Cambridge IGCSE

Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/23**

Paper 2 **May/June 2019**

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 2019 series for most Cambridge IGCSE™, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

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

This document consists of **8** printed pages.

Cambridge Assessment International Education

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
Section A		
1(a)	4 marks for: Data Structure(s) **max 2** • Arrays • Variable(s) for // data entry such as Item Code // index / constant for minimum number of items Further description **max 3** • Data type(s) one or more • Use(s) one or more • Name(s) one or more e.g. `Item_Code, Description, Price, Stock` • Sample data for appropriate arrays e.g. 1234, Notepad, 0.050, 1000 // Sample data for variable or constant e.g. 10 1 mark for: • All four names and suitable data samples	**5**
1(b)	• Keep a counter/number • Add one every time a new item code is added or • Keep a list of item codes used • Using a loop check item codes is not already in the list before a new item code is added	**2**
1(c)	• Check items in stock array • … with a value of 10 or less • Display these values in a different way e.g. in red	**3**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>1(d)</td><td>

Five from:
MP1 Enter the item code
MP2 …and check if valid
MP3 Enter number to purchase and check if less than or equal to number in stock for that item number
MP4 Display error message(s) if needed
MP5 Calculate price to pay
MP6 Using the number to purchase calculate discount …
MP7 Check for new customer AND $50 spend/returning customer eligibility …
MP8 … application of discount(s) if available
MP9 Check order confirmed …
MP10 … update appropriate stock level

Sample answer

```
REPEAT
   PRINT "Enter Item Code"
   INPUT ItemCode
   Found ← False
   Count ← 1
   REPEAT
      IF ItemCode = ItemList(Count} THEN Found ← True ENDIF
      Count ← Count + 1
   UNTIL Found OR Count = NoItems
UNTIL Found
REPEAT
   PRINT "Enter Number to purchase"
   INPUT Number
UNTIL Number > 0 and Number <= Stock(ItemCode)
PriceToPay  ← Number * Price (ItemCode)
IF Number >= 10 AND Number < 20 THEN PriceToPay ← PriceToPay * 0.95 ENDIF
IF Number >= 20 AND Number < 50 THEN PriceToPay ← PriceToPay * 0.9 ENDIF
IF Number >= 50 AND Number < 100 THEN PriceToPay ← PriceToPay * 0.75 ENDIF
IF Number >= 100 AND Number < 500 THEN PriceToPay ← PriceToPay * 0.65 ENDIF
IF Number >= 500 THEN PriceToPay ← PriceToPay * 0.5 ENDIF
PRINT "Enter Value of discount voucher, 0 if no voucher"
INPUT Voucher
PriceToPay ← PriceToPay – Voucher
PRINT "Price to pay is ", PriceToPay
PRINT "Order confirmed? Y/N"
INPUT Confirm
IF Confirm = "Y" THEN Stock(ItemCode) ← Stock(ItemCode) – Number ENDIF
```

</td><td>5</td></tr>
</table>

Question	Answer	Marks
1(e)	Explanation **Five** from: MP1 Search item arrays MP2 For stock level = 0 … MP3 … list these as sold out MP4 Check stock level at end of day with original stock level//search items sold arrays MP5 … if the same, display/store item code as not sold//if 0, display/store item code as not sold MP6 … update largest value if greater than existing greatest value … //use max function on items sold array … //other method to find highest number sold … MP7 … display this item number with a suitable message All programming statements used must be explained.	**5**

Question	Answer	Marks
	Section B	

2(a)	1 mark for each error identified + suggested correction
	`Count ← 1` should be `Count ← 0` or `Count >= 500` should be `Count > 500`
	`AND` should be `OR`
	`Reject ← Reject – 1` should be `Reject ← Reject + 1`
	`Reject ← Reject/100` should be `Reject ← Reject/5` or `Reject * 100 / 500`
		4

2(b)	MP1 Add `Accept ← 0` at start MP2 Add `ELSE Accept ← Accept + 1` after `THEN` <u>AND</u> `Over` and `Under` defined/position described OR Add `Accept ← Accept + 1` after `THEN` <u>AND</u> Replace `IF` statement with …`<= Over AND` …`>= UNDER`… /position described MP3 Add `Accept ← Accept/5` after `UNTIL` AND correct loop/position described MP4 ADD `IF Accept < 50 THEN PRINT "Less than 50% accepted"` at end	**4**

```
Accept ← 0
Count ← 1 // 0
Reject ← 0
Over ← 62
Under ← 58
REPEAT
   INPUT ItemWeight
   IF ItemWeight > Over OR ItemWeight < Under // IF
ItemWeight <= Over AND
   ItemWeight >= Under
      THEN
         Reject ← Reject + 1
      ELSE
         Accept ← Accept + 1//ELSE not required
   ENDIF
   Count ← Count + 1
UNTIL Count > 500  // >= 500
Accept ← Accept / 5

IF Accept < 50
   THEN
      PRINT "Less than 50% accepted"
ENDIF
```

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>3</td><td>
<table>
<tr><th>TreadReject</th><th>Count</th><th>Depth</th><th>OUTPUT</th></tr>
<tr><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>2</td><td>1.7</td><td></td></tr>
<tr><td></td><td>3</td><td>1.9</td><td></td></tr>
<tr><td>1</td><td>4</td><td>1.4</td><td></td></tr>
<tr><td></td><td>5</td><td>1.8</td><td></td></tr>
<tr><td></td><td>6</td><td>2.0</td><td></td></tr>
<tr><td></td><td></td><td></td><td>Car is potentially roadworthy</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
</table>
<table>
<tr><th>TreadReject</th><th>Count</th><th>Depth</th><th>OUTPUT</th></tr>
<tr><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td>1</td><td>2</td><td>1.2</td><td></td></tr>
<tr><td></td><td>3</td><td>1.9</td><td></td></tr>
<tr><td>2</td><td>4</td><td>1.4</td><td></td></tr>
<tr><td></td><td>5</td><td>1.8</td><td></td></tr>
<tr><td></td><td>6</td><td>2.4</td><td></td></tr>
<tr><td></td><td></td><td></td><td>Car is not roadworthy</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
</table>
1 mark for each correct pairs of columns.</td><td>4</td></tr>
</table>

Question	Answer	Marks
4	| Statements | Validation | Verification | | Range check | ✓ | | | Double entry | | ✓ | | Check digit | ✓ | | | Presence check | ✓ | | 1 mark for each correct row	**4**

Statements	Validation	Verification
Range check	✓	
Double entry		✓
Check digit	✓	
Presence check	✓	

Question	Answer	Marks
5	• `FOR … TO … NEXT` • fixed number of repetitions • `REPEAT … UNTIL` • always executed // condition tested at end • `WHILE … DO … ENDWHILE` • may not be executed // condition tested at beginning	**6**

Question	Answer	Marks
6(a)	Number is repeated/not unique	**1**
6(b)	**Three** from: Train number not displayed Departure time before 8:30 Criteria of =Y for Platform not required/incorrect All late trains will be shown/the condition should be on the line above 4 marks, one mark for each correct column	**7**

Field:	Train Number	Platform	Departure Time	Status
Table:	TRAIN	TRAIN	TRAIN	TRAIN
Sort:				
Show:	☑	☑	□	□
Criteria:			> 08:30	= "Late"
or:				