Question	Answer	Marks
6(a)(i)	• The array is 1D • 1 is the lower bound • 5 is the upper bound • size of array / number of elements = 5	**Max2**
6(a)(ii)	• subscript / index	**1**

6(b)	`FUNCTION Lighten() RETURNS BOOLEAN`
	`  DECLARE OldPixelValue : INTEGER`
	`  DECLARE NewPixelValue : INTEGER`
	`  DECLARE PixelTemp : REAL`
	`  DECLARE BurnFlag : BOOLEAN`
	`  DECLARE i : INTEGER`
	`  DECLARE j : INTEGER`
	
	`  BurnFlag ← FALSE`
	
	`  FOR i ← 1 TO 8`
	`  FOR j ← 1 TO 8`
	`  OldPixelValue ← Picture[i, j]`
	`  PixelTemp ← OldPixelValue * 1.1`
	`  NewPixelValue ← INT(PixelTemp)`
	`  IF NewPixelValue >= 255`
	`  THEN`
	`  NewPixelValue ← 255`
	`  BurnFlag ← TRUE`
	`  ENDIF`
	`  Picture[i, j] ← NewPixelValue`
	`  ENDFOR`
	`  ENDFOR`
	
	`  RETURN BurnFlag`
	
	`ENDFUNCTION`
	
	1 mark for each of the following:
	
	1 Function heading as above and ending
	2 Declare **and** initialise local variable for return `BOOLEAN` / other mechanism to record 'burnt out'
	3 Declare local variables for loop counters
	4 Correct nested loops
	5 Accessing element from array
	6 Calculating new value **and** convert to an `INTEGER`
	7 Comparing new value with 255 and if greater:
	8 …limit to 255 and assign to original element
	9 …Set flag / other mechanism if limit applied (Only change once)
	10 Return a `BOOLEAN` (following conversion if other mechanism used) MUST WORK
		MAX8

Question	Answer	Marks

7	'Pseudocode' solution included here for development and clarification of mark scheme.
	Programming language solutions appear in the Appendix.
	
	`FUNCTION ProcessMarks(Mark: ARRAY[1:20] OF INTEGER)`
	`RETURNS INTEGER`
	`  DECLARE Highest : INTEGER`
	`  DECLARE Average as REAL`
	`  DECLARE Total as INTEGER`
	`  DECLARE Position as INTEGER`
	
	`  Total ← 0`
	`  Highest ← Mark[1] //The highest mark is the first one`
	`  Position ← 1`
	
	`  FOR i ← 1 to 20`
	`    Total ← Total + Mark[i]`
	`    IF Mark[i] > Highest`
	`          THEN`
	`            Highest ← Mark[i]`
	`            Position ← i`
	`    ENDIF`
	`  ENDFOR`
	`  Average ← Total/20`
	`  Output ("The average mark is " & Average & " and the highest mark is " & Highest)`
	`  RETURN Position`
	
	`ENDFUNCTION`
	
	1 mark for each of the following:
	
	1 Correct Function heading (including `Mark` as parameter) and ending
	2 Declare local variable for `Highest` and initialise
	3 Loop structure (1 to 20 or 0 to 19)
	4 Comparison with current `Highest`
	5 …Assign new `Highest`
	6 Calculate `Average`
	7 Output message including both variables and explanatory text
	8 Return value of index
		Max7

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

© UCLES 2018

Appendix

Program Code Example Solutions

Q5(c)

Visual Basic

```
Sub AddNewScores()

 Dim FileData As String
 Dim MembershipNumber As String
 Dim Score As Integer
 Dim ScoreDate As String
 Dim FileHandle As IO.StreamWriter

    FileHandle = New IO.StreamWriter("ScoreDetails.txt")

    Console.WriteLine("Input the date for the scores")
    ScoreDate = Console.ReadLine()

    Console.WriteLine("Input the Membership number")
    MembershipNumber = Console.ReadLine()

    Do While MembershipNumber <> ""
       Console.WriteLine("Input the score")
       Score = Console.ReadLine()
       Do While Score < 50 Or Score > 99
          Console.WriteLine("Input a valid score from 50 to 99")
          Score = Console.ReadLine()
       Loop
       FileData = MembershipNumber & ScoreDate & Str(Score)
       FileHandle.WriteLine(FileData)
       Console.WriteLine("Input the Membership number")
       MembershipNumber = Console.ReadLine()

    Loop
    FileHandle.Close()

End Sub
```

© UCLES 2018

Python

```
def AddNewScores():
  #ScoreDate as string
  #MembershipNumber as string
  #Score as integer
  #FileHandle as text file

  FileHandle = open("ScoreDetails.txt", "a")
  ScoreDate = str(input("Input the date for the scores"))
  MembershipNumber = str(input("Input the Membership number"))
  while MembershipNumber != "":
     Score = int(input("Input the score"))
     while Score < 50 or Score > 99:
        Score = int(input("Input a valid score from 50 to 99"))
        FileData = MembershipNumber + ScoreDate + str(Score)
        FileHandle.write(FileData)
        MembershipNumber = str(input("Input the Membership number"))
  FileHandle.close
```

Pascal

```
procedure AddNewScores;
  var FileData, ScoreDate, MembershipNumber: String;
  Score: Integer
  MyFile: text;

  begin
     assign(MyFile,'ScoreDetails.txt');
     append(MyFile);
     writeln('Input the date for the scores');
     readln(ScoreDate);
     writeln('Input the Membership number');
     readln(MembershipNumber);

     while MembershipNumber <> '' do
        begin
           writeln('Input the score');
           readln(Score);
           while (Score < 50) or (Score > 99) do
              begin
                 writeln('Input the score');
                 readln(Score);
              end;
           FileData := MembershipNumber + ScoreDate + IntToStr(Score);
           write(MyFile, FileData);
           writeln('Input the Membership number');
           readln(MembershipNumber);
        end;
     close (MyFile);
  end;
```

Q7

Visual Basic

```
Function ProcessMarks(ByVal Mark() As Integer) As Integer

 Dim Highest As Integer
 Dim Average As Single
 Dim Total As Integer
 Dim Position As Integer
 Dim i As Integer

 Total = 0
 Position = 1
 Highest = Mark(1)
 For i = 1 To 20
    Total = Total + Mark(i)
       If Mark(i) > Highest Then
          Highest = Mark(i)
          Position = i
       End If
 Next
 Average = Total / 20
 Console.WriteLine ("The average mark is " & Average & _
                    " and the highest mark is " & Highest)

 Return Position
End Function
```

Python

```
def ProcessMarks (mark):
  #highest, i, position, total as integer
  #average as real

  highest = mark[0]
  total = 0
  position = 0
  for i in range(0,20):
     total = total + mark[i]
     if mark[i] > highest:
        highest = mark[i]
        position = i
  average = total/20
  print('The average mark is ' + str(average) + \
        ' and the highest mark is ' + str(highest))
  return position
```

Pascal

```
function ProcessMarks (mark:array of integer):integer;

  var highest, total, position, i: integer;
  average: real;

  begin
     highest := mark[1];
     total := 0;
     position := mark[1];
     for i := 1 to 20 do
        begin
           total := total + mark[1];
           if  mark[i] > highest then
           begin
              highest := mark[i];
              position := i;
           end;
        end;
     average := total / 20;
     writeln ('The average mark is ', average, ' and the highest mark is ',
highest);
     ProcessMarks := position;
  end;
```