Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

3799235190

COMPUTING **9691/23**

Paper 2 **May/June 2015**

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 an HB 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 **16** printed pages.

DC (CW/FD) 101099/1

1 **(a)** A high-level programming language has built-in string handling functions defined as follows:

```
MID(ThisString : STRING, x : INTEGER, n : INTEGER) RETURNS STRING
```

returns a string value consisting of `n` characters from the string `ThisString` starting from position `x`.

For example: `MID("STOP", 3, 2)` returns `"OP"`

If the function call is not properly formed an error is generated.

```
CONCAT(String1 : STRING, String2 : STRING) RETURNS STRING
```

returns a string value consisting of `String1` followed by `String2`.

For example: `CONCAT("HE", "LLO")` returns `"HELLO"`

If the function call is not properly formed an error is generated.

```
LENGTH(ThisString : STRING) RETURNS INTEGER
```

returns the number of characters in string `ThisString`.

For example: `LENGTH("Hello")` returns `5`

If the function call is not properly formed an error is generated.

```
String1 ← "RED"

String2 ← "F"
```

What will be returned from the following function calls?

(i) `MID(String1, 3, 1)`

....[1]

(ii) `MID(String1, 3, 2)`

....[1]

(iii) `CONCAT(String2, String1)`

....[1]

(b) Meena wants to write code to reverse a string stored in the variable `Original`. The result is to be stored in the variable `Reverse`.

(i) Use all the functions described in **part (a)** to write the pseudocode required to reverse the string.

....

....

....

....

....

....

....

....

....[5]

(ii) A palindrome is a string that reads the same backwards as forwards.

Complete the Boolean expression to check whether the two strings, `Original` and `Reverse`, are palindromes.

```
IF ......................................................................................................
  THEN
    OUTPUT "This is a palindrome"
ENDIF
```

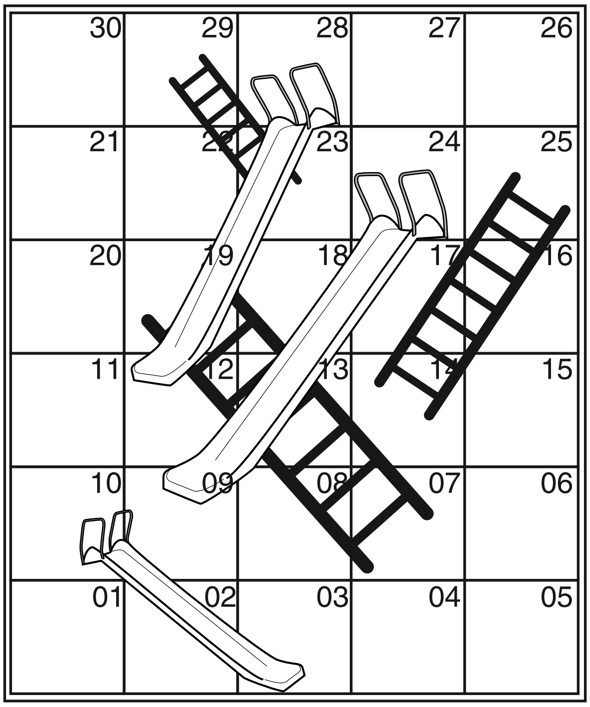
[1]

2 Ali has designed a board game.

The board consists of numbered squares.

Slides and ladders connect some of the squares.

An example layout of a board is:



The object of the game is for a player to move from the bottom square (Square 01) to the top square (Square 30).

A roll of a 6-sided die determines how many squares the player moves.

The table below shows example moves. These use the board layout above, with the player currently on Square 04.

Number rolled by die	Square moved to	Consequence	Final position
2	06	–	06
3	07	The base of a ladder, so climb to top of ladder	19
6	10	The top of a slide, so move down the slide	03

Ali plans to write a program to test different board designs. Each board design will have different numbers, positions and lengths for the slides and ladders.

The program is to test each board design to check that it is possible to complete the game in a reasonable number of moves.

For each board design, the program will simulate playing the game 1000 times and report the average number of moves.

(a) Ali starts the high level design of his program using pseudocode.

```
01 CALL InitialiseArray()  // blank board

02 CALL InputBoardDesign() // add slides and ladders data

03 TotalMoves ← ..............................................................

04 FOR Game ← ................................................................

05   // play next game and update TotalMoves

06   TotalMoves ← TotalMoves + NumberOfMovesInThisGame()

07 ...........................................................................

08 AverageMovesPerGame ← ....................................................

09 OUTPUT AverageMovesPerGame
```

(i) Complete the pseudocode above. [4]

(ii) Identify **one** feature in the pseudocode above which indicates that top-down design has been used.

....

....[1]

(iii) State **one** benefit of top-down design.

