Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME									
CENTRE NUMBER						CANDIDATE NUMBER			

9180624047

COMPUTING **9691/23**

Paper 2 **October/November 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 **12** printed pages.

CAMBRIDGE International Examinations

1 Rema surveys the students in her class to find out which is the most popular sport.

She draws a tally chart:

1	Cricket	\\\\\\\\
2	Football	\\
3	Tennis	\\\
4	Swimming	\\\\\\

Rema plans to collect sport data from students in the whole school. She designs a program to:

- input the number of the sport a student likes best (1, 2, 3 or 4)
- repeatedly ask for input until the input is 0 (zero)
- keep a count of each choice
- on completion of data entry, print out the results as a tally chart (as shown above)

Rema's first attempt is the following pseudocode:

```
Cricket ← 0
Football ← 0
Tennis ← 0
Swimming ← 0
REPEAT
   INPUT Choice
   CASE Choice OF
      1: Cricket ← Cricket + 1
      2: Football ← Football + 1
      3: Tennis ← Tennis + 1
      4: Swimming ← Swimming + 1
   ENDCASE
UNTIL Choice = 0
OUTPUT "Cricket   ", Cricket
OUTPUT "Football  ", Football
OUTPUT "Tennis    ", Tennis
OUTPUT "Swimming  ", Swimming
```

Her friend Aisha suggests that the pseudocode could be improved by:
- using a one-dimensional array, `Tally`, instead of four variables to store the counts
- modularising the design.

The main program should just consist of three procedure calls:

```
InitialiseArrayCounts
InputStudentChoices
OutputTallyChart
```

(a) Write a pseudocode statement to declare the array `Tally`.

.....

.....[2]

(b) (i) Write the pseudocode for the procedures `InitialiseArrayCounts` and `InputStudentChoices`:

```
PROCEDURE InitialiseArrayCounts
```

.....

.....

.....

.....

.....

```
ENDPROCEDURE

PROCEDURE InputStudentChoices
```

.....

.....

.....

.....

.....

```
ENDPROCEDURE
```

[4]

(ii) State the variable that the array element `Tally[2]` has replaced.

.....[1]

(c) Rema wants her output to look like this:

```
1   Cricket  \\\\\\\\
2  Football  \\
3    Tennis  \\\
4  Swimming  \\\\\\
```

She designs two procedures `OutputTallyChart` and `OutputTally`.

`OutputTallyChart` outputs the first two columns of the tally chart and then calls `OutputTally` to output the correct number of bars (\).

Complete the pseudocode:

```
PROCEDURE OutputTallyChart
   OUTPUT "1   Cricket  "

   OutputTally(.....................................)
   OUTPUT "2  Football  "

   OutputTally(.....................................)
   OUTPUT "3    Tennis  "

   OutputTally(.....................................)
   OUTPUT "4  Swimming  "

   OutputTally(.....................................)
ENDPROCEDURE


PROCEDURE OutputTally(SportCount : INTEGER)

   IF ..................................... // check there are bars
                                           // to output
      THEN

         ......................................
           OUTPUT '\'

         ......................................
   ENDIF

    ....................................
ENDPROCEDURE
```

[6]

(d) Rema wants to test each module before she tests the whole program.
The first module she is going to test is the `OutputTally` procedure using different parameter values.

Give **three** different types of test data with an example of each.

Justify your choices.

Type of test data	Example test data	Justification

[9]

(e) **(i)** All programs should be maintainable. Rema has followed good practice in writing her pseudocode. She used some features of maintainable programs.

List **three** such features.

1 .....

.....

2 .....

.....

3 .....

.....[3]

(ii) Give **one** further feature that Rema has not used in her solution so far.

.....

.....[1]

2 Ali sets up user IDs and passwords for his program.

When the user types in their user ID, the program looks up the stored password for this user ID.

The stored password is the encrypted version of the user's password.

