Cambridge International **A Level**

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

9190585504

COMPUTING **9691/32**

Paper 3 **May/June 2016**

2 hours

Candidates answer on the Question Paper.

No Additional Materials are required.

No calculators allowed.

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.
No marks will be awarded for using brand names of software packages or hardware.

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

DC (KN/JG) 126393

CAMBRIDGE International Examinations

[Turn over

1 The grammar for a language is defined using a set of Backus-Naur rules as follows:

Rule Number	Rule
1	`<article>  ::= a | the`
2	`<verb>  ::= sat | walked | ate | slept`
3	`<noun>  ::= dog | cat | person | snake`
4	`<nounPhrase>  ::= <article><noun> | <noun>`
5	`<verbPhrase>  ::= <nounPhrase><verb>`
6	`<sentence>  ::= <verbPhrase> | <sentence><nounPhrase>`

(a) A rule may be recursive.

State what is meant by recursive.

.....

.....

Identify the rule which is recursive.

Rule number ..... [2]

(b) Using the above rules, explain the following statements:

(i) `dog`
is a valid `<nounPhrase>`

.....

..... [2]

(ii) `a puppy sat`
is an invalid `<sentence>`

.....

.....

.....

..... [2]

(iii) `a cat slept the snake`
is a valid `<sentence>`

.....

.....

.....

.....

.....

..... [4]

(iv) The adverbs `quietly`, `quickly` and `slowly` are to be included in the rules.
When an adverb is used in a sentence it always precedes the verb.

The rules are to be changed to make:

`the cat quietly slept`
a valid `<sentence>`

This needs a new rule. Write this new rule.

.....

It also needs a change to an existing rule. Write the amended rule.

.....

.....

.....

..... [3]

2 A number of music bands use a manager to organise their bookings.
Each manager will manage one or more bands.
Each band:

- has one manager
- has a set fee payable for a booking
- plays a particular genre of music

Data for bands and managers are to be organised and stored in a relational database.

(a) A first attempt at the table design is:

```
MANAGER(ManagerName, Telephone, BandName, Genre, NumberInBand, SetFee)
```

ManagerName	Telephone	BandName	Genre	NumberInBand	SetFee($)
Yi Ling Chen	908765	Rachael Daz	Covers	1	200
		Jazz Tones	Modern jazz	4	450
Lockwood Bros	674442	Midnight Blues	Modern jazz	5	350
		Loose Beats	Dance	3	400
		Asyraf Duo	Covers	2	250
		The Flying Fleas	Hip hop	8	700
Rachael Wang	118976	Li Wei Chen	Covers	1	250
Gig Associates	567575	Buster Beats	Reggae	6	600
		Hits Forever	Covers	3	300
		Rap Incorporated	Hip hop	5	800

State why the table `MANAGER` is not in First Normal Form (1NF).

.....

..... [1]

(b) The database design is changed to:

```
BAND(BandName, Genre, NumberInBand, SetFee, ManagerName)

MANAGER(ManagerName, Telephone)
```

The primary key for each table has not been shown.

Assume that the band names and manager names are unique.

(i) State the relationship that exists between `BAND` and `MANAGER`.

..... [1]

(ii) Explain how this relationship is implemented.

.....

.....

..... [2]

(c) The database is to be re-designed. It will store a date for each booking made.

A band does not have more than one booking on any one date.

This additional table is suggested:

BOOKING(BandName, BookingDate, Genre, NumberInBand, SetFee)

Describe why the table BOOKING is not in Second Normal Form (2NF).

.....

.....

.....

..... [2]

(d) There are three issues which the final database design must address:

Issue 1 - Managers often negotiate a fee which is higher or lower than the set fee.

Issue 2 - Some bands are now able to perform more than one booking on any one date.

Issue 3 - The database is to store data for the venue where the band will perform.

