

CAMBRIDGE INTERNATIONAL EXAMINATIONS

GCE Advanced Subsidiary Level and GCE Advanced Level

MARK SCHEME for the October/November 2012 series

9691 COMPUTING

9691/21

Paper 2 (Written Paper), maximum raw mark 75

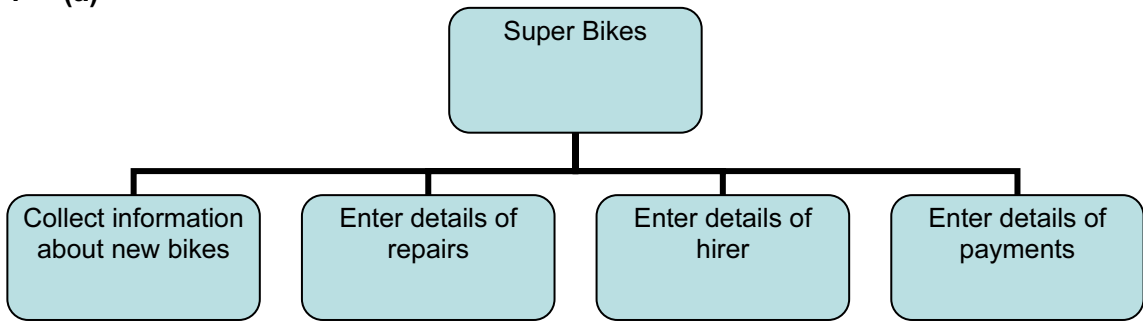
This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, 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 2012 series for most IGCSE, GCE Advanced Level and Advanced Subsidiary Level components and some Ordinary Level components.

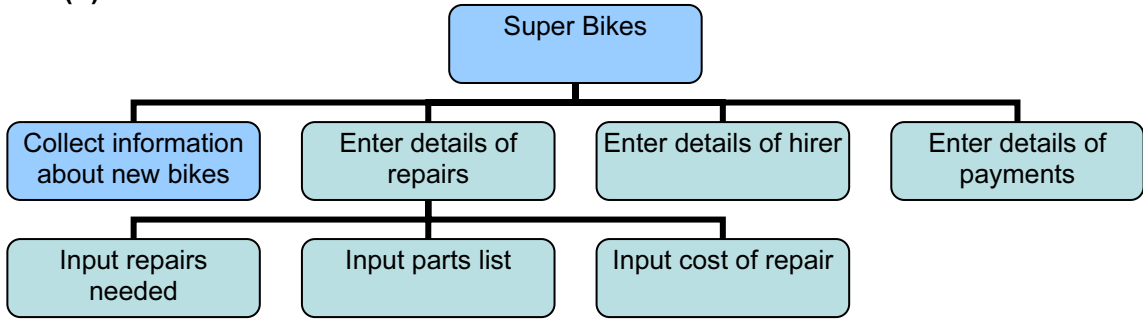
1 (a)



1 mark for left 2 blocks, 1 mark for right 2 blocks

[2]

(b)



1 mark for 3 blocks under Repairs

[1]

- (c) – to enable modular testing/maintenance/debugging
 – to enable different blocks to be worked on by different staff
 – easier to understand // reduce complexity

[2]

- (d) – the scope
 – of a variable is the range of statements for which it is valid
 – normally within a subprogram
 – enables the same identifier to be used for different purposes without conflict

[2]

- (e) – OR
 – OR

[2]

Page 3	Mark Scheme	Syllabus	Paper
	GCE AS/A LEVEL – October/November 2012	9691	21

(f) (i) e.g. Pascal

```

1  VAR BikeIDValid : BOOLEAN;
2  BikeIDValid := TRUE;
3  IF length(BikeID) <> 6
4    THEN BikeIDValid := FALSE;
5  IF NOT((Right(BikeID,2)>='00')
6    AND (Right(BikeID,2)<='99'))
7    THEN BikeIDValid := FALSE;
8  IF LEFT(BikeID,4) <> 'BIKE'
9    THEN BikeIDValid := FALSE;
10 IF BikeIDValid
11   THEN WriteLn('valid')
12   ELSE WriteLn('invalid');
```

e.g. VB 2005

```

1  BOOLEAN BikeIDValid
2  BikeIDValid = TRUE
3  IF LEN(CarReg) <> 6 THEN
4    BikeIDValid = FALSE
5  END IF
6  IF NOT(MID(BikeID,5,2)>="00"
7    AND MID(BikeID,5,2)<="99") THEN
8    BikeIDValid = FALSE
9  END IF
10 IF MID(BikeID,1,4) <> "BIKE" THEN
11   BikeIDValid = FALSE
12 END IF
13 IF BikeIDValid THEN
14   Console.WriteLine("valid")
15 ELSE
16   Console.WriteLine("invalid")
17 END IF
```

e.g. C#

```

1  bool bikeIDValid = true;
2  if (bikeID.Length != 6)
3  {
4    bikeIDValid := false;
5  }
6  if (!(bikeID.Substring(5,2)>="00"
7    && (bikeID.Substring(5,2)<="99"))
8  {
9    bikeIDValid := false;
10 }
11 if (bikeID.Substring(1,4) != "BIKE")
12 {
13   bikeIDValid := false;
14 }
15 if (bikeIDValid)
16 {
17   Console.WriteLine("valid");
18 }
19 else
20 {
21   Console.WriteLine("invalid");
22 }
```

