Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/22**

Paper 2 **May/June 2018**

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 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.

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

This document consists of **7** printed pages.

© 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.

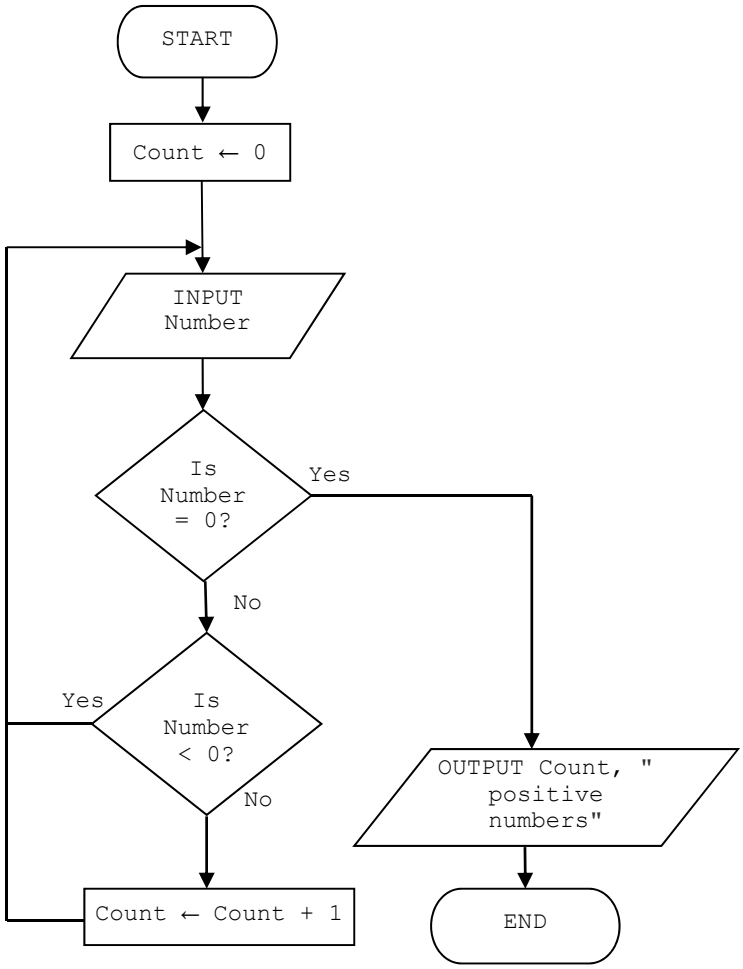
© UCLES 2018

Question	Answer	Marks
	Section A	
1(a)(i)	Variable name, data type and use **one** mark, max **two** Several correct answers, the names chosen must be meaningful. Variables must relate to task 2 Example Name `totalMilk` Data type integer/real Use to store the total volume of the milk for the week (to the nearest whole litre) (1) Name `weeklyAverage` Data type integer/real Use to store the average yield per week (1)	**2**
1(a)(ii)	**One** mark per bullet point. • Data structure(s) given (1) • Data type (1) • Sample data (1) • More than one data structure <u>described</u> (1) Example A real array for each milking and an array of strings for the identity codes. There would be 14 arrays for the milking e.g. `mondayMorning, mondayEvening` Sample data for a cow could be 123, 23.5, 22.7	**4**
1(b)	Entering/selecting the identity code (1) method to ensure it is not a duplicate (1) Example Enter new identity code number Check if already in the list of code numbers	**2**

Question	Answer	Marks

1(c)	Any **five** from:
	1 Initialisation for total weekly volume
	2 loop control
	3 calculation of running total for yield
	4 calculation of average yield
	5 output total and average yield per week with message outside loop
	6 value(s) rounded
	
