Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

4137415794

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2018**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **14** printed pages and **2** blank pages.

DC (RW) 150987/4

CAMBRIDGE
International Examinations

Question 1 begins on the next page.

1 Consider the following user-defined data type.

```
TYPE Book
     DECLARE ISBN     : INTEGER
     DECLARE Author   : STRING
     DECLARE Title    : STRING
     DECLARE Supplier : (Amazone, Stones, Smiths, Blackwalls, Greens,
                        Coals, Boarders)
ENDTYPE
```

(a) Name the data type of `Book`.

....[1]

(b) Name the non-composite data type used in the `Supplier` declaration.

....[1]

(c) **(i)** Write a pseudocode statement to declare a variable, `BestSeller`, of type `Book`.

....[1]

(ii) Write a pseudocode statement to assign "`John Williams`" to the author of `BestSeller`.

....[1]

2 **(a)** A computer system stores real numbers using floating-point representation. The floating-point numbers have:

- eight bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(i) Calculate the denary value of the following floating-point number.

Mantissa

0	0	1	1	1	0	0	0

Exponent

0	1	1	1

Show your working.

Working ....

....

....

....

....

Answer .... [3]

(ii) State how you know the floating-point number in **part (a)(i)** is not normalised.

....

....[1]

(iii) Normalise the floating-point number in **part (a)(i)**.

Mantissa

Exponent

[2]

(b) **(i)** Write the largest positive number that this system can represent as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(ii) Write the smallest positive number that can be stored as a normalised floating-point number in this format.

Mantissa									**Exponent**			

[2]

(c) The number of bits available to represent a real number is increased to 16.

State the effect this has on the numbers that can be represented, if the additional four bits are used in the:

(i) mantissa ....

....[1]

(ii) exponent ....

....[1]

(d) A student enters the following code into an interpreter.

```
X = 0.1
Y = 0.2
Z = 0.3
OUTPUT (X + Y + Z)
```

The student is surprised to see the output:

```
0.6000000000000001
```

Explain why this is output.

....

....

....

....

....

....[3]

3 A local college has CSMA/CD in operation on its Local Area Network (LAN).

(a) One function of CSMA/CD is to monitor traffic on the network.

State **two** other tasks performed by CSMA/CD.

1 ....

2 ....
[2]

(b) The network uses the TCP/IP protocol to transfer files across the network.

(i) State **three** functions of the **TCP** part of this protocol.

1 ....

....

2 ....

....

3 ....

....
[3]

(ii) State **two** functions of the **IP** part of this protocol.

1 ....

2 ....
[2]

(iii) Identify **one** other common protocol that could be used to transfer files across the college network.

....[1]

(c) Protocols are essential for successful transmission of data over a network. The TCP/IP protocol suite operates on many layers.

Give an appropriate protocol for each layer in the table.

Layer	Protocol
Application	
Transport	
Internet	

[3]

(d) The TCP/IP protocol is used to send an email message from one node on a LAN to a node on a different LAN.

State the steps that take place when the email message is sent and received.

....

....

....

....

....

....

....

....[4]

4 **(a)** A Boolean expression corresponds to the following truth table.

INPUT			OUTPUT
A	**B**	**C**	**X**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(i) Write the Boolean expression for the truth table by applying the sum-of-products.

X = ...[2]

(ii) Complete the Karnaugh Map (K-map) for the truth table.

AB

		00	**01**	**11**	**10**
C	**0**				
	1				

[1]

(iii) The K-map can be used to simplify the expression in **part (a)(i)**.

Draw loop(s) around appropriate groups of 1s in the table in **part (a)(ii)** to produce an optimal sum-of-products. [3]

(iv) Write the simplified sum-of-products expression for your answer to **part (a)(iii)**.

X = ...[3]

(b) A logic circuit with four inputs produces the following truth table.

INPUT				OUTPUT
A	**B**	**C**	**D**	**X**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(i) Complete the K-map that corresponds to the truth table.

