Cambridge Assessment International Education

Cambridge IGCSE™

CANDIDATE NAME									
CENTRE NUMBER						CANDIDATE NUMBER				

8130505114

COMPUTER SCIENCE **0478/22**

Paper 2 Problem-solving and Programming **October/November 2021**

1 hour 45 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- **Do not attempt Tasks 1, 2 and 3** in the copy of the pre-release material on page 2; these are for information only.
- Use a black or dark blue pen. You may use an HB pencil for any diagrams or graphs.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 50.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

Section A

You are advised to spend no longer than 40 minutes answering this section.

Here is a copy of the pre-release material.

DO NOT attempt Tasks 1, 2 and 3 now.

Use the pre-release material and your experience from attempting the following tasks before the examination to answer Question 1.

Pre-release material

An integrated transport system has been designed to reduce the need for privately owned vehicles. A vehicle is booked to take a passenger from home to a start station, from where they will travel to an end station. A vehicle at the end station will take the passenger to their destination. Each stage of the journey has a price code to represent the distance travelled. The prices for each stage are shown:

Home to start station	
Code	**Price ($)**
C1	1.50
C2	3.00
C3	4.50
C4	6.00
C5	8.00

Start station to end station	
Code	**Price ($)**
M1	5.75
M2	12.50
M3	22.25
M4	34.50
M5	45.00

End station to destination	
Code	**Price ($)**
F1	1.50
F2	3.00
F3	4.50
F4	6.00
F5	8.00

To book a journey, a passenger will enter a code for each stage and the start time of their journey. The total price is calculated by adding together the price for each of the three stages. The total price will be reduced by 40% when the start time of the journey is after 10:00.

Write and test a program or programs for the integrated transport booking system.
- Your program or programs must include appropriate prompts for the entry of data; data must be validated on entry.
- Error messages and other output need to be set out clearly and understandably.
- All variables, constants and other identifiers must have meaningful names.

You will need to complete these **three** tasks. Each task must be fully tested.

Task 1 – setting up the booking system
Write a program to set up arrays to record the following:
- codes and prices for each of the three stages
- passenger accounts that include a unique passenger account number and name
- bookings that include a unique passenger account number, a start time of the journey, a code for each stage of the journey, and a unique booking number for the journey.

Store the data for the code and price for each stage.

Task 2 – using the booking system
Extend **Task 1** to achieve the following:
- Allow passengers to open an account by generating a unique passenger account number and storing it along with their name in the arrays.
- Allow passengers to make a booking by first entering their unique passenger account number, the start time of their journey, and a code for each stage of their journey. Check if the passenger account number already exists.
- Generate a unique booking number for the journey.
- Calculate the total price of the journey, without any discount, and store the journey details.

Task 3 – applying a discount and checking the entry
Extend **Task 2** to check the start time of the journey and if it is after 10:00, apply a 40% discount to the total price.
Display the total price and booking details for the passenger to check, and allow them to either confirm the details are correct or start again.

1 All variables, constants and other identifiers must have meaningful names.

(a) (i) Identify **one** variable you could have used for **Task 2** and state its use.

Variable .....

Use .....

.....

.....
[2]

(ii) Describe the arrays you could have used in **Task 1**. Include the name, data type, use and sample data for each array.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

(b) Describe how you could validate the data entry for the input of the codes for the different stages of the journey in **Task 2**.

.....

.....

.....

.....

.....

..... [3]

(c) Write an algorithm to show how your program carries out **Task 3**, using either pseudocode, programming statements or a flowchart. Assume **Tasks 1** and **2** have already been completed.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.... [6]

(d) Explain how your program could be changed to count and store the number of bookings made by each passenger. Then, after ten bookings have been made by a passenger, apply an additional 10% discount to every future booking. Any programming statements used in your answer must be fully explained.

.... [4]

Section B

2 Tick (✓) one box in each row to identify if the statement is about validation, verification or neither.

Statement	Validation (✓)	Verification (✓)	Neither (✓)
a check where data is re-entered to make sure no errors have been introduced during data entry			
an automatic check to make sure the data entered has the correct number of characters			
a check to make sure the data entered is sensible			
a check to make sure the data entered is correct			

[3]

3 A program checks that the data entered is between 1 and 100 inclusive.

Identify **one** piece of normal, extreme and erroneous test data for this program, and give a reason for each.