....

....[1]

(iv) Give the line number of a statement which shows:

Assignment ....

Iteration ....

A function call ....[3]

(v) List the variable identifiers used in the pseudocode.

....

....

....

....[2]

(b) Ali decides to store the board design in a one-dimensional array `Board`.

Each array element represents one square of the board.

If the square is the **start** of a slide or ladder, the array element will store the number of the square where the slide or ladder ends.

For example:

`Board`

[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]		[28]	[29]	[30]
						19			3		12		

(i) State what value Ali should use for the array elements that do not contain the start of a slide or ladder. Explain your choice.

....

....

....

....[2]

(ii) In a high-level programming language, write statements to declare the array `Board`.

Programming language ....

Declaration ....

....[2]

(iii) In the high-level programming language you chose in **part (b)(ii)**, write statements to initialise the array `Board` before there are any slides or ladders.

....

....

....

....[2]

(c) To input a board design, Ali wants the program to read in pairs of numbers in the form a, b where:

- a is the start square of a slide or ladder
- b is the end square of a slide or ladder
- the rogue value pair 0, 0 terminates the input

Write **pseudocode** for the procedure `InputBoardDesign` which:

- reads one or more number pairs
- updates the initialised array `Board`

```
PROCEDURE InputBoardDesign()
```

....

....

....

....

....

....

....

```
ENDPROCEDURE
```

[4]

(d) (i) A high-level programming language has a built-in function that generates a random number. The function is defined as follows:

```
RANDOM(n : INTEGER) RETURNS INTEGER
```

returns an integer value in the range `0` to `n` inclusive.

For example: `RANDOM(4)` returns `0`, `1`, `2`, `3` or `4`

If the function call is not properly formed an error is generated.

Ali's program is to use this random number generator to simulate the rolling of a die.

Complete the statement to assign a number between `1` and `6` to `NumberRolled`.

`NumberRolled` ← ....

....[1]

(ii) The function `NumberOfMovesInThisGame` counts how many moves the player took to complete the game and returns this value.

The game ends when the player reaches the final square (Square 30).

If a die roll produces a number that would go beyond the final square the die roll counts as a move but the player's position remains as before.

Complete the pseudocode on the next page.

```
FUNCTION NumberOfMovesInThisGame()
   DECLARE ..................................................................
   DECLARE ..................................................................
   DECLARE ..................................................................
   PlayerPosition ← .........................................................
   MovesSoFar ← .............................................................
   //
   REPEAT
      NumberRolled ← ........................................................
      MovesSoFar ← ..........................................................
      // check that move does not go beyond final square
      IF ....................................................................
         THEN  // make move
            PlayerPosition ← ................................................
            // check for slide or ladder and, if required, move
            .................................................................
            .................................................................
            .................................................................
            .................................................................
      ENDIF
   UNTIL ....................................................................
    //
   ..........................................................................
ENDFUNCTION
```

[8]

(e) Ali intends to save his board design to a text file.

In a high-level programming language write the procedure `SaveBoardDesign` to save to the file `Design.txt`:

- the contents of array `Board`
- the value of `AveragePerMovesGame` calculated for this board design

Programming language ....

Code ....

....

....

....

....

....

....

....

....

....[5]

(f) Ali wants to test board designs which have a different number of squares; for example 36 squares or 49 squares.

Ali wishes to minimise the changes required to his program code when the number of squares for the board changes.

Explain what programming language feature he should use.

....

....

....

....[2]

Question 3 begins on Page 12.

3 (a) Meena has written the algorithm below and wants to check that it works correctly.

```
FOR i ← 1 TO 4
  FOR j ← 1 TO 4
    IF Numbers[j] > Numbers[j + 1]
      THEN
        w ← Numbers[j]
        Numbers[j] ← Numbers[j + 1]
        Numbers[j + 1] ← w
    ENDIF
  ENDFOR
ENDFOR
```

(i) Dry run the algorithm by completing the trace table.

For each iteration, use a new row in the table.

i	j	Numbers[j] > Numbers[j + 1]	w	Numbers				
				[1]	[2]	[3]	[4]	[5]
–	–	–	–	49	98	36	70	51

[6]

(ii) Explain what Meena's algorithm does.

....

....

....

....[2]

(iii) State how many iterations of the outer loop have occurred before there are no more changes to the contents of the array.

....[1]

(iv) Describe why this algorithm is inefficient.

....

....

....

....[2]

(v) Meena has started to write an improved version of the algorithm.

Complete the pseudocode.

```
n ← 4
REPEAT
   ..........................................................................................
   FOR j ← 1 TO ..........................................................................
      IF Numbers[j] > Numbers[j + 1]
         THEN
            w ← Numbers[j]
            Numbers[j] ← Numbers[j + 1]
            Numbers[j + 1] ← w
            ...............................................................................
      ENDIF
   ENDFOR
   ..........................................................................................
UNTIL ..........................................................................................[5]
```

(b) Meena has written the algorithm with some features that make it easier to understand.

