UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS
General Certificate of Education
Advanced Subsidiary Level and Advanced Level

CANDIDATE NAME	

CENTRE NUMBER						CANDIDATE NUMBER				

COMPUTING **9691/23**

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.

UNIVERSITY of CAMBRIDGE
International Examinations

[Turn over

Soni works for a software house which has been asked to design software for Super B[...] company that specialises in hiring out motorbikes.

1   Soni decides on the main tasks:

  • enter bike details
    ○   bike specification
    ○   service history
    ○   hire rates
  • enter bike hire details

(a) Complete the structure diagram showing these tasks.

```
                    ┌──────────────┐
                    │  Super Bikes │
                    └──────┬───────┘
                           │
        ┌──────────────────┴──────────────────
        │
  ┌─────────────┐
  │ Enter bike  │
  │  details    │
  └─────────────┘
```
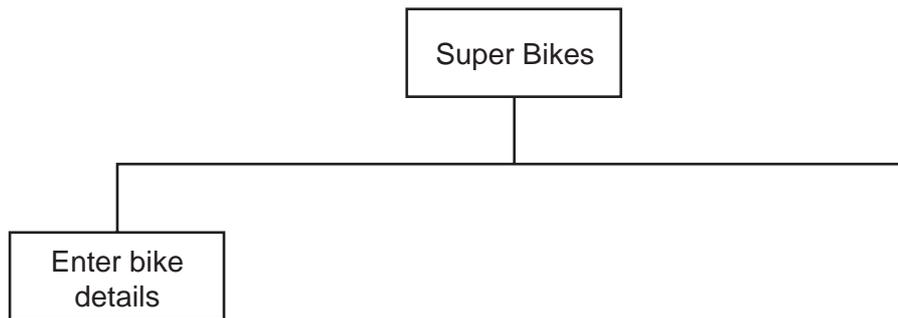
[2]

(b) Entering hire rate details has two subtasks:

  • input daily rates
  • input weekly rates

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

(c) The type of each bike is also recorded. The intention of this pseudocode is to only trial, scooter and road as valid bike types.

```
INPUT BikeType
IF ((BikeType = "trial") AND (BikeType = "scooter"))
                                        OR (BikeType = "road")
    THEN
        OUTPUT BikeType " is valid"
    ELSE
        OUTPUT "Invalid"
ENDIF
```

State the output when the BikeType input is:

(i) scrambler ....

(ii) trial ....

(iii) road .... [3]

(d) Rewrite the line in the pseudocode in part **(c)** which needs changing in order to correctly validate bike types.

....

.... [1]

(e) (i) All bike registrations are five characters long, and have the format BKXXX, X is a digit.
For example, `BK572` is a valid registration.

In a high-level programming language write code that will validate the format of a registration input into the variable `BikeReg`. (Add extra line numbers if required.)

Programming language used ....

Code

1 ....

2 ....

3 ....

4 ....

5 ....

6 ....

7 ....

8 ....

....

....

....

....

....

....

....

....

....

....

....

.... [10]
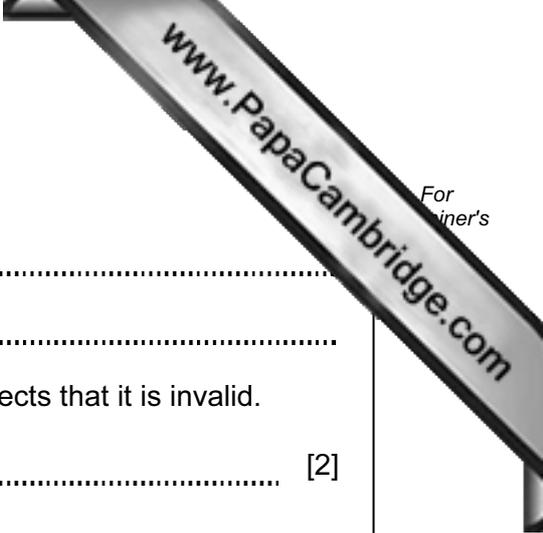
(ii) Explain why `487BK` is an invalid bike registration.

....

....

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

Line Number .... [2]

(f) The code in part **(e)(i)** will be subject to black box testing and white box testing.

Explain how this testing will be carried out.

(i) Black box testing

....

....

....

....

....

.... [3]

(ii) White box testing

....

....

....

....

....

