

## COMPUTING

9597/01

Paper 1

October/November 2015

3 hours 15 minutes

Additional Materials: Pre-printed A4 Paper  
Removable storage device  
Electronic version of BUBBLE .TXT data file  
Electronic version of ADMISSIONS-DATA .TXT data file  
Electronic version of LICENCE-KEYS .TXT data file  
Electronic version of EVIDENCE .DOCX document

### READ THESE INSTRUCTIONS FIRST

Type in the EVIDENCE .DOCX document the following:

- Candidate details
- Programming language used

Answer **all** questions.

All tasks must be done in the computer laboratory. You are not allowed to bring in or take out any pieces of work or materials on paper or electronic media or in any other form.

All tasks and required evidence are numbered.

The number of marks is given in brackets [ ] at the end of each task.

Copy and paste required evidence of program listing and screenshots into the EVIDENCE .DOCX document.

**At the end of the examination, print out your EVIDENCE .DOCX document and fasten your printed copy securely together.**

This document consists of **10** printed pages and **2** blank pages.



Singapore Examinations and Assessment Board



CAMBRIDGE  
International Examinations



- 1 The file `ADMISSIONS-DATA.TXT` contains the daily total admissions to a theme park over a period of 50 days.

The task is to read the numbers from the file and display a sorted list.

You will program two different sort algorithms:

- A bubble sort.
- Either a quick sort or an insertion sort but not both.

### Task 1.1

Write code for a procedure to display a menu with the following options:

1. Read file data
2. Bubble sort
3. Quick sort / Insertion sort
4. End

### Task 1.2

Write the program code for a procedure to implement menu option 1.

#### Evidence 1

- The program code for the menu.
- Program code for menu option 1.

[5]

Options 2 and 3 will sort and display the sorted data.

The algorithm for a bubble sort is given in file `BUBBLE.TXT`.

Write program code as a procedure to implement the bubble sort.

#### Evidence 2

- The bubble sort code procedure.

[1]

Write program code as a procedure to implement the quick sort or the insertion sort.

#### Evidence 3

- Indicate the sort method used.
- The program for the sort method used.

[4]

**Task 1.3**

Additional code is to be written for each sort procedure. The sort methods will count and display the number of comparisons made in completing the sort process. This will provide an indicator of the efficiency of each algorithm.

Write the additional code to count and display the number of comparisons made for each sort method.

**Evidence 4**

- The output from menu option 2. [2]

**Evidence 5**

- The output from menu option 3. [2]

[Total: 14]



- 2 The pseudocode procedure below is given a denary number. The procedure then outputs the binary equivalent of the denary number.

```

PROCEDURE Converter(DenaryNumber : INTEGER)
  IF DenaryNumber = 0 OR DenaryNumber = 1
    THEN
      OUTPUT DenaryNumber
    ELSE
      OUTPUT DenaryNumber MOD 2
      Converter(DenaryNumber DIV 2)
    ENDIF
  ENDPROCEDURE

```

### Task 2.1

Write program code to implement the given procedure.  
Execute the procedure using 56 as the parameter.

#### Evidence 6

- Program code.
- Screenshot showing the output.

[6]

### Task 2.2

There is an error with the given algorithm.  
Describe the error and the effect it created on the output in Evidence 6.

#### Evidence 7




- Statement(s) to answer Task 2.2.

[1]

### Task 2.3

Make changes to the procedure `Converter` which will correct the error.

Draw up a list of test cases for the testing of the amended code, by completing a table with the following headings:

DenaryNumber	Purpose of the test	Expected output
		

#### Evidence 8

- The amended `PROCEDURE Converter` program code.
- The completed table.
- Screenshots for **two** of the tests.

[11]

[Total: 18]

- 3 When buying software, the purchaser is issued with a licence key. The product licence can be purchased for either one or three computers. A file is maintained of all the licence keys currently active and whether the licence was for a single-user or 3-users.

The licence key is a 10 character code as follows:

CCCCCCCCCD

- C = a randomly generated upper-case letter.
- D = a check digit character calculated from the preceding nine letters.

A new licence key is generated for each purchase.

An example key is produced as follows:

- randomly generated letters: FGKWRDFTA
- a set of products is calculated as shown:

Randomly generated letter	ASCII code	Multiplier	Product
F	70	1	70
G	71	2	142
K	75	3	225
W	87	4	348
R	82	5	410
D	68	6	408
F	70	7	490
T	84	8	672
A	65	9	585

- Then the total of the products is calculated;

Total	3350
-------	------

- The total 3350 is then divided by 11 to give remainder 6, which becomes the check digit character.
- This gives the complete licence key: FGKWRDFTA6
- If the calculation gives remainder 10, the check digit character used is X.



### Task 3.1

Design a function `LicenceKey` to generate a new licence key.

Write program code to implement the function.  
Test the function for **three** new licence keys.

#### Evidence 9

- Program code for the `LicenceKey` function.
- Screenshot(s) showing the generation of the three new licence keys. [10]

### Task 3.2

A file `LICENCE-KEYS.TXT` is maintained storing all licence keys which are currently active. This test file has 20 licence records. You will need this file for the programming which follows.

Typical data for two licences are shown:

SYNCTKMMF8 1  
                    indicates this is a single-user licence.

SNPHHUATV7 3 1  
                    purchased as a 3-user licence, but currently has only one registered user.

