Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary Level and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

COMPUTING **9691/21**

Paper 2 **May/June 2014**

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, 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 **11** printed pages and **1** blank page.

1 A teacher wants to write a program to help young children learn their multiplication tables.

(a) (i) Draw a suitable layout for the initial screen to let a child choose which multiplication table between 1 and 10 they want to learn.

[3]

(ii) Explain how the child can choose a number using your screen design in **part (a)(i)**.

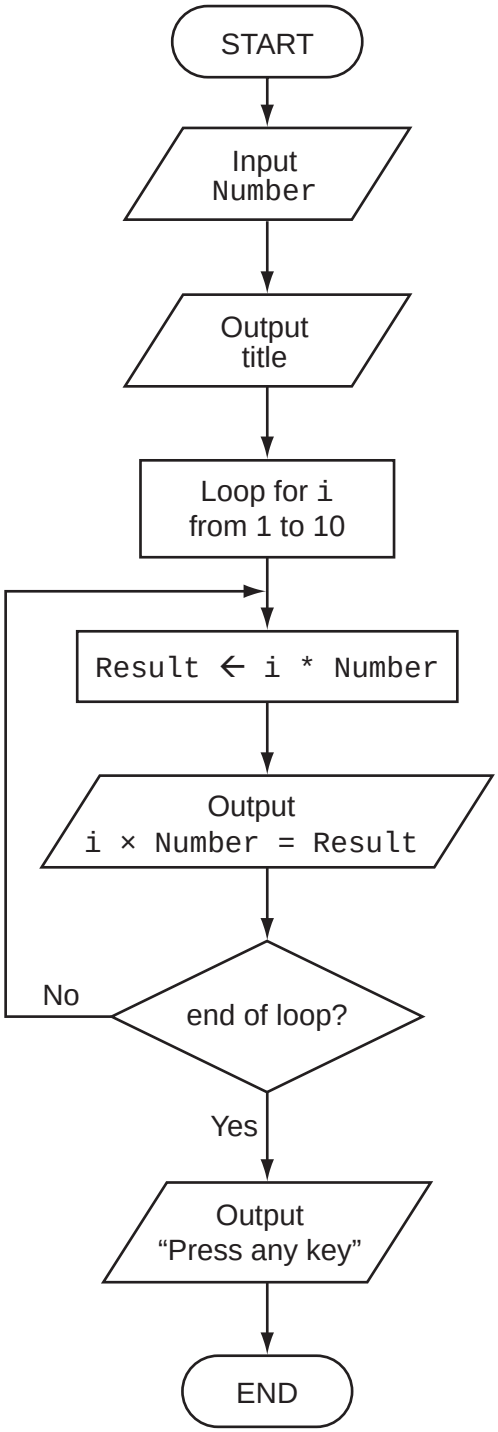
.....

..... [1]

(b) If the child chooses the number 7, the screen displays:

```
7 Times Table
=============
  1 × 7 =  7
  2 × 7 = 14
  3 × 7 = 21
  4 × 7 = 28
  5 × 7 = 35
  6 × 7 = 42
  7 × 7 = 49
  8 × 7 = 56
  9 × 7 = 63
 10 × 7 = 70

Press any key
```

The algorithm to produce this output is represented by the flowchart below:



The program needs the following three integer variables:

- `Number`
- `i`
- `Result`

In a high-level language, write programming statements to implement the algorithm.

Language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

(c) The teacher wants the program to:
- show a visual representation of a multiplication
- ask the child to key in an answer.

For example, the multiplication of 3 × 4 is represented as shown.

```
            3 × 4

      ****
      ****
      ****

     What is the answer? _______
```

This grid of asterisks (*) is produced by the procedure call:

```
ShowMultiplicationGrid(3, 4)
```