The final design (not showing all of the primary keys) is:

BAND(BandName, Genre, NumberInBand, SetFee, ManagerName)

MANAGER(ManagerName, Telephone)

BOOKING(BandName, BookingDate, BookingTime, AgreedFee, VenueName)

VENUE(VenueName, VenueAddress, Capacity)

(i) State how Issue 1 has been addressed.

.....

..... [1]

(ii) State how Issue 2 has been addressed.

.....

..... [1]

(iii) State how Issue 3 has been addressed.

.....

..... [1]

(iv) Complete the table below to show the primary key and any foreign key(s) present in each database table.

Table	**Primary key**	**Foreign key(s)** (if any)
BAND	BandName	....
MANAGER	ManagerName	....
BOOKING	.....	....
VENUE	.....	....

[4]

3 **(a)** Convert the following infix expressions into reverse Polish notation:

(i) `x / (a + b)`

..... [1]

(ii) `p ^ 2 + (2 + q) / 3`
Note: ^ denotes 'to the power of'

..... [2]

(b) Convert the following reverse Polish expression to infix notation:

`3 a b + c + d + e - *`

..... [2]

(c) The reverse Polish expression `RPNString` consists of a sequence of characters. Each character is either:

- a single character identifier or digit
- an operator

The following algorithm inputs `RPNString` and output the corresponding infix expression `InfixString`.

The algorithm uses a stack data structure, `Stack`.

```
INPUT RPNString

WHILE characters in RPNString

   ThisChar ← next character

   IF ThisChar is an identifier or digit

      THEN

         Push ThisChar to Stack

      ELSE   // next character is an operator

         Pop item from Stack and store as Temp

         InfixString ← Temp

        // the & operator concatenates two strings

         InfixString ← ThisChar & InfixString

         Pop item from Stack and store as Temp

         InfixString ← Temp & InfixString

        Place brackets around InfixString

         Push InfixString to Stack

    ENDIF

ENDWHILE
```

Complete the trace table for the input of the reverse Polish expression:

```
x y +
```

The first two iterations of the `WHILE` loop have been done for you.

RPNString	**ThisChar**	**Stack contents**	**Temp**	**InfixString**
x y +	x	x		
	y	y x		

[8]

4 **(a)** A computer system stores integers in 8-bit two's complement form.

State the two denary numbers represented by Byte 1 and Byte 2.

Byte 1								Byte 2							
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	1

Byte 1 .....

Byte 2 ..... [2]

(b) Byte 3 and Byte 4 together are used to represent a Binary Coded Decimal (BCD) number.

State the denary number.

Byte 3								Byte 4							
1	0	0	1	0	0	1	1	0	0	0	0	0	1	1	1

Denary ..... [2]

(c) Byte 5 and Byte 6 together represent a 16-bit colour code used in a drawing program.

Byte 5								Byte 6							
0	1	1	0	1	0	1	0	1	1	1	1	0	1	0	1

The drawing program displays the colour code as a hexadecimal number.

State the hexadecimal number for this colour code. ..... [2]

(d) Most computer systems need to store and process real numbers.

A computer uses two bytes to store a real number.

The first byte (Byte 7) stores the mantissa. The second byte (Byte 8) stores the exponent.

Both mantissa and exponent use two's complement form.

Byte 7								Byte 8							
1	0	1	0	1	0	0	0	0	0	0	0	1	0	1	1

(i) State, without any working out, how you can recognise that this 16-bit pattern (Byte 7 and Byte 8) is a negative number.

..... [1]

(ii) State the denary values for the mantissa and the exponent represented by Byte 7 and Byte 8.

Mantissa .....

.....

Exponent .....

..... [2]

(iii) Calculate the denary value represented by Byte 7 and Byte 8.

..... [1]

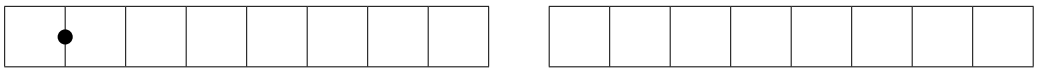
(e) State how you can recognise that this 16-bit pattern (Byte 7 and Byte 8) is normalised.

