Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME									
CENTRE NUMBER						CANDIDATE NUMBER			

COMPUTER SCIENCE **9608/04**

Paper 4 Further Problem-solving and Programming Skills **For Examination from 2015**

SPECIMEN PAPER

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 in the spaces at the top of this page.
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.

Answer **all** questions.
No marks will be awarded for using brand names for software packages or hardware.
No calculators allowed.

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 **15** printed pages and **1** blank page.

Throughout the paper you will be asked to write either **pseudocode** or **program code**.

Complete the statement to say which high-level programming language you will use.

Programming language used: .....

1 **(a)** Complete the algorithm, written in pseudocode, for a binary search.
The data being searched is stored in the array `SearchData[63]`. The item of data being searched is stored in the variable `SearchItem`.

```
X ← 0
Low ← 1
High ← .....
WHILE (High>=Low) AND (.....)
   Middle ← INT((High + Low)/2)
   IF SearchData[Middle] = SearchItem
      THEN
         X ← Middle
      ELSE
         IF SearchData[Middle] < SearchItem
            THEN
               Low ← Middle + 1
            ELSE
               IF SearchData[Middle] > SearchItem
                  THEN
                        .....
               ENDIF
         ENDIF
   ENDIF
ENDWHILE
```

[3]

(b) **(i)** The binary search only works if the data in the array being searched is:

.....

[1]

(ii) The maximum number of comparisons that are required to find an item which is present in the array `SearchData` is:

.....

[1]

(iii) At the end of the algorithm, the variable `X` contains:

either the value ..... which indicates .....

or the value ..... which indicates ..... [4]

(c) You will change the binary search algorithm to a recursive algorithm and write the equivalent program code in the form of a procedure. Name the recursive procedure `BinarySearch`.

Use these variables.

Variable	Data Type	Description
`SearchData`	`ARRAY[63] : INTEGER`	global array
`SearchItem`	`INTEGER`	global variable
`X`	`INTEGER`	global variable
`Low`	`INTEGER`	parameter
`High`	`INTEGER`	parameter
`Middle`	`INTEGER`	local variable

Write **program code** for the recursive procedure `BinarySearch`.

.....

.....

.....

.....

.....

.....

..... [5]

(d) Write the initial call to the recursive procedure.

..... [1]

2 A manufacturer has an assembly line that produces a particular product. At the end of the assembly process, each product item is comprehensively tested to decide whether that item is acceptable or not. The tests are split into three groups:

- Group 1: tests to check all dimensions are correct
- Group 2: tests to check strength at various points on the product item
- Group 3: tests to check paint colour and coverage

Only if the item passes all three group tests is it accepted. If the Group 1 tests are passed, but exactly one of the other two group tests fails, the item is sent for repair. Otherwise the item is rejected.

(a) Complete the decision table showing all the possible outcomes and results.

Conditions	Group 1 tests								
	Group 2 tests								
	Group 3 tests								
Actions	Accepted								
	Repair								
	Rejected								

[4]

(b) Simplify your solution by removing redundancies.

Conditions	Group 1 tests								
	Group 2 tests								
	Group 3 tests								
Actions	Accepted								
	Repair								
	Rejected								

[5]

(c) The simplified table produced in **part (b)** is used to design program code.
Three functions are already available: `G1Tests`, `G2Tests` and `G3Tests`.
These functions return `TRUE` or `FALSE`, indicating the success or otherwise of the group tests.

Write code for a function `Reject` which will return `TRUE` if the product item is to be rejected, otherwise the function will return `FALSE`.

[3]

3 A linked list Abstract Data Type (ADT) has these associated operations.

1. Create linked list
2. Add item to linked list
3. Remove item from linked list

Each node in a linked list consists of data and a pointer to the next item in the linked list. Items in the linked list are maintained in order.

(a) A linked list is to be set up that stores names in alphabetical order. Show the final state of this linked list after the following operations are carried out.

```
CreateLinkedList
AddItem("Nushie")
AddItem("Kellie")
AddItem("Scarlett")
RemoveItem("Nushie")
AddItem("Jon")
```

[2]

(b) A programming language provides built-in array data structures. This linked list is to be implemented using these array data structures.