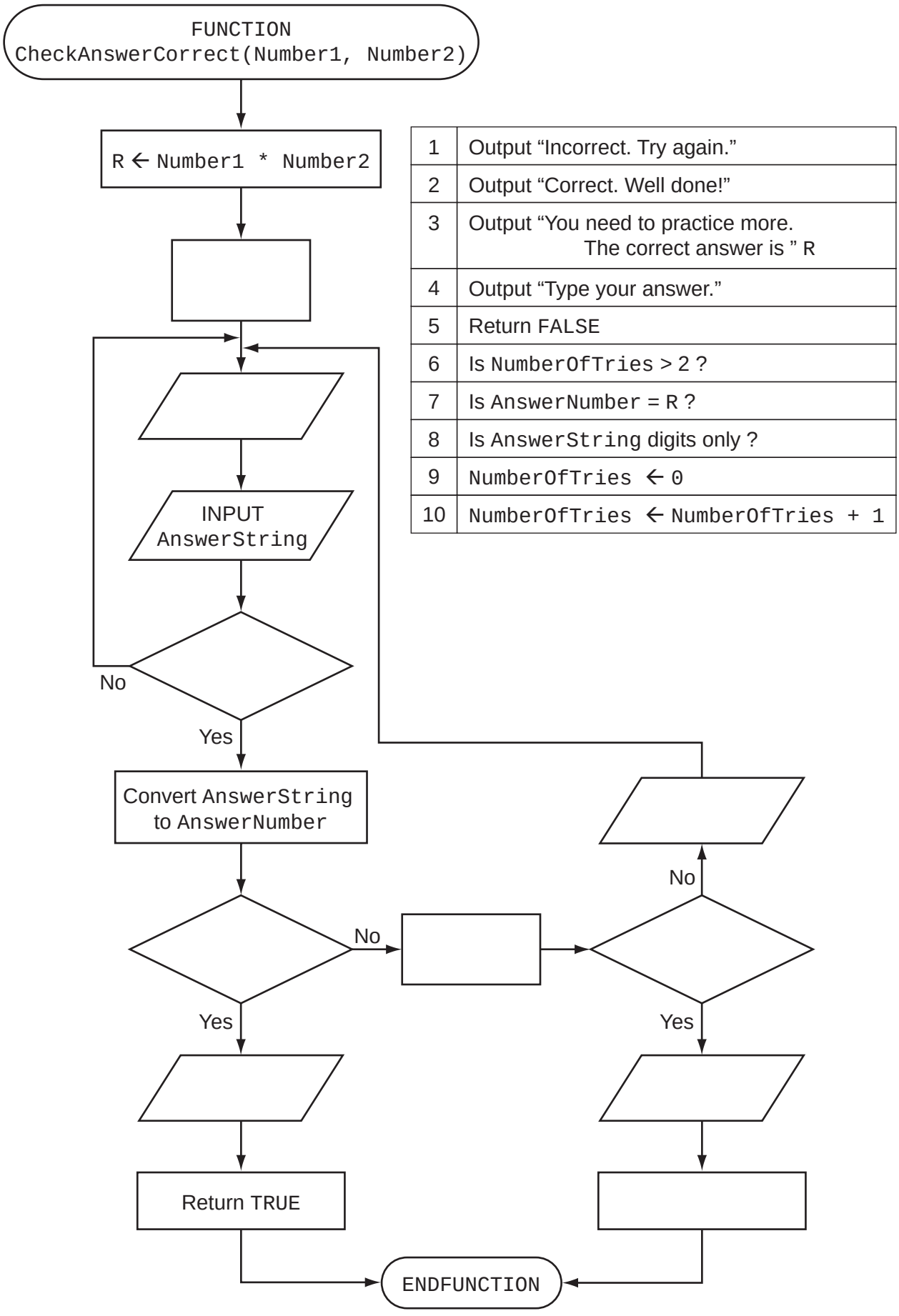
Complete the pseudocode for this procedure:

```
PROCEDURE ShowMultiplicationGrid(Number1, Number2)

  FOR Row ← .....

  FOR Column ← .....

    OUTPUT .....

    .....

    .....

  ENDFOR

ENDPROCEDURE
```

[5]

(d) The function `CheckAnswerCorrect` gives the child three chances to type in the correct answer. The function returns `TRUE` if the child typed the correct answer, and `FALSE` if all three attempts are incorrect.

Complete the flowchart opposite, using the given statements. Label each blank symbol with the correct statement number.



1	Output “Incorrect. Try again.”
2	Output “Correct. Well done!”
3	Output “You need to practice more. The correct answer is ” R
4	Output “Type your answer.”
5	Return FALSE
6	Is NumberOfTries > 2 ?
7	Is AnswerNumber = R ?
8	Is AnswerString digits only ?
9	NumberOfTries ← 0
10	NumberOfTries ← NumberOfTries + 1

[10]

2 The teacher wants a new program to test children. A sequence of multiplication questions is displayed. If a child inputs 3 incorrect answers for a question, the program outputs the number of questions the child answers correctly. Then the program stops.

