CANDIDATE NAME	

CENTRE NUMBER						CANDIDATE NUMBER			

COMPUTING   **9691/21**

Paper 2   **October/November 2012**

                                                                 2 hours

Candidates answer on the Question Paper.

No additional materials are required.

READ THESE INSTRUCTIONS FIRST

Write your Centre number, candidate number and name on all the work you hand in.
Write in dark blue or black pen.
You may use a soft pencil for any diagrams, graphs or rough working.
Do not use staples, paper clips, highlighters, glue or correction fluid.
DO **NOT** WRITE IN ANY BARCODES.

Answer **all** questions.

At the end of the examination, fasten all your work securely together.

The number of marks is given in brackets [ ] at the end of each question or part question.

This document consists of **14** printed pages and **2** blank pages.

UNIVERSITY *of* CAMBRIDGE
International Examinations

[Turn over

www.theallpapers.com

1 Soni works for a software house which has been asked to design software for a cycle hire company, Super Bikes.
Soni decides on the main tasks:

- collecting the information about new bikes
- entering details of repairs
- entering details of hirer
- entering details of payment

(a) Complete the structure diagram showing these tasks.

```
        ┌──────────────┐
        │  Super Bikes │
        └──────┬───────┘
    ───────────┼───────────
```
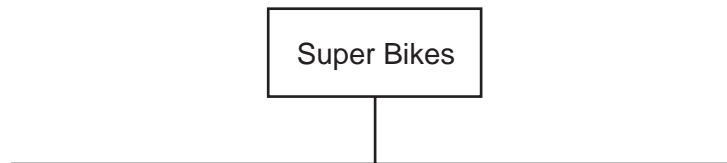
[2]

(b) The collection of information about repairs has three subtasks:

- input the repair needed
- input the parts list
- input the cost of the repair

Add these to the structure diagram in part **(a)**. [1]

(c) State **two** reasons for dividing the main task into smaller tasks.

1 ....

....

2 ....

.... [2]

(d) Each subtask is coded as a program module.

Explain how it is possible to use the same variable name for different variables in different modules.

....

....

....

.... [2]

(e) When validating the type of bike (road, hybrid, track) Soni uses

```
INPUT Bike

   IF (Bike = "road") .......... (Bike = "hybrid") ......... (Bike = "track")

      THEN

         OUTPUT Bike " is valid"

      ELSE

         OUTPUT "Invalid"

   ENDIF
```

Insert the correct Boolean operator in each space. [2]

(f) All bikes in stock have a bike ID. Each bike ID is six characters long and has the format BIKEXX, where X is a digit.
For example, BIKE65 is a valid bike ID.

(i) In a high-level programming language write code that will validate the format of a bike ID input into the variable BikeID and output a suitable message. (Add extra line numbers if required.)

Programming language used ....

Code

1 ....

2 ....

3 ....

4 ....

5 ....

6 ....

7 ....

8 ....

....

....

....

....

....

....

....

....

....

....

.... [10]

....

(ii) Explain why `Bike46` is an invalid bike ID.

State the line number of your code in part **(f)(i)** which detects that it is invalid.

Line Number [2]

(g) (i) Soni will need to test the logic of this program module.

State whether this is black box testing or white box testing.

[1]

(ii) Soni is also planning to use alpha testing of the software.