.....

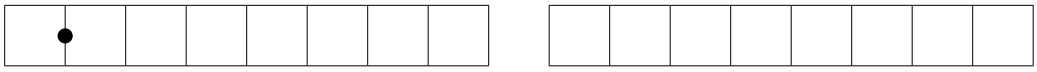
..... [1]

(f) Write the binary patterns for the largest and smallest positive number that can be represented with this 16-bit format for normalised real numbers.

Smallest



Largest



[2]

5 A company stores customer data in a database using linked lists.

An array of records of data type `CustomerNode` implements a linked list of surnames.

The linked list stores the surnames in alphabetical order.

The `CustomerNode` record has two fields as defined below:

```
TYPE CustomerNode
   Surname     : STRING
   Pointer     : INTEGER
ENDTYPE
```

The linked list is created using the array and variable shown below.

Identifier	Data type	Description
`SurnameList`	`ARRAY[1 : 10000] OF CustomerNode`	An array to store the surname and pointer values
`HeadSurname`	`INTEGER`	Stores the index position of the node at the head of the linked list

(a) (i) Explain the difference between a static data structure and a dynamic data structure.

.....

.....

.....

..... [2]

(ii) State a benefit gained from using a dynamic data structure to implement a linked list.

.....

..... [1]

The database initially contains the records of four customers. These have the surnames:

`GREENE`, `HASAN`, `ALI` and `ABBOT`.

The data are stored as shown below:

HeadSurname : 4

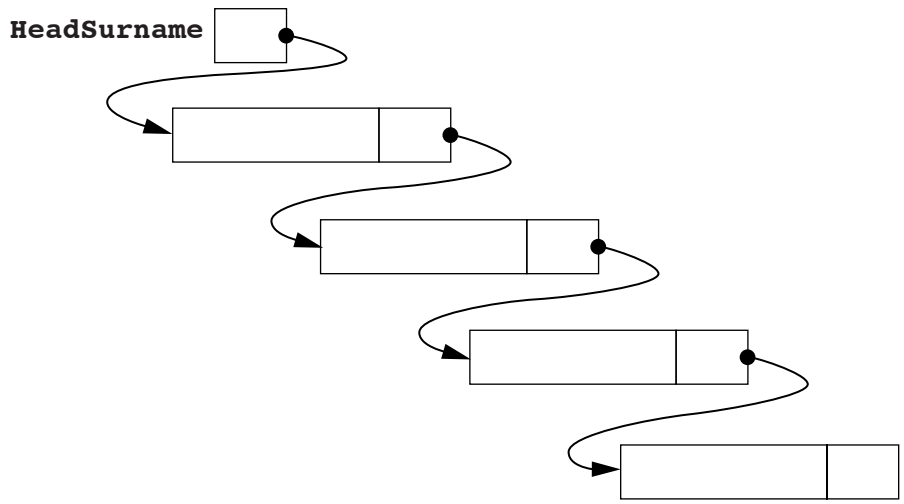
	SurnameList	
	Surname	**Pointer**
1	GREENE	2
2	HASAN	0
3	ALI	1
4	ABBOT	3
:		
:		
9999		
100000		

(b) State the value of:

`SurnameList[HeadSurname].Surname` .....

