Cambridge IGCSE

Cambridge Assessment International Education
Cambridge International General Certificate of Secondary Education

COMPUTER SCIENCE **0478/23**

Paper 2 **October/November 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 International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the October/November 2017 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

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.

Cambridge Assessment International Education

Question	Answer	Marks

1(a)(i)	1 mark per bullet:
	At least one array declaration
	At least one array has an appropriate name
	All arrays with appropriate names
	Many correct answers, they must be meaningful. These are examples only.
	`Array_2Seater[ ]`
	`Array_4Seater[ ]`
	`Array_Historic[ ]`
		3

1(a)(ii)	1 mark per bullet: Name of variable Purpose of variable Name of constant Purpose of constant Many correct answers, they must be meaningful. These are examples only. Variable `NumFlights` to store the number of flights in a day Constant `FlightCost2Seat30` to store the cost of a 30 minute flight in a 2 seater plane	**4**

Question	Answer	Marks

1(b)	Any **five** from:
	Prompt for plane
	Input plane
	Prompt for another input length of flight along with the input.
	Attempt at calculation of maximum number of flights in a day
	Using correct values for maximum number of flights (from calculation or otherwise)
	Calculation/determination of cost of a single flight for selected plane and duration
	Calculation of income that can be generated for one combination of plane and flight
	Output of total possible income for one combination of plane and flight with message(s)
	
	Algorithm example:
	
	`OUTPUT "Please Enter Type of Plane"`
	`OUTPUT "1: 2 Seater"`
	`OUTPUT "2: 4 Seater"`
	`OUTPUT "3: Historic"`
	`INPUT PlaneType`
	`OUTPUT "Please Enter Length of Flight"`
	`INPUT FlightLength`
	`CASE FlightLength of`
	`  30: OUTPUT "Maximum number of flights is 10"`
	`  60: OUTPUT "Maximum number of flights is 6"`
	`OTHERWISE OUTPUT "Invalid length of flight"`
	`ENDCASE`
	
	`CASE PlaneType of`
	`  1: Price30 ← 100; Price60 ← 150`
	`  2: Price30 ← 120; Price60 ← 200`
	`  3: Price30 ← 120; Price60 ← 500`
	`OTHERWISE OUTPUT "Invalid type of plane"`
	`ENDCASE`
	
	`CASE FlightLength of`
	`  30: OUTPUT "Total Possible Income is ", Price30 * 10`
	`  60: OUTPUT "Total Possible Income is ", Price60 * 6`
	`ENDCASE`
		5

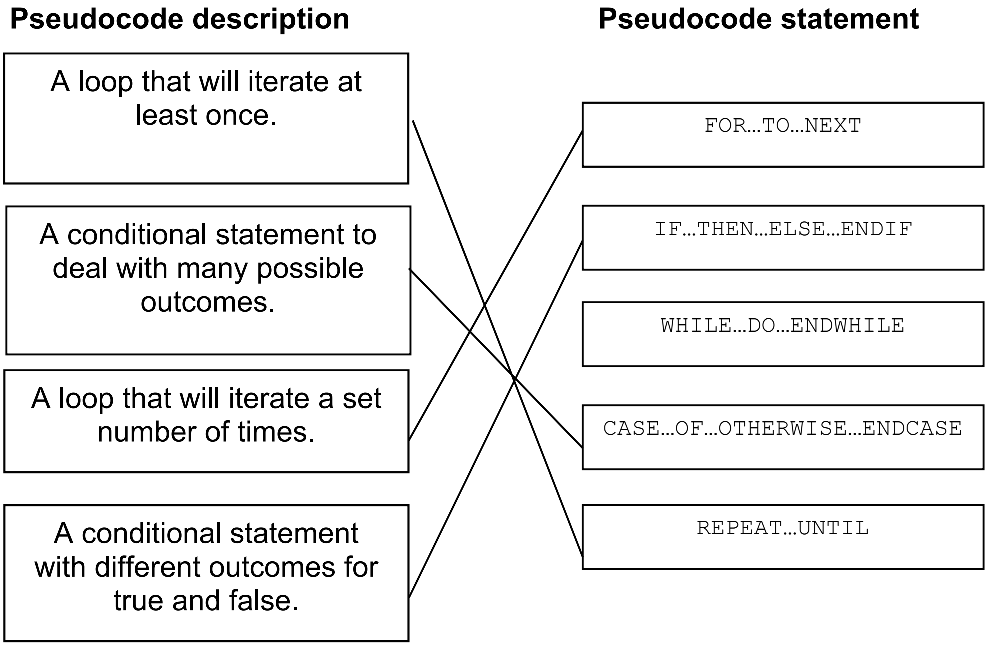
1(c)	1 mark for each correct point related to the inputs for **Task 1** Description of how the program would validate the input Description/identification of input(s) Type of validation check Checking inputs against stored data/maxima/correct data Dry-running the program Use of test data Identification of types of test data Example(s) of test data	4