AB

CD

[4]

(ii) Draw loop(s) around appropriate groups of 1s in the table in **part (b)(i)** to produce an optimal sum-of-products. [2]

(iii) Write the simplified sum-of-products expression for your answer to **part (b)(ii)**.

X = ...[2]

5 A computer process can be in one of three states: running, ready or blocked.

(a) Explain how the processes are affected when the following events take place.

(i) The running process needs to read a file from a disk.

....

....

....

....[2]

(ii) The running process uses up its time slice.

....

....

....

....[2]

(b) **(i)** State the conditions that are necessary for a process to move from the ready to the running state.

....

....

....

....[2]

(ii) State the conditions that are necessary for a process to move from the blocked to the ready state.

....

....

....

....[2]

(c) Give **three** reasons why process scheduling is needed.

1 ....

....

2 ....

....

3 ....

....

[3]

6 The compilation process has a number of stages. The first stage is lexical analysis.

A compiler uses a keyword table and a symbol table. Part of the keyword table is shown.

- Tokens for keywords are shown in hexadecimal.
- All of the keyword tokens are in the range `00` – `5F`.

Keyword	**Token**
←	01
*	02
=	03
⋮	⋮
IF	4A
THEN	4B
ENDIF	4C
ELSE	4D
FOR	4E
STEP	4F
TO	50
INPUT	51
OUTPUT	52
ENDFOR	53

Entries in the symbol table are allocated tokens. These values start from 60 (hexadecimal).

Study the following code.

```
Start ← 1
INPUT Number
// Output values in a loop
FOR Counter ← Start TO 12
    OUTPUT Number * Counter
ENDFOR
```

(a) Complete the symbol table to show its contents after the lexical analysis stage.

Symbol	**Token**	
	Value	**Type**
`Start`	`60`	Variable
`1`	`61`	Constant

[3]

(b) The output from the lexical analysis stage is stored in the following table. Each cell stores one byte of the output.

Complete the output from the lexical analysis stage. Use the keyword table and your answer to **part (a)**.

`60`	`01`														

[2]

(c) The output of the lexical analysis stage is the input to the syntax analysis stage.

Identify **two** tasks in syntax analysis.

1 ....

....

2 ....

....

[2]

(d) The final stage of compilation is optimisation.

(i) Code optimisation produces code that minimises the amount of memory used.

Give **one** additional reason why code optimisation is performed.

....

....[1]

(ii) A student uses the compiler to compile some different code.

After the syntax analysis stage is complete, the compiler generates object code.

The following lines of code are compiled.

```
X  ←  A + B
Y  ←  A + B + C
Z  ←  A + B + C + D
```

The compilation produces the following assembly language code.

```
LDD 236     //     loads value A to accumulator
ADD 237     //     adds value B to accumulator
STO 512     //     stores accumulator in X
LDD 236     //     loads value A to accumulator
ADD 237     //     adds value B to accumulator
ADD 238     //     adds value C to accumulator
STO 513     //     stores accumulator in Y
LDD 236     //     loads value A to accumulator
ADD 237     //     adds value B to accumulator
ADD 238     //     adds value C to accumulator
ADD 239     //     adds value D to accumulator
STO 514     //     stores accumulator in Z
```

Rewrite the assembly language code after it has been optimised.

....

....

....

....

....

....

....

....

....

....[5]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

7142109804

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2021**

2 hours

You must answer on the question paper.

No additional materials are needed.

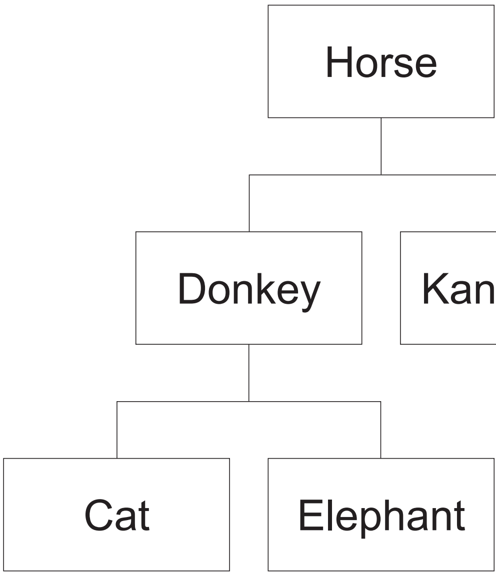
INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **20** pages. Any blank pages are indicated.