Explain who will do the testing, when the testing would be carried out, and what will be the purpose of this testing stage.
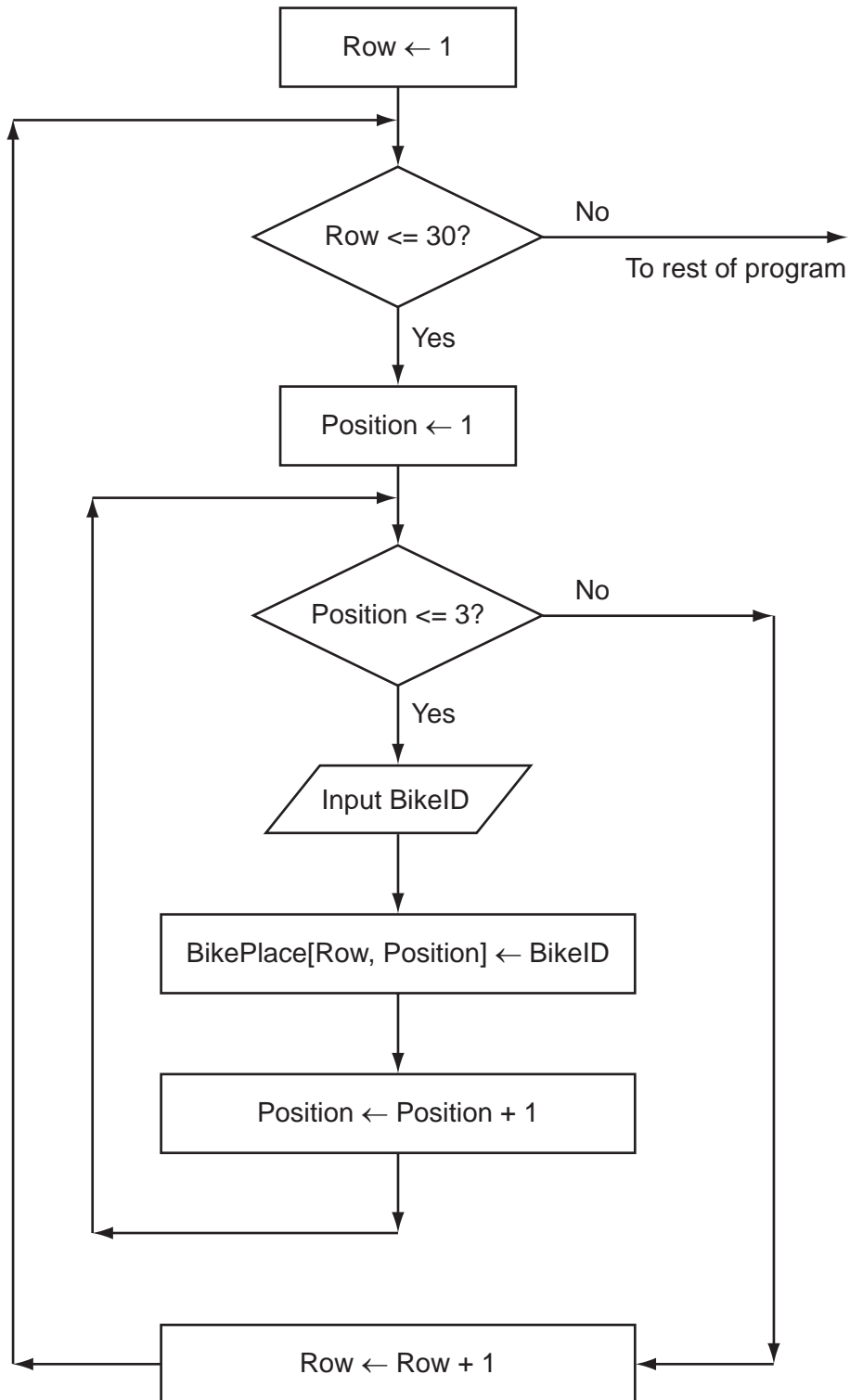
who

when

purpose

[3]

2 The company stores the bikes in bike racks. The bike racks are in 30 rows, each with 3 places. Each bike is always kept in the same place.