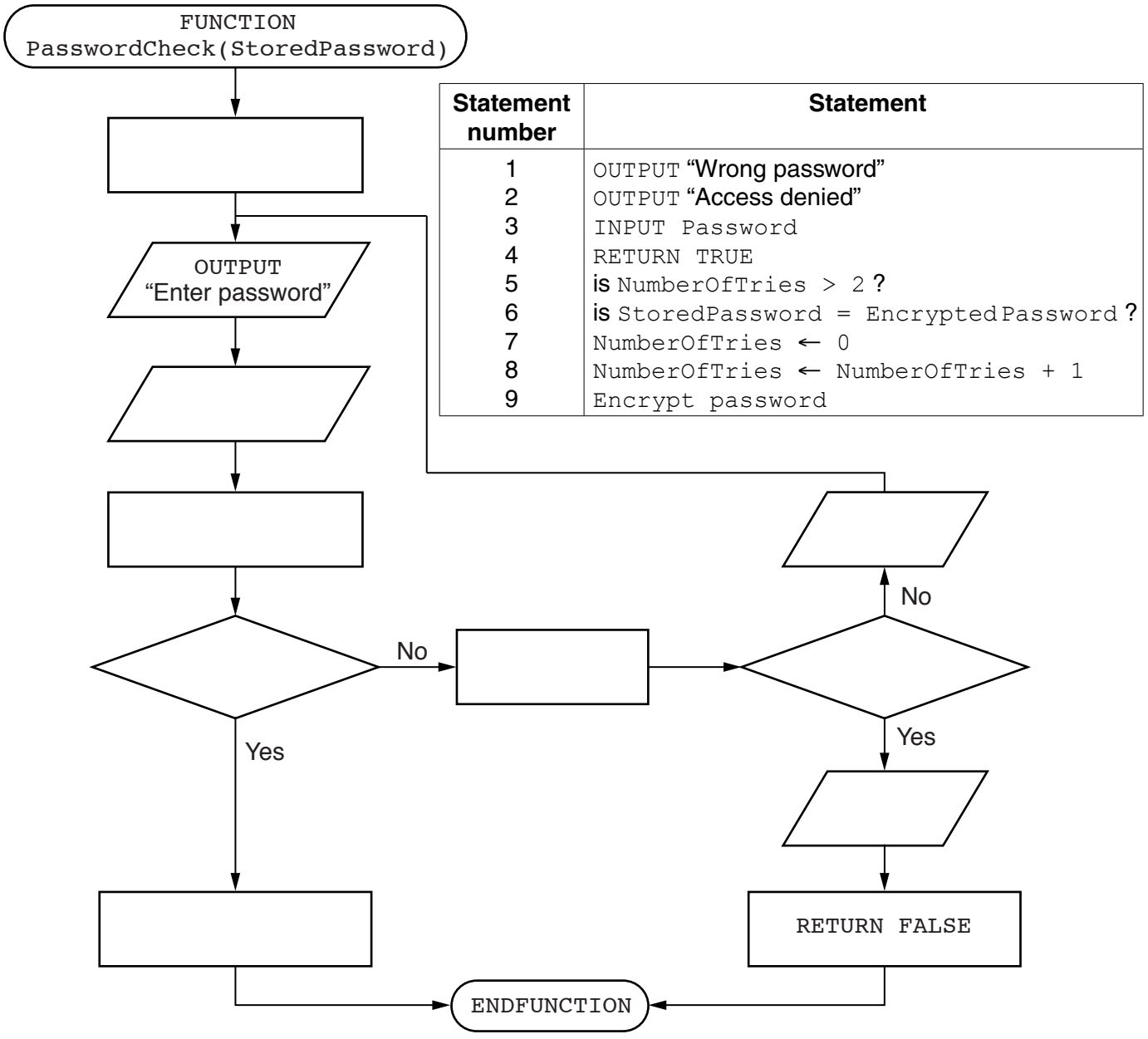
(a) The program calls the function `PasswordCheck` with the stored password as parameter.

If the user enters the correct password the function returns the value `TRUE`.

Each time the password is entered incorrectly, the message "Wrong password" is output.

If the user enters an incorrect password 3 times, the user is told that access is denied, and the function returns the value `FALSE`.

Complete the flowchart using the given statements. Ensure that only statement numbers are present on the flowchart.



Statement number	Statement
1	OUTPUT "Wrong password"
2	OUTPUT "Access denied"
3	INPUT Password
4	RETURN TRUE
5	is NumberOfTries > 2 ?
6	is StoredPassword = EncryptedPassword ?
7	NumberOfTries ← 0
8	NumberOfTries ← NumberOfTries + 1
9	Encrypt password

[9]

(b) Ali uses a sequential file of records to store the user IDs and encrypted passwords.