Normal test data .....

Reason .....

.....

.....

Extreme test data .....

Reason .....

.....

.....

Erroneous test data .....

Reason .....

.....

.....

[6]

4 The pseudocode algorithm should work as a calculator and output the result.

```
1   Continue ← 1
2   WHILE Continue = 0
3     OUTPUT "Enter 1 for +, 2 for -, 3 for * or 4 for /"
4     INPUT Operator
5     OUTPUT "Enter the first value"
6     INPUT Value1
7     OUTPUT "Enter the second value"
8     OUTPUT Value2
9     IF Operator
10      1: Answer ← Value1 + Value2
11      2: Answer ← Value1 - Value2
12      3: Answer ← Value1 * Value2
13      4: Answer ← Value1 / Value2
14    ENDCASE
15    OUTPUT "The answer is ", Value1
16    OUTPUT "Do you wish to enter more values (Yes or No)?"
17    INPUT MoreValues
18    IF MoreValues = "No"
19      THEN
20        Continue ← 1
21    ENDIF
22  UNTIL Continue = 0
```

(a) Find the **five** errors in the pseudocode and suggest a correction for each error.

Error 1 .....

Correction .....

.....

Error 2 .....

Correction .....

.....

Error 3 .....

Correction .....

.....

Error 4 .....

Correction .....

.....

Error 5 .....

Correction .....

.....

[5]

(b) The algorithm needs changing to allow only the numbers 1, 2, 3, or 4 to be entered for the input variable `Operator`.

Write the pseudocode to perform this task and state where in the algorithm it would be located.

Pseudocode .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Location in algorithm .....

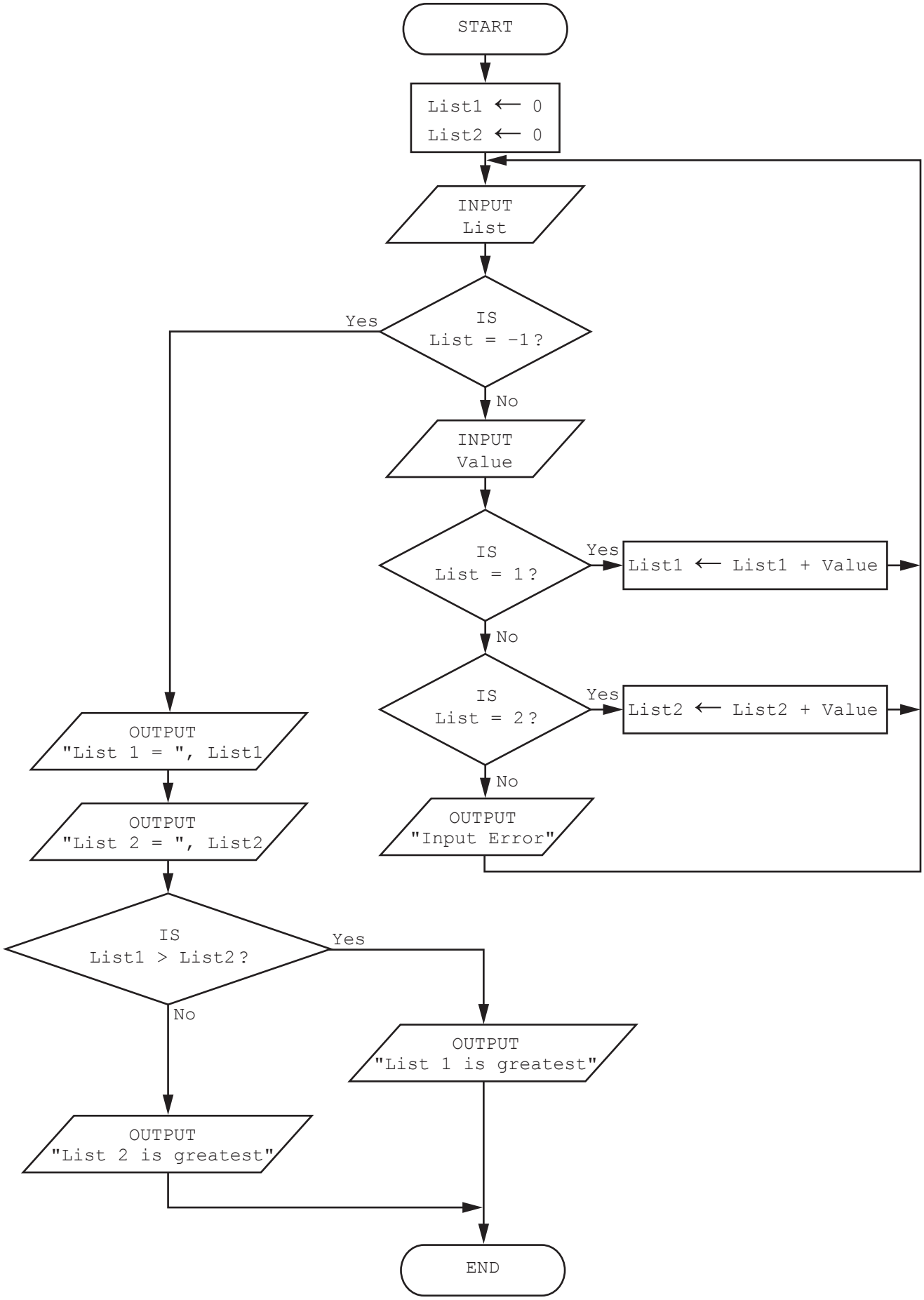
.....