.... [3]

2 Super Bikes owns a rectangular parking area with 30 rows; each row has 4 bike spaces.
Each bike is always parked in the same space.
The array `BikeSpace[30,4]` stores the bike registrations.
Soni uses a flowchart to help him design a module to populate the array with the bike registrations.
Input is terminated using the rogue value "`BK000`".

```
            ┌──────────────┐
            │   Row ← 1    │
            └──────┬───────┘
                   │
        ┌──────────┤
        │          ▼
        │      ◇ Row <= 30? ◇ ──No──→ To rest of program
        │          │
        │         Yes
        │          ▼
        │   ┌──────────────┐
        │   │ Position ← 1 │
        │   └──────┬───────┘
        │          │
        │   ┌──────┤
        │   │      ▼
        │   │  ◇ Position <= 4? ◇ ──No──→
        │   │      │                       │
        │   │     Yes                      │
        │   │      ▼                       │
        │   │  / Input BikeReg /           │
        │   │      │                       │
        │   │      ▼                       │
        │   │  ◇ BikeReg <> "BK000"? ◇ ──No──→ To rest of program
        │   │      │                       │
        │   │     Yes                      │
        │   │      ▼                       │
        │   │ ┌────────────────────────┐   │
        │   │ │BikeSpace[Row, Position]│   │
        │   │ │      ← BikeReg         │   │
        │   │ └──────────┬─────────────┘   │
        │   │            ▼                 │
        │   │ ┌────────────────────────┐   │
        │   │ │ Position ← Position + 1│   │
        │   │ └──────────┬─────────────┘   │
        │   └────────────┘                 │
        │                                  │
        │   ┌────────────────────────┐     │
        └───│     Row ← Row + 1      │◄────┘
            └────────────────────────┘
```
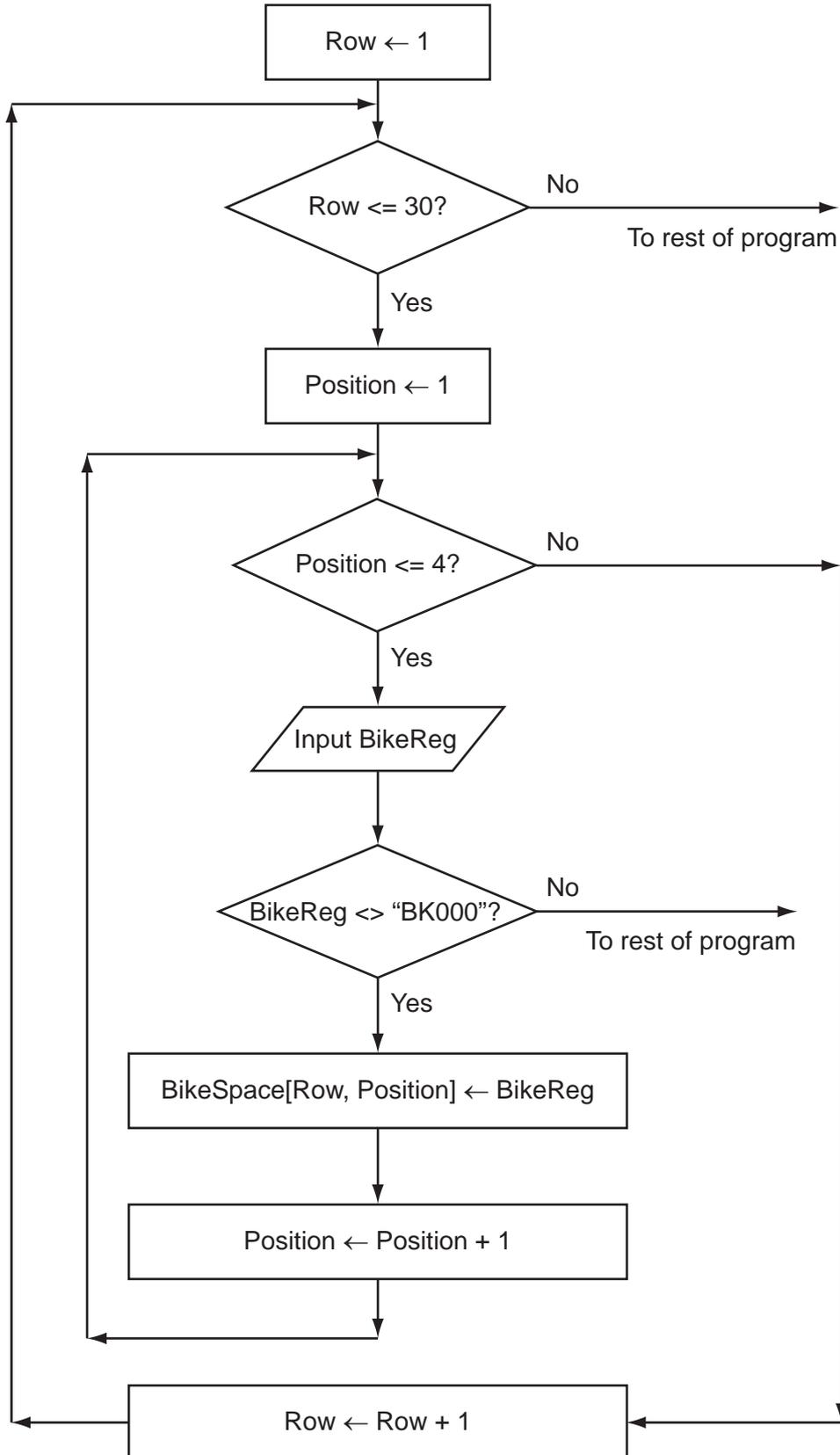
(a) Complete the trace table for the flowchart with the first five bike registration... values to be entered are BK707, BK380, BK162, BK747 and BK913.
The row with the headings will also have to be completed.