When a user types in their User ID, the program calls a function, `FindPassword`, with parameter `ThisUserID`.

The function searches each record in the file for `ThisUserID` and returns the encrypted password.

If `ThisUserID` is not stored in the file, the function returns an error code.

Complete the pseudocode:

```
FUNCTION FindPassword(ThisUserID : STRING) RETURNS .................
   DECLARE Found : BOOLEAN
   OPENFILE FOR INPUT                              // for reading
   Found ← FALSE

   WHILE  ..............................................................
      FILEREAD next record

      IF ...............................................................
         THEN

            ...........................................................
      ENDIF
   ENDWHILE

   IF  .................................................................
      THEN

          ..............................................................
      ELSE

          ..............................................................
   ENDIF
   CLOSEFILE
ENDFUNCTION
```

[8]

3 Ahmed runs his own business. He lays floor tiles in rooms for customers. Ahmed wants a program that calculates how many tiles he needs when he inputs the measurements of the length and width of the room he is working on.

(a) (i) Draw a screen design that is suitable for the following:

- to enter the length of the room in cm
- to enter the width of the room in cm
- to display the number of tiles required.

[4]

(ii) Explain how Ahmed can enter the data using your screen design in **part (i)**.

.....

.....

.....[1]

(b) The width of a room will measure at least 100 cm and less than 1000 cm.

The program must validate the width input.

Write a logic expression that is `TRUE` when the width stored in the variable `RoomWidth` is within the expected range.

.....

.....[3]

(c) Ali is going to write the program for Ahmed. The size of one floor tile is 30 cm × 30 cm. Ali knows that the following calculations are required:

```
TilesForWidth ← RoomWidth DIV 30
TilesForLength ← RoomLength DIV 30
TilesRequired ← TilesForWidth * TilesForLength
```

(i) If the room width measures 100 cm, give the value stored in `TilesForWidth`.

.....[1]

If the room measurements are not exact multiples of 30 cm, the number of tiles must be rounded up so that Ahmed has enough tiles.

Ali knows that he can use the `MOD` operator to test for this.

(ii) Write a logic expression that is `TRUE` when the room width is **not** an exact multiple of 30 cm.

.....[1]

(iii) Ahmed wants the program to add an extra 10% to the number of tiles required in case any tiles get broken during the work.

In a high-level language, write programming statements to calculate the total number of tiles required, `TilesRequired`, after the values for `RoomLength` and `RoomWidth` have been input and validated.

Language .....

Code .....

.....

.....

.....

.....

.....

.....

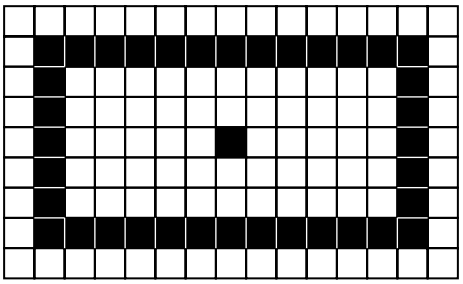
.....

.....

.....

.....[5]

4 Ahmed combines white tiles with tiles of one other colour to make a pattern. He draws a design. Here is one example:



Ahmed wants a program to calculate how many tiles he needs of each colour.

(a) Ali stores the design in a 2-dimensional array, `FloorDesign`. The length and width of the room will be no more than 35 tiles each.

(i) Suggest a value Ali should use to represent the white and coloured tiles.

White tile .....

Coloured tile .....[1]

(ii) Initially, every tile should be white.

In a high-level language, write statements to declare the array `FloorDesign` and initialise it.

Language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[5]

(b) Ali stores a design of size 15 tiles by 10 tiles in array `FloorDesign`.

Using pseudocode **or** a flowchart, show the algorithm to calculate the number of white tiles and the number of coloured tiles in the design.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[6]

5 **(a)** Dry-run the pseudocode algorithm by completing the trace table.

```
PROCEDURE Guess(a)
   x ← 8
   REPEAT
      IF a >= x
         THEN
            OUTPUT '1'
            a ← a - x
         ELSE
            OUTPUT '0'
      ENDIF
      x ← x / 2
   UNTIL a = 0
ENDPROCEDURE
```

a	x	a >= x
13		
	8	

OUTPUT .....[4]

(b) What does the procedure `Guess` do?

.....

.....[1]