7 Algorithm design and problem-solving – Error Checking

ANSWERS

Question	Answer	Marks
2	1 mark for each error identified and suggested correction (the corrected code must be written in full) *Line* 2 *Correct code* `Counter = 0` (1) *Line* 7 *Correct code* `Total = Total + Number // Number + Total` (1) *Line* 8 *Correct code* `Counter = Counter + 1 // 1 + Counter` (1) *Line* 10 *Correct code* `Average = Total / Counter //` `Average = Total / 50` (1)	4

2 **One** mark for each error identified + suggested correction
line 5 or `IF Num < 0`: this should read `IF Num > 0 (THEN Total = Total + Num)`

line 6 or (`IF Num > 0`) `THEN Counter = Counter + 1`:
this should read (`IF Num > 0 THEN`) `Poscount = Poscount + 1`

line 7 `Average = Total/Poscount`: this should come after the end of the repeat loop

line 9 or `PRINT Num`: this should read `PRINT Average` [4]

2 **(a)** 1 mark for each change
- Line 2: `OutRange = 0`
- Line 6: should be `OutRange = OutRange + 1`
- Line 7: not needed
- Line 8: `NEXT X` should be `NEXT Count` / Line 3: `FOR Count = 1 TO 10` should be `FOR X = 1 TO 10` [4]

(b)

Number	Within range (✓)	Outside range (✓)	Reason
10		✓	Range greater than 10, so 10 not included .....
20		✓	Range less than 20, so 20 not included .....

[4]

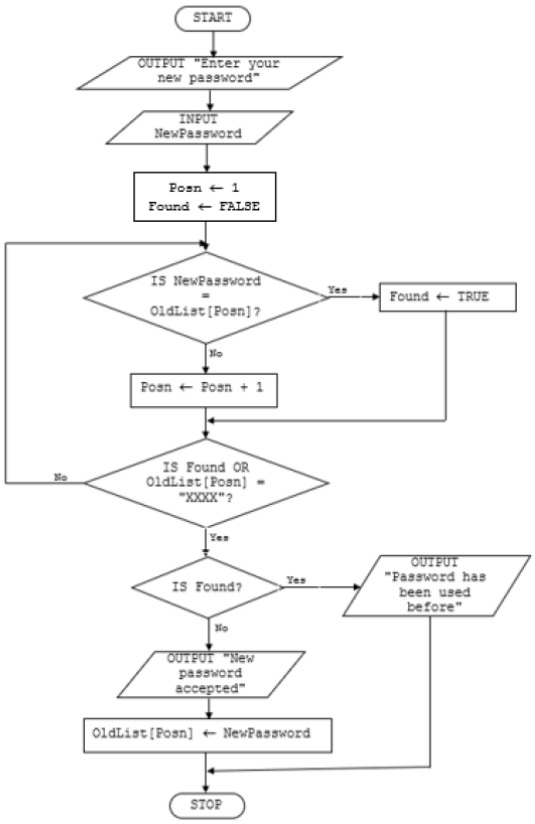
7 Algorithm design and problem-solving – Error Checking
ANSWERS

Question	Answer	Marks

4(a)	**One** mark per mark point, max **four**
	
	MP1 Line 01 / `DECLARE City ARRAY[1:50, 1:2] OF BOOLEAN` should be `DECLARE City : ARRAY[1:50, 1:2] OF STRING` Line 05 / `IF` should be `REPEAT`
	MP2 Line 07 / `INPUT City[Count, 2]` should be `INPUT City[Count, 1]`
	MP3 Line 11 / `UNTIL Count = 50` // Line 04 / `Count ← 1` **AND** Line 10 / `Count ← Count + 1` should be `UNTIL Count = 51` / `UNTIL Count > 50` // Line 04 / `Count ← 0` **AND** move Line 10 to beginning of loop / Line 06
	MP4 Line 12 / `FOR Out ← 1 TO 1` should be `FOR Out ← 1 TO 50`
	
	Correct algorithm:
	`01 DECLARE City : ARRAY[1:50, 1:2] OF STRING`
	`02 DECLARE Count : INTEGER`
	`03 DECLARE Out : INTEGER`
	`04 Count ← 1`
	`05 REPEAT`
	`06  OUTPUT "Enter the name of the city"`
	`07  INPUT City[Count, 1]`
	`08  OUTPUT "Enter the name of the country"`
	`09  INPUT City[Count, 2]`
	`10  Count ← Count + 1`
	`11 UNTIL Count > 50`
	`12 FOR Out ← 1 TO 50`
	`13  OUTPUT "The city ", City[Out, 1], " is in ", City[Out, 2]`
	`14 NEXT Out`
		4

Question	Answer	Marks
5(a)	**One** mark for each error identified and correction given • Line `06 Password` should be `NewPassword` • Line `11 AND` should be `OR` • Line `16 INPUT` should be `OUTPUT`	**3**