Row	Position	Row<=30			BikeSpace				
					[1,1]				
1	1	TRUE			BK707				

[6]

(b) From the flowchart write the module code, using nested `REPEAT…UNTIL` loops. Ensure that your code demonstrates good programming practice.

```
                        ┌─────────────────┐
                        │    Row ← 1      │
                        └─────────────────┘
                                │
                                ▼
                           ╱ Row <= 30? ╲ ──── No ────►
                           ╲            ╱         To rest of program
                                │
                               Yes
                                │
                                ▼
                        ┌─────────────────┐
                        │  Position ← 1   │
                        └─────────────────┘
                                │
                                ▼
                         ╱ Position <= 4? ╲ ──── No ────►
                         ╲                ╱
                                │
                               Yes
                                │
                                ▼
                        ╱ Input BikeReg ╱
                                │
                                ▼
                      ╱ BikeReg <> "BK000"? ╲ ──── No ────►
                      ╲                     ╱      To rest of program
                                │
                               Yes
                                │
                                ▼
                  ┌───────────────────────────────────┐
                  │ BikeSpace[Row, Position] ← BikeReg │
                  └───────────────────────────────────┘
                                │
                                ▼
                  ┌───────────────────────────────────┐
                  │    Position ← Position + 1         │
                  └───────────────────────────────────┘
                                │
                                ▼
                  ┌───────────────────────────────────┐
                  │         Row ← Row + 1              │
                  └───────────────────────────────────┘
```

Programming language used ....

Code

....

....

....

....

....

....

....

....

....

....