Question	Answer	Marks
1(d)	Any **four** from: Input timeslot Check 3 types of plane Methodology for checking time slot Identify any planes available Output plane(s) available Output if no planes available	**4**

Question	Answer	Marks
2	1 mark for each error identified plus suggested correction (the corrected lines must be written in full) Line 4 **correct line** `WHILE Number <= 99 OR Number > 1000` Line 7 **correct line** `Num[Index] = Number` Line 9 **correct line** `NEXT (Index)` Line 10 **correct line** `PRINT Count`	**4**

Question	Answer	Marks
3(a)	1 mark per bullet: Validation checks whether data to be entered is possible/sensible // computer check Verification checks that data entered is the data that was intended to be entered // can be a human check // matches the source	**2**
3(b)	1 mark for each valid point **Either** Double Entry // suitable practical example the data will be entered twice compared by the computer or by a human if a discrepancy is found, the data entry operator is asked to re-enter the data **Or** Visual Verification // suitable practical example the data will be compared to the source ‘document’ compared by a human if a discrepancy is found, the data is re-entered	**2**

Question	Answer	Marks
3(c)	1 mark for explanation and 1 mark for an expansion Library routine is a list of instructions // block of code // subroutine … that is used often … … which is given a name … and which can be called from other programs Library routines make writing programs easier and faster as the code is already written Library routines make program testing easier as the code has already been tested and debugged	2

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>1 mark for each correct line

Pseudocode description
Pseudocode statement
A loop that will iterate at least once.
A conditional statement to deal with many possible outcomes.
A loop that will iterate a set number of times.
A conditional statement with different outcomes for true and false.
FOR…TO…NEXT
IF…THEN…ELSE…ENDIF
WHILE…DO…ENDWHILE
CASE…OF…OTHERWISE…ENDCASE
REPEAT…UNTIL

</td><td>4</td></tr>
<tr><td>4(b)</td><td>1 mark per bullet:
Appropriate loop controls
Read from array
Print from array (the last two points can be in one statement)

Note reading and printing <b>MUST</b> be within the same loop

Example algorithm:<pre>
Count ← 0
WHILE Count < 50 DO
   OUTPUT Name[Count]
   Count ← Count + 1
ENDWHILE
</pre></td><td>3</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td>
<table>
<tr><th>Flag</th><th>Count</th><th>Name[1]</th><th>Name[2]</th><th>Name[3]</th><th>Name[4]</th><th>Temp</th></tr>
<tr><td></td><td></td><td>Jamal</td><td>Amir</td><td>Eve</td><td>Tara</td><td></td></tr>
<tr><td>0</td><td>1</td><td>Amir</td><td>Jamal</td><td>Eve</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>1</td><td>2</td><td>Amir</td><td>Jamal</td><td>Eve</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>1</td><td>3</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>1</td><td>4</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>0</td><td>1</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>0</td><td>2</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>0</td><td>3</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td>0</td><td>4</td><td>Amir</td><td>Eve</td><td>Jamal</td><td>Tara</td><td>Jamal</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
(1 Mark) (1 Mark) (--------1 Mark--------) (--------1 Mark--------) (1 Mark)
</td><td>5</td></tr>
<tr><td>5(b)</td><td>1 mark per bullet:
Sorting the names
Ascending order / A to Z / lowest to highest / Alphabetic order</td><td>2</td></tr>
</table>

Question	Answer	Marks
6(a)	1 mark for any **sensible appropriate** field name 1 mark for data type, purpose + example data Example 1: Field Name: SPECIESID Data Type: Alphanumeric Purpose: Primary key Example Data: SP06583 Example 2: Field name: NUMBER Data Type: Integer Purpose: To record how many of that species there are at the park Example Data: 30	2

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>6(b)</td><td>
<table>
<tr><td>Field:</td><td>Species</td><td>Classification</td><td>Diet</td><td>Legs</td><td></td></tr>
<tr><td>Table:</td><td>LIVESTOCK</td><td>LIVESTOCK</td><td>LIVESTOCK</td><td>LIVESTOCK</td><td></td></tr>
<tr><td>Sort:</td><td>Ascending/ Descending</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Show:</td><td>☑</td><td>□</td><td>□</td><td>□</td><td>□</td></tr>
<tr><td>Criteria:</td><td></td><td>“Mammal”</td><td>“Herbivore”</td><td>4</td><td></td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>(1 Mark)</td><td>(1 Mark)</td><td>(1 Mark)</td><td>(1 Mark)</td><td></td></tr>
</table>
1 mark per completely correct column.</td><td>4</td></tr>
</table>