Define a record type, `ListNode`, for each node.

.....

.....

.....

.....

.....

..... [3]

(c) Write an array declaration to reserve space for 50 nodes in array `NameList`.

.....

[2]

(d) (i) The `CreateLinkedList` operation links all nodes to form the free list and initialises the `HeadPointer` and `FreePointer`.

Complete the diagram to show the value of all pointers.

`NameList`

`HeadPointer`

	`Name`	`Pointer`
[1]		
[2]		
[3]		
[4]		
:		
[49]		
[50]		

`FreePointer`

[4]

(ii) Write **pseudocode** to implement the `CreateLinkedList` operation.

.....

.....

.....

.....

.....

[2]

(e) The algorithm for adding an item into the linked list is implemented as a procedure with the header:

```
PROCEDURE AddItem(NewItem)
```

where `NewItem` is the new item to be added to the linked list.

Study the algorithm that will add a new item, `NewItem`, to the linked list.

Variable	Data Type	Description
NameList	ARRAY[50] : ListNode	
NewItem	STRING	item to be added
FreePointer	INTEGER	pointer to next free node in array
HeadPointer	INTEGER	pointer to first node in the list
CurrentPointer	INTEGER	pointer to current node
PreviousPointer	INTEGER	pointer to previous node accessed

```
01 PROCEDURE AddItem(NewItem)
02 //
03   NameList[FreePointer].Name ← NewItem
04   CurrentPointer ← ……..
05 //
06   REPEAT
07     IF NameList[…….].Name < NewItem
08       THEN
09         PreviousPointer ← CurrentPointer
10         CurrentPointer ← ……
11     ENDIF
12   UNTIL NameList[CurrentPointer].Name > NewItem
13 //
14   IF CurrentPointer = HeadPointer
15     THEN
16       NameList[FreePointer].Pointer ← HeadPointer
17       HeadPointer ← FreePointer
18     ELSE
19       NameList[FreePointer].Pointer
20                         ← NameList[PreviousPointer].Pointer
21       NameList[PreviousPointer] ← FreePointer
22   ENDIF
23   FreePointer ← NameList[FreePointer].Pointer
24 ENDPROCEDURE
```

(i) Complete the algorithm on page 8. [3]

(ii) Write a comment for line `02` (to explain the code on line `03`).

.....

..... [1]

(iii) Write a comment for line `05` (to explain the code on lines `06` to `12`).

.....

.....

.....

..... [2]

(iv) Write a comment for line `13` (to explain the code on lines `14` to `22`).

.....

.....

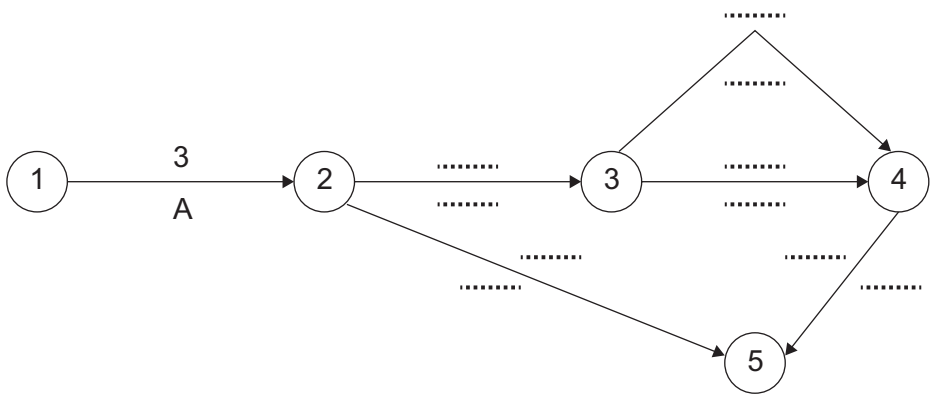
.....