DC (DH) 198992/5

1 An ordered binary tree stores the following data:



(a) Identify the data in the root node of the binary tree.

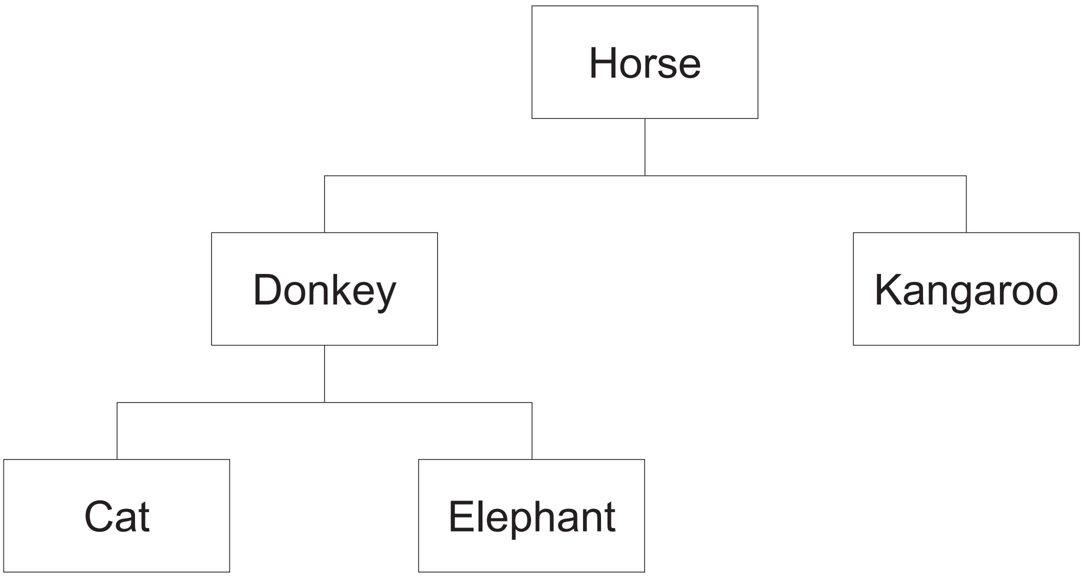
.... [1]

(b) Identify the data in **one** leaf node from the binary tree.

.... [1]

(c) Complete the binary tree by adding the following data in the order given:

Fish Iguana Rabbit Jaguar



[2]

(d) Explain how an algorithm will search the binary tree to find **Elephant**.

....

....

....

....

.... [2]

2 A company stores bookings in a random file.

For each booking, the booking ID, customer ID, item ID and quantity are stored. These four values are all integers.

(a) Write the **pseudocode** record declaration for the data type `Booking`.

....

....

....

....

....

....

.... [2]

(b) Each booking ID is a value between 100 000 and 999 999 inclusive.

The hash value is calculated by dividing the booking ID by 100 000 and adding 3 to the remainder.

(i) The function `Hash()` takes the booking ID as a parameter, calculates and returns the hash value.

Write **program code** for the function `Hash()`.

Programming language ....

Program code

....

....

....

....

.... [2]

(ii) Calculate the hash value for each booking ID in the table.

Booking ID	Hash value
5 012 345	
8 212 350	

[1]

(c) The function `StoreBooking()` takes a record as a parameter and stores it in the random file `TheBookings.dat`. The function uses `Hash()` to calculate the hash value for that record. The record is only stored if the value at the hashed value is `NULL`.
`FALSE` is returned if there is already a record in that location and `TRUE` otherwise.