The array `BikePlace[30,3]` stores the bike ID.
Soni uses a flowchart to help him design a module to populate the array with bike IDs.

```
                         ┌─────────────┐
                         │  Row ← 1    │
                         └─────────────┘
                                │
                                ▼
                           ╱─────────╲                No
                          ╱  Row <= 30? ╲ ─────────────────────▶
                          ╲             ╱        To rest of program
                           ╲─────────╱
                                │ Yes
                                ▼
                         ┌─────────────┐
                         │ Position ← 1│
                         └─────────────┘
                                │
                                ▼
                           ╱─────────╲
                          ╱ Position <= 3? ╲  No
                          ╲               ╱ ──────────────────▶
                           ╲─────────╱
                                │ Yes
                                ▼
                          ╱───────────╲
                         ╱ Input BikeID ╱
                         ╱─────────────╱
                                │
                                ▼
              ┌──────────────────────────────────────┐
              │ BikePlace[Row, Position] ← BikeID     │
              └──────────────────────────────────────┘
                                │
                                ▼
              ┌──────────────────────────────────────┐
              │      Position ← Position + 1          │
              └──────────────────────────────────────┘
                                │
                                ▼

              ┌──────────────────────────────────────┐
              │           Row ← Row + 1               │
              └──────────────────────────────────────┘
```

www.theallpapers.com

(a) Complete the trace table for the flowchart with the first five bike IDs. The values to be entered are BIKE34, BIKE56, BIKE70, BIKE51 and BIKE19.
The row with the headings will also have to be completed.

Row	Position	Row<=30		BikePlace				
				[1,1]				
1	1	TRUE		BIKE34				

[6]

[Turn over
www.theallpapers.com

(b) (i) From the flowchart write the module code, using nested FOR loops.
Ensure that your code demonstrates good programming practice.

```
                    Row ← 1

              Row <= 30?     No
                             To rest of program
                  Yes

                  Position ← 1

              Position <= 3?     No
                  Yes

              Input BikeID

        BikePlace[Row, Position] ← BikeID

        Position ← Position + 1

              Row ← Row + 1
```

Programming language used ....

Code ....

....

....

....

....

....

....

....

....

....   [7]

(ii) Describe what is meant by a reserved word.

....

....

....   [2]

(iii) Write down **two** reserved words from your code in **(i)**.

Programming language ....

Words ....