..... [3]

4 A software development project consists, in part, of these activities.

		Weeks to complete
A	identify requirements	3
B	produce design	5
C	write code	9
D	black box testing	2
E	acceptance testing	3
F	prepare documentation	6

From this data, a Program Evaluation Review Technique (PERT) chart is constructed.



(a) Complete the PERT chart. [4]

(b) (i) State the critical path.

..... [1]

(ii) State the minimum time for the completion of this development.

..... [1]

(c) For activity D:

(i) state the earliest start time.

..... [1]

(ii) state the latest finish time.

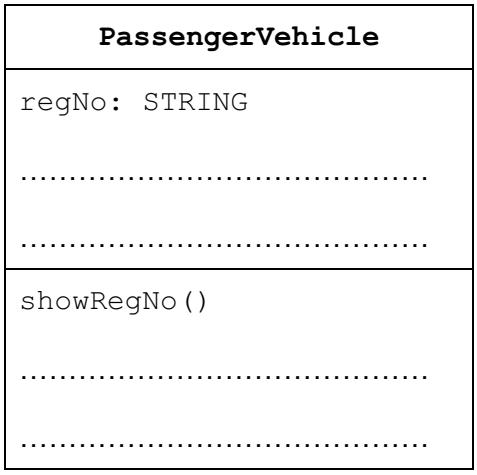
..... [1]

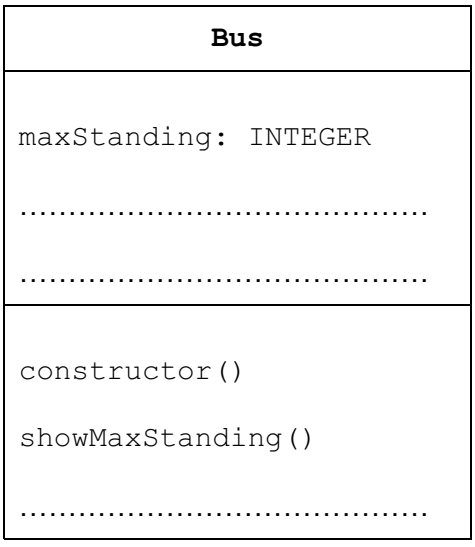
BLANK PAGE

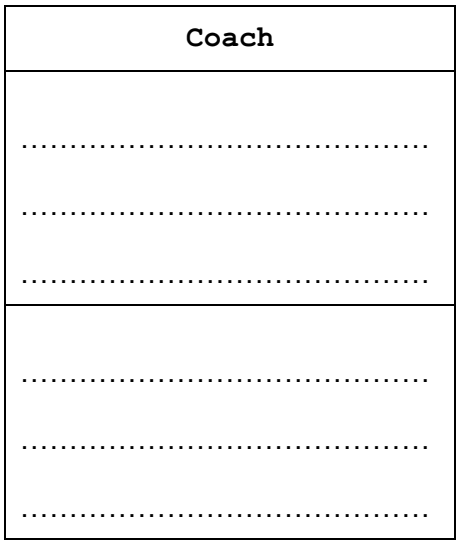
5 A transport company has a number of vehicles which can carry passengers. Each vehicle is classified either as a bus or as a coach. All vehicles have a registration number and have a certain number of seats for the passengers. A bus can have a maximum number of standing passengers, but a coach is not allowed to carry any standing passengers. Some of the coaches are fitted with seat belts, but seat belts are never fitted in a bus.

Object-oriented software is written to process data about the vehicles.

(a) Complete the class diagram.







[4]

(b) Write **program code** for the `PassengerVehicle` class.

[5]

(c) Write **program code** for the `Bus` class.

[5]

The transport company has a bus with registration number 'NBR 123'. The bus has seats for 51 passengers and is allowed to carry 10 standing passengers.

(d) Write **program code** to:

(i) create an instance of an object with identifier `pv1` that has the properties of the bus.

.....

.....

.....

.....

..... [1]

(ii) demonstrate the successful creation of the object by displaying its property values.

.....

.....

.....

.....

..... [3]