Write program code for a menu with the following options:

1. Purchase of a new licence for either a single-user or a 3-user licence
2. Register an additional user to an active 3-user licence
3. End

### Task 3.3

Write code as a procedure for menu option 1.

The requirement will be:

- Input from the user the type of licence.
- Generate the new licence key.
- Display licence key issued.
- Save the data as a new record in the `LICENCE-KEYS.TXT` file.
- Display final contents of `LICENCE-KEYS.TXT` file.

#### Evidence 10

- Program code for menu option 1.
- Screenshot(s), showing evidence for the issue of the two types of licence, displaying:
  - the licence key issued
  - the final contents of `LICENCE-KEYS.TXT` file.

[6]

### Task 3.4

Program menu option 2.  
Carry out **three** relevant tests.

#### Evidence 11

- Program code for menu option 2.
- Screenshot evidence of three test cases.

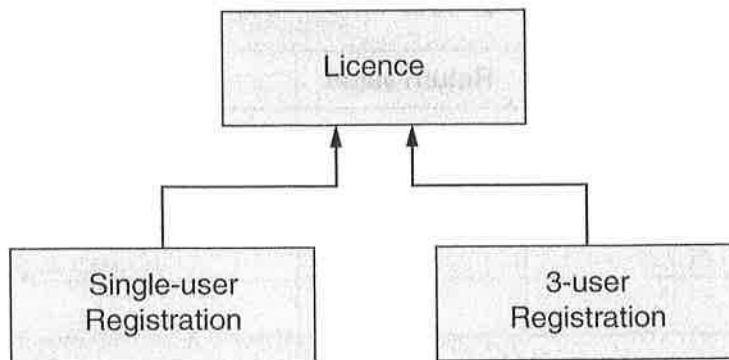
[5]

When a licence is purchased, the licence key, licence type (single-user or 3-user), purchase date and name of the purchaser are recorded.

A registration process then follows for each computer.

- The computer to which a licence is registered has its MAC address and the date of registration recorded.

The program design to manage purchases and registrations is to be implemented with object-oriented programming with the following three classes:



### Task 3.5

Write program code **only** for the three classes shown.  
Do not attempt to develop the application further.

### Evidence 12

- Program code for the three classes.

[9]

[Total: 30]



- 4 Users of a local area network each have a network account ID. The IDs have the format 2015\_NNNN, where N is a digit.

### Task 4.1

Complete the test case table with the addition of **three** more invalid User IDs. The reasons for their invalidity should be different.

The return value is a code as follows:

- 0 – valid User ID
- 1 – the User ID was not 9 characters
- you will use other integer numbers for other invalid cases.

Test Number	User ID	Return value	Explanation of the test case
1	2015_0987	0	Valid User ID
2			
3			
4			

### Evidence 13

- The completed test case table.

[6]

### Task 4.2

Write program code for a function to validate a User ID. The function header has the format:

```
FUNCTION ValidateUserID (ThisUserID : STRING) RETURNS INTEGER
```

Write a program to:

- Input an ID entered by the user
- Validate the input using the function `ValidateUserID`
- Output a message describing the validity of the input.

### Evidence 14

- Program code for the function `ValidateUserID`
- **Three** screenshots showing the testing of Test Numbers 2, 3, and 4.

[4]

[3]

You are to design an object-oriented program which simulates a print queue for a printer on a local area network (LAN). The print queue consists at any time of none, one, or more print jobs.

Each user can send a print job from any of the terminals on the LAN. Each terminal on the network is identified by an integer number in the range 1 to 172.

The program you are to design will record for each print job:

- the user ID
- the terminal number from which the print request was sent
- the file size (integer in Kbytes).



In practice, there are several print queues each associated with a different printer. Each printer is identified by a short name, such as `Room16`.

### Task 4.3

Design and write program code to define one or more classes and other appropriate data structures for this application.

#### Evidence 15

- Program code for the class(es).

[6]

A print queue behaves as a queue data structure.

Assume, for testing purposes:

- there is a single printer on the LAN
- the maximum print queue size for the printer is five print jobs.

The main program will simulate:

- the sending of print jobs to the printer by different users
  - that is, the addition of a print job to the print queue
- the output of a job from the print queue
  - that is, the removal of a print job from the print queue

The program design has the following menu:

1. New print job added to print queue
2. Next print job output from printer
3. Current print queue displayed
4. End

The program simulates the working of the print queue as follows:

1. The empty print queue is initialised.
2. The program user selects menu options 1, 2 and 3 in any order.
3. The program user selects menu option 4.

### Task 4.4

Write program code to:

- display the main menu
- input the choice by the user
- run the appropriate code for the choice made.

#### Evidence 16

- The program code.

[3]



**Task 4.5**

Write program code to initialise the print queue for the Room16 printer.

Write program code to display the current state of the print queue.

**Evidence 17**

The program code for:

- initialising the print queue
- output of the current print queue.

[6]

**Task 4.6**

Write program code to add a new print job to the print queue.

The requirement will be:

- program user enters data for the new print job
- print job is added to the print queue.

Test the code by adding one new print job.

**Evidence 18**

- Program code to add a new print job.
- Screenshot following menu option 1 then menu option 3.

[4]

**Task 4.7**

Write program code to output the next print job from the printer.

This code will execute from menu option 2.

Test the code by:

- adding three print jobs
- outputting the next print job.

**Evidence 19**

- Program code to output next print job.
- Screenshot following menu option 1 three times, then menu option 2, and menu option 3. [6]

[Total: 38]