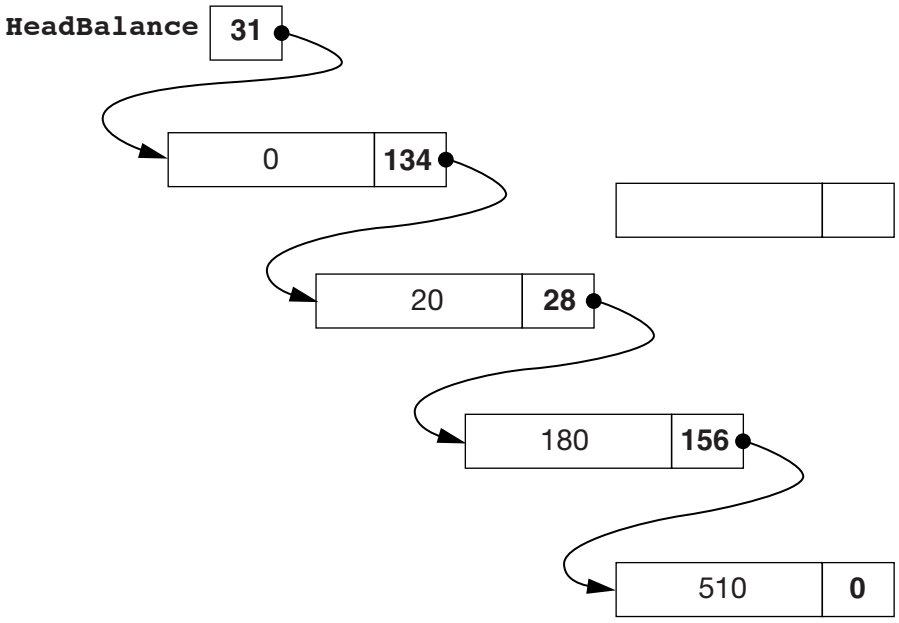
`SurnameList[3].Pointer` ..... [2]

(c) Complete the linked list diagram by adding the data and pointer values for each customer node.



[3]

(d) A similar linked list is maintained by the database management software to organise customers according to the money owed to the company (the balance). The data structure used is a second linked list. The pointer to the head of this linked list is `HeadBalance`.



(i) A new customer with an outstanding balance of $130 is to be inserted into the linked list.

On the diagram above, add the required data to the empty node. Adjust pointer(s) where necessary. [3]

(ii) Describe the method to locate the position in the linked list for inserting a new record. Include how to change the pointer values.

Ignore the special case in which the linked list is empty so that the new record becomes the first record in the linked list.
Do not write pseudocode or program code.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [4]

Question 6 begins on page 16.

6 (a) There are four key features that apply to most robots. They are:

- mechanical devices
- movable
- able to sense their surroundings
- controlled by a computer program

One application of the use of a robot is a paint sprayer on a car production line.

Explain how the four features listed above apply to this application.

The robot is a mechanical device.

.....

.....

.....

It is movable.

.....

.....

.....

It can sense its surroundings.

.....

.....

.....

It is controlled by a computer program.

.....

.....

..... [4]

(b) Name **two** hardware components that would be found in the paint sprayer.

Explain how each hardware component is used in the operation of the robot.

1 .....

.....

.....

.....

2 .....

.....

.....

..... [4]

7 The following are the initial lines of a program written in a high-level language.

```
// scheduling program
// program written 09 June 2015
DECLARE Counter       : INTEGER ;
DECLARE Jobs[1000]    : STRING ;
DECLARE Position[1000]: INTEGER ;
CONSTANT ChangeRate = 5.0 ;

// start of main program
CALL InitialiseGrid ;
WHILE Counter < 1000
    a = x + y ;
    b = x + y + 7 ;
```

This program is about to be translated by the language compiler.

(a) State what is meant by:

(i) source code

.....

..... [1]

(ii) object code

.....

..... [1]

(b) During the lexical analysis stage, the compiler will use a keyword table and a symbol table.

(i) Describe, in general, the information contained in a keyword table.

.....

.....

.....

..... [2]

(ii) State **three** entries that will be in the keyword table for the given program.

1 .....

2 .....

3 ..... [1]

(iii) Describe, in general, the information contained in a symbol table.

.....

.....

..... [2]

(iv) State **three** entries that will be entered in the symbol table for the given program.

1 .....

2 .....

3 ..... [1]

(c) Explain what happens during the lexical analysis stage of compilation. Include how the contents of the keyword table and symbol table are used.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

(d) The final stage of the compilation process is to optimise the code.

(i) State what is meant by code optimisation.

.....

..... [1]

(ii) Suggest where in the given program optimisation may be possible.

.....

..... [1]

BLANK PAGE