	Sample answer
	`total ← 0`
	`FOR counter ← 1 TO numCows`
	`  total ← total + mondayMorning(counter)`
	`  total ← total + mondayEvening(counter)`
	`  total ← total + tuesdayMorning(counter)`
	`  total ← total + tuesdayEvening (counter)`
	`  total ← total + wednesdayMorning(counter)`
	`  total ← total + wednesdayEvening (counter)`
	`  total ← total + thursdayMorning(counter)`
	`  total ← total + thursdayEvening (counter)`
	`  total ← total + fridayMorning(counter)`
	`  total ← total + fridayEvening (counter)`
	`  total ← total + saturdayMorning(counter)`
	`  total ← total + saturdayEvening (counter)`
	`  total ← total + sundayMorning(counter)`
	`  total ← total + sundayEvening(counter)`
	`NEXT counter`
	`Average ← ROUND(total/numCows)`
	`OUTPUT "Total volume of milk for week ", ROUND(total)`
	`OUTPUT "Average weekly yield ", average`
		5

1(d)(i)	Explanation Any **five** from: 1 Check each cow 2 Initialise day counter to zero 3 Check every day of the week 4 If daily yield is less than 12 … 5 … add one to day counter 6 If day counter >= 4 … 7 … identify/output identity code number(s)	**5**
1(d)(ii)	Explanation • Add new storage space to store code numbers for example new array/table/list • Add extra code to store these values if the condition was met	**2**

Question	Answer	Marks
	Section B	
2(a)	**One** mark per correct pair of actions, process, Input/Output, Tests (apart from START and END) max 3 **One** mark complete Flowlines, **one** mark working flowlines, **one** mark correct use flowchart symbols  	**6**
2(b)	Any **two** from: • Use another counter/variable • Update this counter/variable when the number is less than zero/count all numbers **and** subtract the positive numbers • Output this counter/variable at the end // Output both counters at the end	**2**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>3(a)</td><td>
<table>
<tr><th>Number1</th><th>Number2</th><th>Sign</th><th>Answer</th><th>OUTPUT</th></tr>
<tr><td>5</td><td>7</td><td>+</td><td>12</td><td>12</td></tr>
<tr><td>6</td><td>2</td><td>-</td><td>4</td><td>4</td></tr>
<tr><td>4</td><td>3</td><td>*</td><td>12</td><td>12</td></tr>
<tr><td>7</td><td>8</td><td>?</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td>/</td><td>(0)</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>
← 1 mark → ← 1 mark → ← 1 mark →
</td><td>3</td></tr>
<tr><td>3(b)</td><td>
<code>CASE Sign OF … ENDCASE</code> (1)
List +, -, *, / with correct assignments (1)
<code>OTHERWISE Answer ← 0</code> (1)
Example
<pre>CASE Sign OF
    '+' : Answer ← Number1 + Number2
    '–' : Answer ← Number1 - Number2
    '*' : Answer ← Number1 * Number2
    '/' : Answer ← Number1 / Number2
    OTHERWISE Answer ← 0
ENDCASE</pre>
</td><td>3</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>
Max **4** in total
Any **3** from:

To ensure no changes are made on input / <u>accuracy of transcription</u>
Because the details do not have fixed, values or lengths to validate
Because there is no clear set of rules that can be used for validation

Any **3** from:

The programmer could ask the contributor to type in each detail twice …
… and then check that both values are equal
… If they are not equal then the input should be rejected
The programmer could ask the contributor to check the details on the screen …
… and confirm that they are correct / same as the original
… or change them

</td><td>4</td></tr>
<tr><td>4(b)</td><td>
One mark for email and **one** mark for password
Email – check for @ / format check / no spaces /valid characters // presence check // length check (not more than 254 characters) // uniqueness check

Password – length check / numbers and letters etc. // uniqueness check not been used before // presence check
</td><td>2</td></tr>
</table>

Question	Answer	Marks
5	**One** mark per value and reason, max **3** Example 1.00 – boundary rejected//rejected (underweight) // out of range(1) 1.02 – normal // valid // accepted weight in range (1) 1.10 – abnormal // erroneous // invalid // rejected (overweight) (1)	**3**

Question	Answer	Marks
6(a)	Fields 5	**1**
6(b)	**One** mark description of new code that will allow more than 1000 values **One** mark for example matching candidate's description Example Use a new character instead of N TT345	**2**
6(c)		**4**

Field:	At Risk	Age in Years	Type	Map Position
Table:	TREES	TREES	TREES	TREES
Sort:				
Show:	☐	☐	☑	☑
Criteria:	True	>100		
or:				

One mark per correct column