.....

[5]

5 The flowchart represents an algorithm.

The algorithm will terminate if –1 is entered at the `List` input.



Complete the trace table for the algorithm using this input data:

2, 77, 2, 16, 1, 35, 2, –7, 5, 18, 1, 11, 1, 12, 2, 20, –1, 18

List	Value	List1	List2	OUTPUT

[5]

6 A pet supplier uses the database table, STOCK, to keep records of its products for pets.

The fields are:

Field name	Description
ProductID	code to identify the product
ProductName	name of product
ProductDescription	information about the product
Animal	type of animal the product is for, e.g. cat, bird, horse
ProductType	type of product, e.g. food, toy, medicine
InStock	whether the product is in stock or **not**

(a) (i) Identify the field that could have a Boolean data type.

..... [1]

(ii) Identify the field that should be used as the primary key.

..... [1]

(b) Complete the query-by-example grid to output the products intended for a cat that are in stock. Display only the primary key and the name of the products. The output should be sorted by the primary key.

Field:					
Table:					
Sort:					
Show:	☐	☐	☐	☐	☐
Criteria:					
or:					

[4]

Cambridge Assessment International Education

Cambridge IGCSE™

COMPUTER SCIENCE **0478/22**

Paper 2 **October/November 2021**

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 2021 series for most Cambridge IGCSE™, Cambridge International A and AS Level components and some Cambridge O Level components.

[Turn over

PUBLISHED

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)(i)	**One** mark per point • Variable PassengerID// StartStage • Use Storing the unique ID number of the passenger// Storing/inputting the start stage of the journey	**2**
1(a)(ii)	**One** mark per point MP1 Name of array MP2 Data type of array MP3 Sample data for array MP4 Use of array MP5 At least two complete arrays with all of the above **Array name** **Data type** **Sample data** **Use** `JourneyStage1` string `C1` to store the code for the home to start station `PriceStage1` real 1.50 to store the price of first stage of the journey	**5**
1(b)	**One** mark per bullet point MP1 Use of validation check, e.g. range check, type check, presence check, length check, format check MP2 Use of conditional statement to check if the validation fails ... MP3 … a re-entry is requested MP4 Use of loop to repeat the process until an acceptable answer is input MP5 More than one appropriate validation check used / described.	**3**
1(c)	Any **six** from: MP1 Conditional statement to check departure time against 10:00 MP2 … calculate 40% discount // calculate 60% of the original price MP3 … calculate discounted total price MP4 Output the discounted total price MP5 Output the **booking details** with suitable messages MP6 Input with prompt for passenger confirmation … MP7 … attempt at action following the confirmation input MP8 Repeating **booking data entry** if incorrect MP9 Re-checking journey details for correctness	**6**