....   [2]
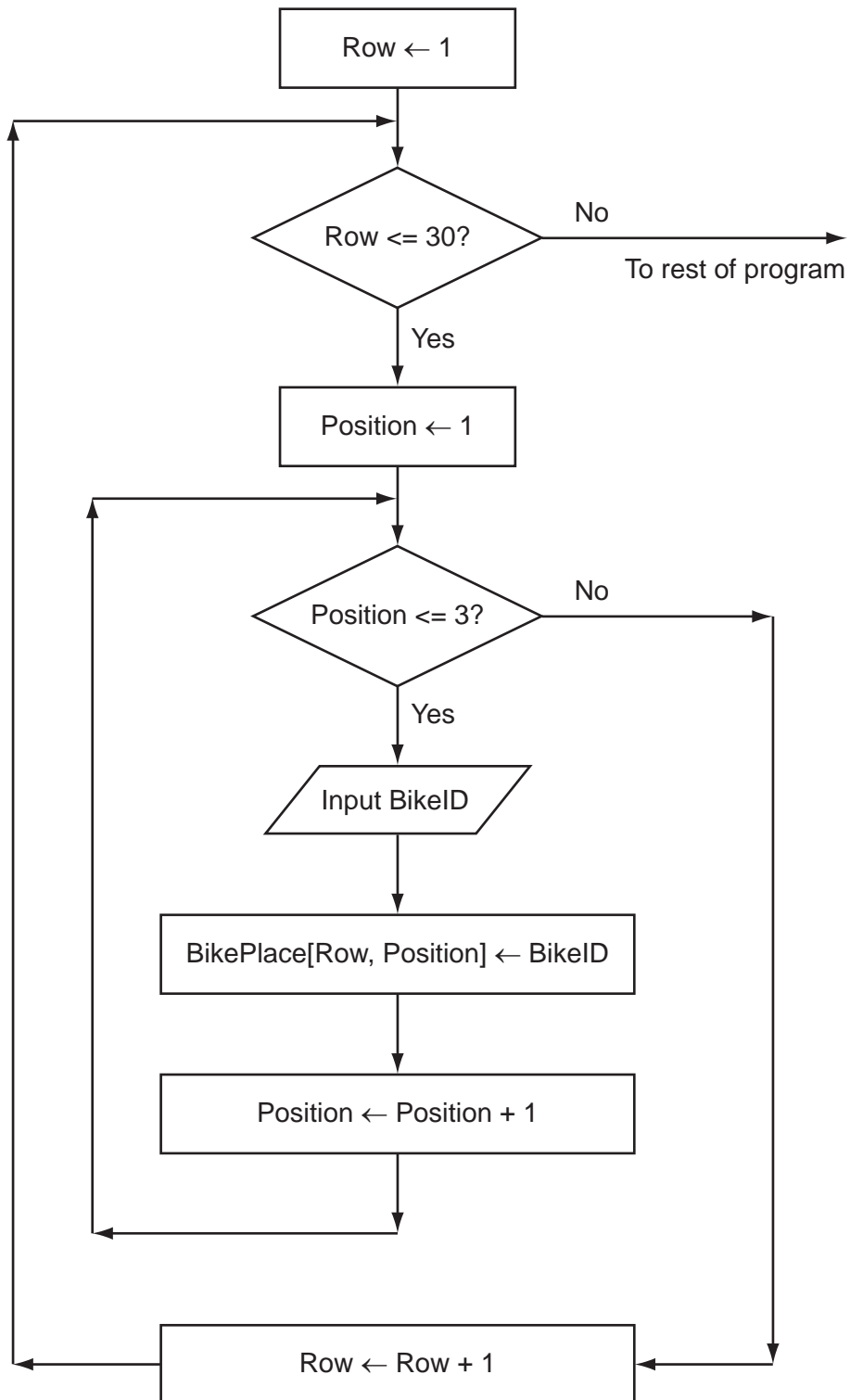
(c) (i) One of the modules uses the expression:

```
BuildingArea - RackArea /(HandleBarWidth * BikeLength –
                                        EmptySpaces)
```

State the value of the part of the expression inside the brackets when:

```
HandleBarWidth ← 2
BikeLength ← 4
EmptySpaces ← 8
```

[1]

(ii) If this expression is used with the above values, there will be an error.

State the name of this type of error.

[1]

(iii) Explain how the programmer can stop this error occurring during execution.

[2]

(d) There are a number of debugging tools associated with high-level languages.

Describe how a variable check is used.

[2]

3 Super Bikes will need a report each month that shows how many times the 5 most used bikes have been hired that month, the income each generated and any repairs needed.

Design the layout of the report.

[7]

4 Super Bikes will store the data in files.

(a) One file will store the following data:

- bike ID
- bike type
- date bought
- currently needs repair

Complete the following table. Use a single value for Field Size.

Field Name	Data Type	Field Size (in bytes)
BikeID		
BikeType		
DateBought		
NeedsRepair		

[4]

(b) Estimate the size, in kilobytes, of the file if it stores the details of 90 bikes. Show your working.

....

....

....

....

....

....

....

.... [4]

(c) Using a high-level programming language, define a bike record with identifier `HireBike` and the fields listed in part **(a)**.

Programming language used ....

Code ....

....

....

....

....

....

....

.... [5]

(d) Adding a `HireBike` record to the file could be programmed as procedure `AddBikeRecord`.

(i) Explain why adding a `HireBike` record to the file would not be programmed as a function.

....

....

....

.... [2]

(ii) Describe **two** different ways in which values can be passed between the main program and a procedure.

1 ....

....

....

....

2 ....

....

....

.... [4]

(iii) State **one** parameter that would be passed to the procedure `AddBikeRecord`.

.... [1]

BLANK PAGE

BLANK PAGE

www.theallpapers.com