.... [7]
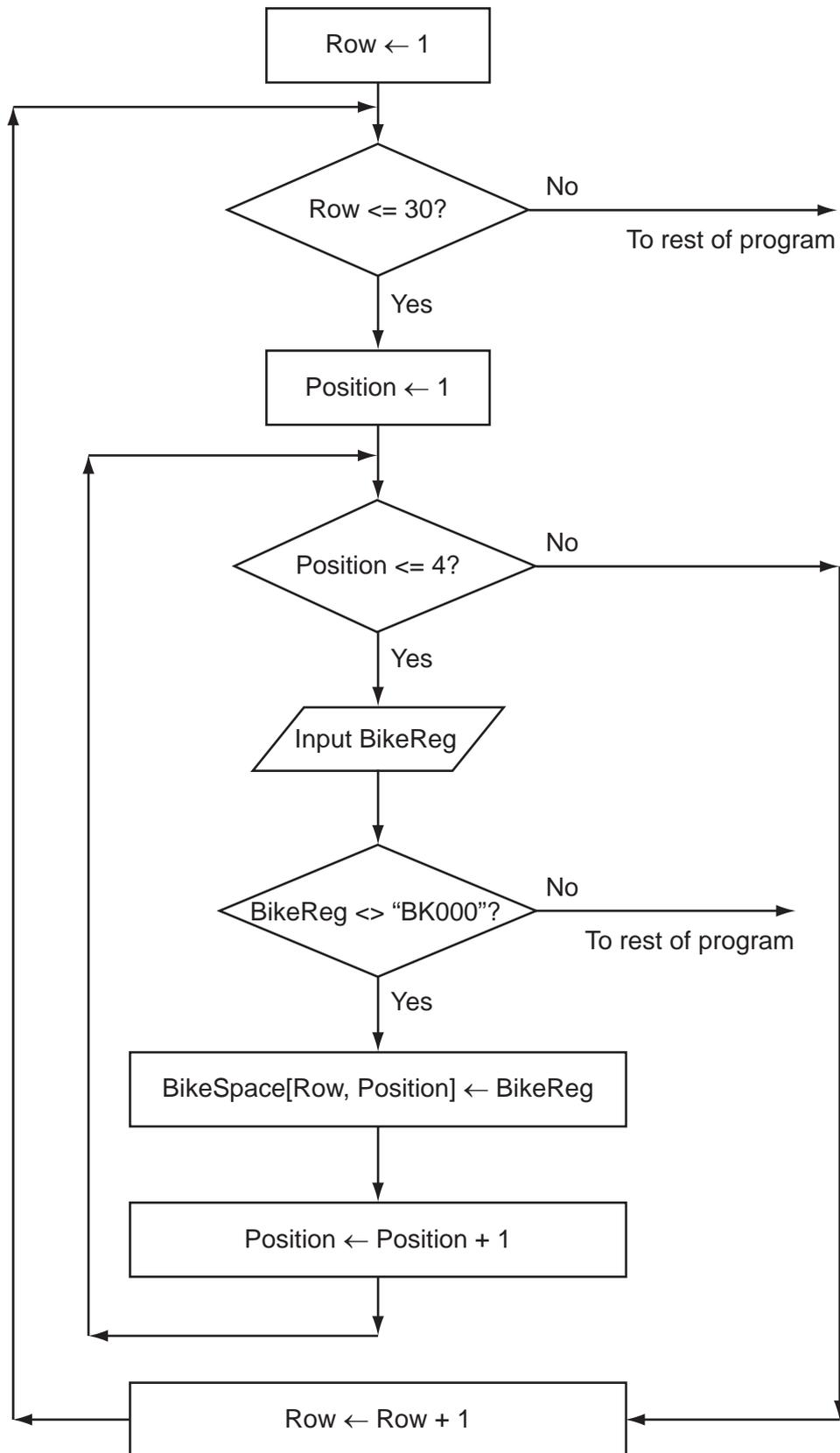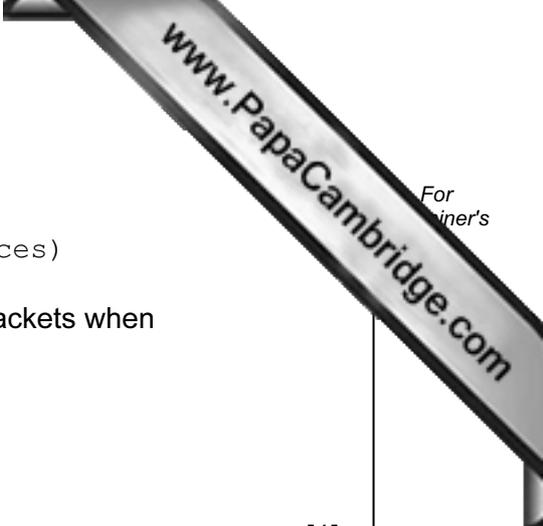
(c) (i) One of the modules uses the expression:

```
Area /(SpaceWidth * SpaceLength – EmptySpaces)
```

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

```
SpaceWidth ← 7
SpaceLength ← 4
EmptySpaces ← 28
```

.... [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 type of error occurring during execution.

....

....

....

.... [2]

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

Describe how breakpoints and a variable watch can be used when developing a program.

....

....

....

....

....

....

....

.... [4]

3 Super Bikes will need a printed report that shows all the hires made in a given [obscured] grouped by insurance rating (A, B or C) and the income each hire generated. The re[obscured] should include the total income figure for each insurance group.

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 registration
       •   purchase cost of bike
       •   insurance rating (A, B or C only)
       •   whether or not service is due

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

Field Name	Data Type	Field Size (in bytes)
BikeReg		
PurchaseCost		
InsuranceRating		
ServiceDue		

[4]

(b) Estimate the size, in kilobytes, of the file if it stores the details of 1000 bikes. Show working.

....

....

....

....

....

....

....

.... [4]

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

Programming language ....

Code

....

....

....

....

....

....

....

.... [5]

(ii) Bike records are stored in a file `SuperBikes`.

Write the code to read every bike record, count the number of bikes where service is due and output the result. You should assume that the file is not already in use.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [9]

BLANK PAGE

BLANK PAGE

BLANK PAGE