Question	Answer	Marks

1(c)	**Example answer**
	`// Tasks 1 and 2 completed`
	`IF CollectedTime[Index] > 10:00`
	`  THEN`
	`    JourneyCost[Index] ← JouneyCost[Index] * 0.6`
	`ENDIF`
	`PRINT "Your journey cost is: ", JourneyCost[Index]`
	`PRINT "Your journey details are: ", PassengerID[Index], JourneyTime[Index],`
	`       JourneyCodes[Index], JourneyID[Index]`
	`PRINT "Are these details correct? (Y or N)"`
	`INPUT Correct`
	`IF Correct = "N"`
	`  THEN`
	`    WHILE Correct = "N"`
	`      PRINT "Re-enter your journey details"`
	`      PRINT "Correct passenger ID "`
	`      INPUT PassengerID[Index]`
	`      PRINT "Correct journey time "`
	`      INPUT JourneyTime[Index]`
	`      PRINT "Correct journey codes "`
	`      INPUT JourneyCodes[Index]`
	`      PRINT "Your revised journey details are: ", PassengerID[Index],`
	`             JourneyTime[Index], JourneyCodes[Index]`
	`      PRINT "Are these details correct? (Y or N)"`
	`      INPUT Correct`
	`    ENDWHILE`
	`ENDIF`
	`//Program continues`	

1(d)	**Explanation of how** each of the following could be done Any **four** from: MP1 Declaring/using a counter to store the number of bookings for each passenger MP2 Updating the counter for the number of bookings made by each passenger MP3 Attempt to check the number of bookings … MP4 … for the correct condition e.g. if the number of bookings is more than 10 / equal to 10 MP5 Apply the extra discount to the total price of future journeys	**4**

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td></td><td>Section B</td><td></td></tr>
<tr><td>2</td><td>One mark for two correct rows
Two marks for three correct rows
Three marks for four correct rows.

<table>
<tr><th>Statement</th><th>Validation (✓)</th><th>Verification (✓)</th><th>Neither (✓)</th></tr>
<tr><td>a check where data is re-entered to make sure no errors have been introduced during data entry</td><td></td><td>✓</td><td></td></tr>
<tr><td>an automatic check to make sure the data entered has the correct number of characters</td><td>✓</td><td></td><td></td></tr>
<tr><td>a check to make sure the data entered is sensible</td><td>✓</td><td></td><td></td></tr>
<tr><td>a check to make sure the data entered is correct</td><td></td><td></td><td>✓</td></tr>
</table></td><td>3</td></tr>
</table>

Question	Answer	Marks
3	**One** mark per bullet point **Normal test data** • Test data e.g. 50 (allow any number between 1 and 100 inclusive) • Reason Data that is within range and should be **accepted** **Extreme test data** • Test data 100 / 1 • Reason Data at the **maximum** / **minimum** end of the range and should be **accepted** **Erroneous test data** • Test data e.g. 300 (allow anything that isn't between 1 and 100 inclusive, including other data types) • Reason Data outside the range that should be **rejected**	**6**

Question	Answer	Marks
4(a)	**One** mark for error identified and suggested correction (**Max three**) Line 8 `OUTPUT Value2` – should be `INPUT Value2` Line 9 `IF Operator` – should be `CASE OF Operator` Line 15 `OUTPUT "The answer is ", Value1` – should be `Answer` The loop may be corrected using a number of alternative methods: **One** mark for error identified and suggested correction (**Max two**) **Method 1** Line 1 `Continue ← 1` should be `Continue ← 0` Line 22 `UNTIL Continue = 0` should be `ENDWHILE //` Line 2 `WHILE Continue = 0` should be `REPEAT` **and** Line 22 `UNTIL Continue = 0` should be `Until Continue = 1` **OR** **Method 2** Line 2 `WHILE Continue = 0` should be `REPEAT` Line 20 `Continue ← 1` should be `Continue ← 0 //` Line 1 `Continue ← 1` should be `Continue ← 0` **and** Line 22 `UNTIL Continue = 0` should be `Until Continue = 1` **OR** **Method 3** Line 2 `WHILE Continue = 0` should be `WHILE Continue = 1` Line 20 `Continue ← 1` should be `Continue ← 0` **and** Line 22 `UNTIL Continue = 0` should be `ENDWHILE`	**5**