(i) State **one** such feature.

....

....[1]

(ii) State one **other** feature that Meena could introduce to this algorithm to make it easier to understand.

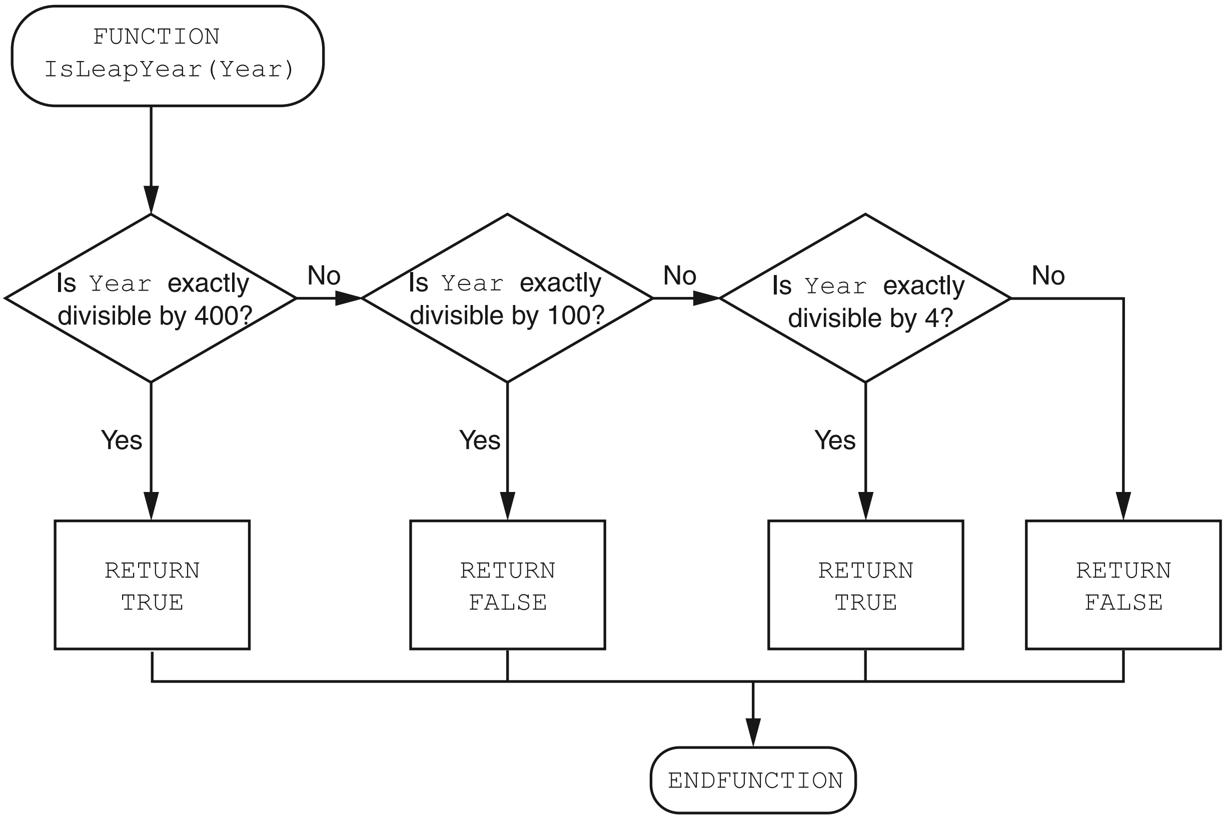
....

....[1]

4 A leap year is a year with special numerical properties.

Ahmed is planning to write a function to check whether a year is a leap year.

He starts by drawing a flowchart.


FUNCTION IsLeapYear(Year)
Is Year exactly divisible by 400?
No
Is Year exactly divisible by 100?
No
Is Year exactly divisible by 4?
No
Yes
Yes
Yes
RETURN TRUE
RETURN FALSE
RETURN TRUE
RETURN FALSE
ENDFUNCTION

(a) Using nested `IF` statements, convert the flowchart into **program code**.

Hint: The operator `MOD` gives the remainder of a division.

Programming language ....

Code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(b) Ahmed wants to carry out white box testing of the function.

Give **four** integers which thoroughly test the function. For each one, give the expected return value and justify your choice.

	`Year`	Expected return value	Justification
1			....
2			....
3			....
4			....

[4]

(c) When Ahmed has tested the function, he plans to use it in a program.

Give **two** types of testing that Ahmed could do with the completed program.

1 ....

....

2 ....

....[2]