7 Algorithm design and problem-solving – Error Checking
ANSWERS

Question	Answer	Marks
5(b)		**Max six** marks from: **Max four** from: **one** mark for data entry with message **one** mark for initialisation **one** mark for checking list // decision box comparing input with array **one** mark for updating // updating the two variables position and found **one** mark for loop control // second decision box **one** mark for setting new password to position in list **one** mark for outputs // two outputs **Two marks**: **one mark** for correct use of flow chart symbols **one mark** for correct use arrows and labels	**6**

Question	Answer	Marks

5(a)	**One** mark per mark point, max **four**
	
	• Line 04 / `IF Number < 0`
	should be `IF Number > 0`
	
	• Line 10 / `Exit ← 1` // Line 01/ `Exit ← 1` **and** Line 02 / `WHILE Exit <> 0`
	should be `Exit ← 0` // should be `Exit ← 0` **and** `WHILE Exit = 0`
	
	• Line 13 / `ENDIF`
	should be `ENDWHILE`
	
	• Line 14 / `OUTPUT "The total value of your numbers is ", Number`
	should be `OUTPUT "The total value of your numbers is ", Total`
	
	Correct algorithm:
	
	`01 Exit ← 1`
	`02 WHILE Exit <> 0 DO`
	`03  INPUT Number`
	`04  IF Number > 0`
	`05  THEN`
	`06  Total ← Total + Number`
	`07  ELSE`
	`08  IF Number = 0`
	`09  THEN`
	`10  Exit ← 0`
	`11  ENDIF`
	`12  ENDIF`
	`13 ENDWHILE`
	`14 OUTPUT "The total value of your numbers is ", Total`
		4

7 Algorithm design and problem-solving – Error Checking
ANSWERS

Question	Answer	Marks
5(b)	**One** mark per mark point, max **four** • Initialise a new (counting) variable • … `Count ← 0` // to count the acceptable numbers • Insert a counting statement between lines 05 and 07 • … `Count ← Count + 1` • Add a new output after the loop/after line 13 / at the end (of the program) • … `OUTPUT Count`	4

Question	Answer	Marks

6(a)	**One** mark per mark point, max **four**
	
	• Line 01 / `Counter ← 100`
	should be `Counter ← 0`
	
	• Line 03 / `While Counter > 100 DO`
	should be `While Counter < 100 DO`
	
	• Line 07 / `Total ← Total + Counter`
	should be `Total ← Total + Number`
	
	• Line 09 / `ENDCASE`
	should be `ENDIF`
	
	Correct algorithm
	`01 Counter ← 0`
	`02 Total ← 0`
	`03 WHILE Counter < 100 DO`
	`04  INPUT Number`
	`05  IF Number > 0`
	`06  THEN`
	`07  Total ← Total + Number`
	`08  Counter ← Counter + 1`
	`09  ENDIF`
	`10 ENDWHILE`
	`11 OUTPUT "The total value of your numbers is ", Total`
	`12 OUTPUT "The average value of your numbers is ", Total / 100`
		4

Question	Answer	Marks
6(b)	**One** mark per mark point, max **five** MP1 replace line 03 MP2 with `FOR` MP3 … with limits 0 to 99 / 1 to 100 MP4 replace line 05 to check if `Number` is not positive MP5 … (if `Number` is not positive) insert a validation and re-input routine between lines 06 and 07 … MP6 … that will repeat until a positive value is entered MP7 remove the counter update / line 08 MP8 replace line 10 / `ENDWHILE` with `NEXT`	5

Question	Answer	Marks
7(a)	• `07` • `04/12` or `16/18` • `02/20`	3
7(b)	**One** mark for each error identified and correction • Line `07 Total ← Total + Number * Counter` should be `Total ← Total + Number[Counter] * Counter` • Line `08 IF Number[Counter] = 0` should be `IF Number[Counter] = -1` // should be `IF Number[Counter] < 0` • Line `16 FOR Counter ← 0 TO 5` should be `FOR Counter ← 1 TO 5`	3

7 Algorithm design and problem-solving – Error Checking
ANSWERS

Question	Answer	Marks

6(a)	**One** mark for each error identified and correction
	• Line `05 OUTPUT UsefulEnergyOut` should be `INPUT UsefulEnergyOut`
	• Line `06 IF TotalEnergyIn <> -1 AND UsefulEnergy <> -1` should be:
	`IF TotalEnergyIn <> -1 AND UsefulEnergyOut <> -1`
	• Line `11 UNTIL TotalEnergyIn <> -1 OR UsefulEnergyOut <> -1` should be:
	`UNTIL TotalEnergyIn = -1 OR UsefulEnergyOut = -1`
		3

6(b)	**One** mark for checking for >= 92
	One mark for outputting "**A-rated**" only if the condition is met
	For example
	`IF Efficiency >= 92`
	`  THEN`
	`    OUTPUT "A-rated"`
	`ENDIF`
		2