Question	Answer	Marks
4(a)	**Corrected algorithm example 1** 1 Continue ← 0 2 WHILE Continue = 0 (DO) 3 OUTPUT "Enter 1 for +, 2 for -, 3 for * or 4 for /" 4 INPUT Operator 5 OUTPUT "Enter the first value" 6 INPUT Value1 7 OUTPUT "Enter the second value" 8 INPUT Value2 9 CASE OF Operator 10 1: Answer ← Value1 + Value2 11 2: Answer ← Value1 - Value2 12 3: Answer ← Value1 * Value2 13 4: Answer ← Value1 / Value2 14 ENDCASE 15 OUTPUT "The answer is ", Answer 16 OUTPUT "Do you wish to enter more values (Yes or No)?" 17 INPUT MoreValues 18 IF MoreValues = "No" 19 THEN 20 Continue ← 1 21 ENDIF 22 ENDWHILE	

Question	Answer	Marks
4(a)	**Corrected algorithm example 2**	

```
1 Continue ← 1
2 REPEAT
3   OUTPUT "Enter 1 for +, 2 for -, 3 for * or 4 for /"
4   INPUT Operator
5   OUTPUT "Enter the first value"
6   INPUT Value1
7   OUTPUT "Enter the second value"
8   INPUT Value2
9   CASE OF Operator
10    1: Answer ← Value1 + Value2
11    2: Answer ← Value1 - Value2
12    3: Answer ← Value1 * Value2
13    4: Answer ← Value1 / Value2
14  ENDCASE
15  OUTPUT "The answer is ", Answer
16  OUTPUT "Do you wish to enter more values (Yes or No)?"
17  INPUT MoreValues
18  IF MoreValues = "No"
19    THEN
20      Continue ← 0
21  ENDIF
22 UNTIL Continue = 0
```

Question	Answer	Marks

4(b)	**One** mark per bullet
	MP1 Appropriate loop (begin and end) / otherwise selection
	MP2 Testing both ends of condition
	MP3 Suitable message
	MP4 Input/re-input
	
	`WHILE Operator < 1 OR Operator > 4 (DO)`
	`  OUTPUT "Enter 1, 2, 3 or 4"`
	`  INPUT Operator`
	`ENDWHILE`
	
	Alternative answer
	`REPEAT`
	`  IF Operator < 1 OR Operator > 4`
	`  THEN`
	`  OUTPUT "Enter 1, 2, 3 or 4"`
	`  INPUT Operator`
	`  ENDIF`
	`UNTIL Operator >= 1 AND Operator <= 4`
	
	One mark
	
	After line 4 / between lines 2 and 5
		5

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5</td><td>One mark for each correct column
<table>
<tr><th>List</th><th>Value</th><th>List1</th><th>List2</th><th>OUTPUT</th></tr>
<tr><td></td><td></td><td>0</td><td>0</td><td></td></tr>
<tr><td>2</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>77</td><td></td><td>77</td><td></td></tr>
<tr><td>2</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>16</td><td></td><td>93</td><td></td></tr>
<tr><td>1</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>35</td><td>35</td><td></td><td></td></tr>
<tr><td>2</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>-7</td><td></td><td>86</td><td></td></tr>
<tr><td>5</td><td></td><td></td><td></td><td></td></tr>
</table></td><td>5</td></tr>
</table>

Question	Answer	Marks

5

List	Value	List1	List2	OUTPUT
	18			Input Error
1				
	11	46		
1				
	12	58		
2				
	20		106	
-1				List 1 = 58
				List 2 = 106
				List 2 is greatest

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>6(a)(i)</td><td>InStock</td><td>1</td></tr>
<tr><td>6(a)(ii)</td><td>ProductID</td><td>1</td></tr>
<tr><td>6(b)</td><td>One mark for correct fieldnames
One mark for correct table names and show fields
One mark for correct sort
One mark for correct search criteria in all columns

<table>
<tr><td>Field:</td><td>ProductID</td><td>ProductName</td><td>Animal</td><td>InStock</td><td></td></tr>
<tr><td>Table:</td><td>STOCK</td><td>STOCK</td><td>STOCK</td><td>STOCK</td><td></td></tr>
<tr><td>Sort:</td><td>Ascending</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></td><td>=“cat”</td><td>=Yes</td><td></td></tr>
<tr><td>or:</td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td><td>4</td></tr>
</table>