The program design uses the following functions and procedures.

Function or procedure name	Description
`Random(X)`	A function that returns a whole number in the range 1 to `X` inclusive
`Display(Number1, Number2)`	A procedure that produces the output 5 × 8 = ? when `Number1` is 5 and `Number2` is 8
`CheckAnswerCorrect(Number1, Number2)`	The procedure from Question **1(d)**
`TestScoreTotal`	A function that returns the number of questions answered correctly

(a) Complete the pseudocode:

```
FUNCTION TestScoreTotal RETURNS .....
  DECLARE AnswerCorrect, ..... : BOOLEAN
  DECLARE ....., ....., ..... : INTEGER
  ..... // initialise Score
  Finish ← FALSE
  REPEAT
    Number1 ← ..... // generate two
    Number2 ← ..... //        numbers
    Display(Number1, Number2)
    AnswerCorrect ← CheckAnswerCorrect(Number1, Number2)
    IF ..... // was answer correct?
      THEN Score ← Score + 1
      ELSE ..... // flag set to stop program
    ENDIF
  UNTIL ..... // is flag set?
  RETURN ..... // return the number of
                // questions answered correctly
ENDFUNCTION
```

[9]

(b) During the school day, several children in the class will use this program. The teacher wants to store each child's name and their best test score so far.

Assume there will be no more than 30 children in the class.

(i) Two one-dimensional arrays, `Name` and `BestScore`, are to store the name and score for each child in the class.
Use a high-level programming language to declare these **two** arrays.

Language .....

Code .....

..... [2]

(ii) Alternatively, the data for one child could be stored in a record `StudentScore`.

Use a high-level programming language to declare `StudentScore`.

Language .....

Code .....

.....

.....

..... [2]

(iii) Use the same programming language as in **part (b)(ii)** to declare a one-dimensional array of records to store the data for the whole class. Use the identifier `Student`.

.....

..... [2]

(iv) Write the program statements to assign the following values to the third record in the array:

- Name: 'Anji'
- Score: 15

.....

..... [3]

(c) The teacher wants to store in a file the name and best score so far for each of the children.

A procedure, `SaveToFile`, is needed to save the data stored in the `Student` array of records. The data must be stored in the file `StudentFile`.

Use a high-level programming language to write this procedure.

Language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [6]

3 After a child uses the program again, the teacher wants the program to find the child's record in the array and update the score if it is higher.

(a) (i) A function, `FindArrayIndex`, is needed to return the array index of the record in which the child's name is stored.
Write pseudocode for this function.

Assume:
- the child's name exists in the array
- the children's names are unique
- the children's names are not in any order.

```
FUNCTION FindArrayIndex(ChildName : STRING)
.....
.....
.....
.....
.....
.....
.....
.....
.....
ENDFUNCTION
```
[5]

(ii) A requested name may not be present in the array.
Describe the changes necessary to the design of function `FindArrayIndex`.

.....

.....

.....

..... [2]

(b) A child might type in their name with a mixture of upper and lower case letters and the program must allow for this. For example, ALI and Ali are treated as the same name.

Describe how you can ensure that the program treats a name the same regardless of case.

.....

.....

.....

..... [2]

4 Look at this pseudocode function:

```
FUNCTION Y(s : STRING) RETURNS STRING
  DECLARE x : INTEGER
  x ← LENGTH(s)
  IF x = 1
    THEN
      RETURN s
    ELSE
      RETURN Y(RIGHT(s, x - 1)) + LEFT(s, 1)
      // RIGHT above returns rightmost x - 1 characters of string s
      // LEFT above returns leftmost character of string s
  ENDIF
ENDFUNCTION
```

(i) How can you tell that function `Y` is recursive?

.....

..... [1]

(ii) Dry-run the function when it is called with the string '`BYTE`' as parameter.

Call Number	Function call	s	x	RIGHT (s, x – 1)	LEFT (s, 1)	Return value
1	Y ('BYTE')	'BYTE'				

[7]

(iii) What does function `Y` do?

.....

..... [1]

(iv) List **two** features of the pseudocode above that make it easier to understand.

1 .....

2 ..... [2]

(v) List **one** feature of the pseudocode above that makes it more difficult to understand.

1 ..... [1]

(vi) Re-write the pseudocode function as an iterative function.

```
FUNCTION Y(s : STRING) RETURNS STRING
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

```
ENDFUNCTION
```

[5]

BLANK PAGE