Page 4	Mark Scheme	Syllabus	Paper
	GCE AS/A LEVEL – October/November 2012	9691	21

e.g. Python

```

1     bikeID = input()
2     bikeIDValid = True
3     if len(bikeID) != 6:
4         bikeIDValid = False
5     if ((bikeID[4:6] >='00') & (bikeID[4:6] <= '99')) != True:
6         bikeIDValid = False
7     if bikeID[0:4]!='BIKE':
8         bikeIDValid = False
9     if bikeIDValid:
10        print ('valid')
11    else:
12        print ('invalid')
```

1 mark for length check (6 characters exactly)

1 mark for correct separating 1st four characters

1 mark for testing first four characters are BIKE

1 mark for separating last two characters

1 mark for testing last two characters are digits

1 mark for initialising Boolean value

1 mark for changing Boolean value if error

1 mark for suitable message

1 mark for meaningful variable names used

1 mark for correct use of specified programming language

1 mark for indentation

[10]

- (ii) – 2nd to 4th characters are lower case letters // first 4 characters are Bike not BIKE
– in above example at line number 8 (Pascal), 10 (VB), 11 (C#)

[2]

(g) (i) white box

[1]

(ii) Alpha testing

Who – issue of software to a restricted number of testers within the company

When – it may not be completely finished and could have faults // before beta testing

Purpose – to find faults // to check the logic // to see if it works

[3]

2 (a)

Row	Position	Row<=30	Position <=3	BikePlace				
				[1,1]	[1,2]	[1,3]	[2,1]	[2,2]
1	1	TRUE	TRUE	BIKE34				
	2		TRUE		BIKE56			
	3		TRUE			BIKE70		
	4		FALSE					
2	1		TRUE				BIKE51	
	2		TRUE					BIKE19

[6]

(b) (i) e.g. Pascal

```

FOR Row := 1 TO 30 DO
  BEGIN
    FOR Position := 1 TO 3 DO
      BEGIN
        READLN(BikeID)
        BikePlace[Row,Position] := BikeID;
      END;
    END;
  END;

```

e.g. VB 2005

```

FOR Row = 1 TO 30
  FOR Position = 1 TO 3
    BikeID = CONSOLE.READLINE()
    BikePlace(Row,Position) = BikeID
  NEXT
NEXT

```

e.g. C#

```

for (int row = 1; row<= 30; row++)
{
  for (int position=1; position<=3; position++)
  {
    bikeID = Console.ReadLine();
    bikePlace[row,position] = bikeID;
  }
}

```

Page 6	Mark Scheme	Syllabus	Paper
	GCE AS/A LEVEL – October/November 2012	9691	21

e.g. Python

```
for row in range (1,31):
    for position in range (1,4):
        bikeID = input()
        bikePlace[row,position] = bikeID
```

1 mark for correct FOR loops

1 mark for correctly nested loops

1 mark for input in correct place

1 mark for correct lower and upper boundaries for outer loop

1 mark for correct lower and upper boundaries for inner loop

1 mark for assignment to correct array element

**1 mark for indentation*

Check that FOR and assignment statements are properly formed depending on the programming language

** = language independent marks*

[7]

- (ii) – any word in the vocabulary of a programming language
– which can only have the meaning defined in that language

[2]

- (iii) *Any two examples from (i) above (1 mark each)*
e.g. FOR, TO, NEXT, DO, BEGIN, END, int
follow through

[2]

- (c) (i) 0 (zero)

[1]

- (ii) Run-time error

[1]

- (iii) – check the value of the bracket before the division takes place // write error trapping code
– if bracket = 0 arrange for a message to be output // exception code
Accept answers in code

[2]

- (d) – lists the contents of variables
– at specific points in the program // at breakpoints
– allowing their contents to be compared with expected values

[2]

- 3 – date
– suitable report title
– company name (Super Bikes)
– income and repairs grouped by BikeID
– tabulated or other suitable layout
– headings/labels (must contain income, bike, number of times hired, repairs)
– well spaced out (making use of whole frame)
(if clearly a screen design do not give this mark)

[7]

Page 7	Mark Scheme	Syllabus	Paper
	GCE AS/A LEVEL – October/November 2012	9691	21

4 (a)

Field Name	Data Type	Size of Field (bytes)
BikeID	String/alphanumeric/text	6
BikeType	String/alphanumeric/text	10-20
DateBought	Date/integer/real/string	8 (accept 10, 12)
NeedsRepair	Boolean	1

Give a tick for each correct cell. Marks are half the number of ticks (round up) [4]

(b) $(6 + 20 + 8 + 1)$

* 90 / 1024

* 1.1 (or equivalent)

=approx 3.4 KB

1 mark per row above [4]

(c) e.g. Pascal

```
TYPE HireBike = RECORD
    BikeID: String[6];
    BikeType: String[10];
    DateBought: TDateTime;
    NeedsRepair: Boolean;
END;
```

e.g. VB 2005

```
STRUCTURE HireBike
    DIM BikeID AS String
    DIM BikeType AS String
    DIM DateBought AS Date
    DIM NeedsRepair AS Boolean
END STRUCTURE
```

e.g. C#

```
struct hireBike
{
    public string bikeID, bikeType;
    public dateTime dateBought;
    public bool needsRepair;
}
```

1 mark for correct record structure

1 mark for each field [5]

Page 8	Mark Scheme	Syllabus	Paper
	GCE AS/A LEVEL – October/November 2012	9691	21

- (d) (i) – a function returns a value
– there is no value to be returned from this subroutine [2]
- (ii) – Parameter passed by value:
– A local copy of the data is used
– Parameter passed by reference:
– the memory location of the data is used [4]
- (iii) – filename
– BikeRecord [1]