CAMBRIDGE INTERNATIONAL EXAMINATIONS

Cambridge Ordinary Level

MARK SCHEME for the October/November 2015 series

2210 COMPUTER SCIENCE

2210/23 Paper 2, maximum raw mark 50

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 will not enter into discussions about these mark schemes.

Cambridge is publishing the mark schemes for the October/November 2015 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

® IGCSE is the registered trademark of Cambridge International Examinations.

CAMBRIDGE
International Examinations

Section A

1 (a) (i) Any **two** variables with matching uses, **one** mark for the variable and **one** mark for the matching use. The variables and the matching uses must relate to the tasks on the exam paper. There are many possible correct answers these are examples only.

Variable 1 – `Counter(: INTEGER)`
Use – to use as a loop counter when entering the temperatures
Variable 2 – `HighestTemperature(: REAL)`
Use – to store the highest temperature recorded [4]

(ii) Any **two** constant with matching uses, **one** mark for the constant and **one** mark for the matching use. The constants and the matching uses must relate to the tasks on the exam paper. There are several possible correct answers these are examples only.

Constant 1 – `MinAppartmentTemperature = 21.5/22`
Use – to keep the temperature when the air-conditioning should be switched off
Constant 2 – `MaxAppartmentTemperature = 24.5/24`
Use – to keep the temperature when the air-conditioning should be switched on [4]

(b) Any **four** from:
- initialisation, set highest apartment temperature to a low value, set lowest apartment temperature to a high value outside loop
- input temperature
- store in array
- test for temperature > highest apartment temperature reset highest apartment temperature if this is the case
- test for temperature < lowest apartment temperature reset lowest apartment temperature if this is the case
- calculate range
- output highest temperature, lowest temperature and the range outside loop

(Max **four** marks)

- loop 60 times must have both tests within the loop, initialisation before the loop and output after the loop (**One** mark) [5]

sample algorithm:

```
HighestTemp ← 0; LowestTemp ← 100
FOR Count ← 1 to 60
   INPUT Temperature
   ApartmentTemp[Count] ← Temperature
   IF ApartmentTemp[Count] > HighestTemp
      THEN HighestTemp ← ApartmentTemp[Count]
   ENDIF
   IF ApartmentTemp[Count] < LowestTemp
      THEN LowestTemp ← ApartmentTemp[Count]
   ENDIF
NEXT Count
Range ← HighestTemp – LowestTemp
PRINT 'Highest Temperature recorded ', HighestTemp
PRINT 'Lowest Temperature recorded ', LowestTemp
PRINT 'Range ', Range
```

© Cambridge International Examinations 2015

(c) (i) Explanation **six** marks from:

1 – check if highest temperature <= 24 and lowest temperature >= 22…
– … message temperature always within acceptable range then exit

2 – check if highest out of range
– so count number of times temperature goes above range
– message recorded temperature too high on counted number of occasions

3 – check if lowest out of range
– so count number of times temperature goes below range
– message recorded temperature too low on counted number of occasions

General
– check all recorded temperatures (loop)

[6]

(ii) Any **one** from:
– only checks necessary conditions
– uses results from task 2 [1]

© Cambridge International Examinations 2015

Section B

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]

3 **(a)** **Number 1 Trace Table**

X	T1	T2	Output
37	2	5	5
2			2

← (1 mark) →← (1 mark) →

Number 2 Trace Table

X	T1	T2	Output
191	11	15	F
11			B

← (1 mark) →← (1 mark) → [4]

(b) – convert a denary number to hexadecimal
– and output it in reverse order [2]

4 **(a)** **(i)** Normal

(ii) Acceptable data to test that the results are as expected. [2]

(b) **One** mark for the data set, **one** mark for the type and **one** mark for the matching reason
There are many possible correct answers this is an example only.

Set 1	–	Age 4, height 0.9
Type	–	Boundary/Extreme
Reason	–	Data to test the validation that is just within the limits of acceptability
Set 2	–	Age 10, height 1.4
Type	–	Abnormal
Reason	–	Data that should be rejected and produce an error message [6]

© Cambridge International Examinations 2015

5 **One** mark for every **two** correct types

Boat Name – text
Model – text
Engine Power – number
Number of Seats – number
Life Raft – "yes/no"/text/Boolean
Day Price – currency/number

0, 1 no marks
2, 3 one mark
4, 5 two marks
6 three marks [3]

(b) **One** mark for each correct **different** check

Boat Name	Presence Check/Type Check/Character Check
Model	Format check/Type check/Presence Check/Length check/ Use of Drop-down box to select
Number of Seats	Type check/Presence Check/Range Check/ Use of Drop-down box to select
Day Price	Type check/Presence Check/Range Check [4]

(c)

Field:	Boat Name	Model	Day Price	Number of Seats	Engine Power
Table:	BOAT	BOAT	BOAT	BOAT	BOAT
Sort:					
Show:	☑	☑	☑	☐	☐
Criteria:				= 4	> 100
or:					
	(1 mark)	(1mark)	(1 mark)	(1 mark)	(1 mark) [5]

© Cambridge International Examinations 2015