Cambridge International Examinations
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/22**

Paper 2 **March 2017**

MARK SCHEME

Maximum Mark: 50

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 March 2017 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

® IGCSE is a registered trademark.

This syllabus is approved for use in England, Wales and Northern Ireland as a Cambridge International Level 1/Level 2 Certificate.

© UCLES 2017

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>1(a)(i)</td><td>Many correct answers, the identifier must be meaningful and appropriate size if present. These are examples only
<code>ReactionTime [1:650],ReactionTime [0:649],</code>
<code>ReactionTime [650],ReactionTime[649], ReactionTime[ ]</code></td><td>1</td></tr>
<tr><td>1(a)(ii)</td><td>Many correct answers, the identifier must be the same as part 1(a)(i) including appropriate size if present. These are examples only
<code>ReactionTime [1:50], ReactionTime [0:49], ReactionTime [50],</code>
<code>ReactionTime[49], ReactionTime[ ]</code></td><td>1</td></tr>
<tr><td>1(a)(iii)</td><td>Any two from:
– can store multiple reaction times under a single identifier
– reduces the number variables
– arrays have an index which identifies each stored element
– can use iteration to loop through an array
– allows for more efficient programming
– programs are easier to debug</td><td>2</td></tr>
<tr><td>1(b)</td><td>Any three from:

– an effective loop to accept 650 records
– prompt for all three inputs
– within the loop reads all three INPUT values
– storing input values in appropriate arrays

Sample Answer.
<pre>FOR Counter ← 1 TO 650
   OUTPUT ('Input House, Age and Reaction Time')
   INPUT HouseArray [Counter], AgeArray [Counter],
   ReactionTimeArray[Counter]
NEXT</pre></td><td>3</td></tr>
<tr><td>1(c)</td><td>1 mark for correct type of test data (max 3)
…… 1 mark for appropriate example (max 3)

<table><tr><td>Normal / Valid</td><td>12 / 13 / 14 / 15 / 16</td></tr><tr><td>Erroneous / Abnormal / Invalid</td><td>13.5 / Twelve / 9</td></tr><tr><td>Boundary (accepted)
Boundary (rejected)</td><td>12 or 16
11 or 17</td></tr><tr><td>Extreme</td><td>12 or 16</td></tr></table></td><td>6</td></tr>
<tr><td>1(d)</td><td>Any five from following explanations:
– user input for House and Age
– loop through the arrays
– use selection statements to identity the elements that meet <u>both</u> criteria
– maintain counter of elements (that met criteria of House and Age input)
– maintain a sum of reaction times (that match criteria of House and Age input)
– calculate the average from element counter and sum of reaction times
– create appropriate output message
– output message and average outside of loop</td><td>5</td></tr>
</table>

Question	Answer	Marks
1(e)	Any **two** from following explanations: – variable used to hold fastest time will have to initialised to a high value / variable used to hold **fastest** time will be given first record value – store array value in variable if reaction time less than current value in variable – store array value of age with the same index in a variable – Output age and fastest reaction time	**2**

© UCLES 2017

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td colspan="3">Section B</td></tr>
<tr><td>2</td><td>1 mark for each error identified with effective corrective action
<pre>
01 Num18 = 0
02 INPUT Age
03 WHILE Age >= 0 DO
04   IF Age >= 18 THEN
05   Num18 = Num18 + Age
06   END IF
07 END WHILE
08 PRINT Num18 - Age
</pre>
Error – Line 04 or <code>IF Age >= 18</code> and Correction – <code>IF Age >18</code>

Error – Line 05 or <code>Num18 =Num18 + Age</code> and Correction – <code>Num18 = Num18 + 1</code>

Error – Line 08 or <code>PRINT Num18 - Age</code> and Correction – <code>PRINT Num18</code>

Error – <code>INPUT Age</code> missing inside loop and Correction – Include <code>INPUT Age</code> after test and before exiting loop</td><td>4</td></tr>
<tr><td>3</td><td>1 mark for each correctly completed element of the grid

<table>
<tr><th>Variable</th><th>Data Type</th><th>Appropriate Validation Check</th></tr>
<tr><td>EmployeeID</td><td>String</td><td>Length Check / Presence Check / Format Check / Type check</td></tr>
<tr><td>Manager</td><td>Boolean</td><td>Type Check / Presence Check</td></tr>
<tr><td>AnnualHoliday</td><td>Integer</td><td>Type Check / Range Check / Presence Check</td></tr>
<tr><td>PayGrade</td><td>Char</td><td>Presence Check / Length Check / Type Check</td></tr>
</table></td><td>8</td></tr>
<tr><td>4</td><td>1 mark for each correct column

<table>
<tr><th>A</th><th>B</th><th>C</th><th>Output</th></tr>
<tr><td>4</td><td>4</td><td>4</td><td></td></tr>
<tr><td></td><td>8</td><td>3</td><td></td></tr>
<tr><td></td><td>12</td><td>2</td><td></td></tr>
<tr><td></td><td>16</td><td>1</td><td>16</td></tr>
<tr><td>3</td><td>3</td><td>3</td><td></td></tr>
<tr><td></td><td>6</td><td>2</td><td></td></tr>
<tr><td></td><td>9</td><td>1</td><td>9</td></tr>
<tr><td>-1</td><td></td><td></td><td>Exit</td></tr>
</table></td><td>4</td></tr>
</table>

© UCLES 2017

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>– initialising counter outside the loop
– updating counter inside loop
– suitable exit value at start of loop
– correct use of <code>WHILE … DO … ENDWHILE</code>

Example:

<pre>INPUT Num
Counter ← 1
WHILE Counter <= 12 DO
    Num ← Num * Counter
    A [Counter] ← Num
    Counter ← Counter + 1
ENDWHILE</pre></td><td>4</td></tr>
<tr><td>5(b)</td><td>– <code>WHILE</code> has criteria check at start / pre-test
– may never run
– <code>REPEAT UNTIL</code> has criteria check at end / post-test
– will always run at least once</td><td>4</td></tr>
<tr><td>6(a)</td><td>Alan Swales Chantel Law

• Correct data
• Correct order</td><td>2</td></tr>
<tr><td>6(b)</td><td><table>
<tr><td>Field:</td><td>Device ID</td><td>Device Type</td><td>Purchase Date</td><td>Purchase Price ($)</td></tr>
<tr><td>Table:</td><td>DEVICE</td><td>DEVICE</td><td>DEVICE</td><td>DEVICE</td></tr>
<tr><td>Sort:</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Show:</td><td>☑</td><td>☑</td><td>☐</td><td>☐</td></tr>
<tr><td>Criteria:</td><td></td><td>Like ‘Desktop’</td><td><#31/12/2016#</td><td></td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td><1000</td></tr>
</table>1 mark for each correct column</td><td>4</td></tr>
</table>