You can assume that the file exists.

Write **pseudocode** for the function `StoreBooking()`.

.... [7]

(d) Explain how exception handling can be used when reading from a file.

....

....

....

.... [2]

3 Ejaz is creating a program that will allow the user to create quizzes. He is using object-oriented programming (OOP).

There are two classes: `QuestionClass` and `QuizClass`.

The class attributes and methods are in the following tables. All attributes are declared as private.

QuestionClass	
`Question : STRING`	`// stores the question`
`Answer : STRING`	`// stores the correct answer`
`Difficulty : INTEGER`	`// stores the difficulty as an integer` `// from 0(easy) to 10(hard)`
`Constructor(QuestionP, AnswerP, DifficultyP)`	`// creates an instance of QuestionClass` `// sets the attributes to the parameter` `// values`
`GetQuestion()`	`// returns the question`
`GetDifficulty()`	`// returns the difficulty level`
`GetAnswer()`	`// returns the answer`

QuizClass	
`Questions : ARRAY[0:19] OF QuestionClass`	`// stores maximum 20 questions of` `// type QuestionClass`
`NumberOfQuestions : INTEGER`	`// stores the number of questions` `// in this quiz`
`Constructor()`	`// creates an instance of` `// QuizClass` `// initialises NumberOfQuestions` `// to 0`
`AddQuestion()`	`// adds the parameter question to` `// the array` `// increments NumberOfQuestions`
`GetQuestion()`	`// returns the next question to be` `// asked`
`CheckAnswer()`	`// takes an answer as a parameter` `// and returns TRUE if correct`

(a) Write **program code** to define the class `QuizClass`. You are only required to write code for the attribute declarations and constructor.

If you are writing in Python, include attribute declarations using comments.

Use your programming language's constructor method.

Programming language ....

Program code

....

....

....

....

....

....

....

.... [4]

(b) The QuizClass method AddQuestion() takes a question object as a parameter and stores it in the next available location in the array Questions. It returns TRUE if it is successfully stored, and FALSE otherwise.

Write **program code** for the method AddQuestion().

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(c) The first quiz is created with the identifier FirstQuiz.

The first question in this quiz is: “What is 100 / 5 ?”.

The answer is “20” and the difficulty level is 1.

Write **program code** to:

- declare an instance of QuizClass with the identifier FirstQuiz
- declare an instance of QuestionClass with the identifier Question1
- add Question1 to the array in FirstQuiz using AddQuestion().

Programming language ....

Program code

....

....

....

....

....

....

....

....

.... [5]

(d) The object `FirstQuiz` contains objects of type `QuestionClass`.

State the name of this OOP feature.

....

.... [1]

(e) Ejaz can use an interpreter and a compiler to translate program code during the development process. The program will be distributed without any access to the source code.

(i) State when Ejaz should use an interpreter and a compiler. Each answer must be different.

Interpreter ....

....

Compiler ....

....
[2]

(ii) Give the name of **two** facilities that Ejaz can use to debug his program.

1 ....

....

2 ....

....
[2]

(iii) Describe **one** feature of an editor that Ejaz can use when writing the program.

....

....

....

.... [2]

4 Zara is writing a program to simulate a circular queue.

The queue, `MyNumbers`, has 10 elements. `Enqueue()` takes a parameter value and stores it at the tail of the queue. `Dequeue()` returns the item at the head of the queue.

The current state of the circular queue is:

Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Data			31	45	89	500	23	2		

`HeadIndex: 2`

`TailIndex: 8`

(a) Show the state of the queue, `HeadIndex` and `TailIndex` after the following operations:

```
Enqueue(23)

Enqueue(100)

Dequeue()

Dequeue()

Enqueue(50)
```

Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Data										

`HeadIndex:` ....

`TailIndex:` ....

[3]

(b) The global array, `MyNumbers`, is used to store the positive integer numbers for the queue.

The following global variables are used:

- `HeadIndex` stores the index of the first element in the queue
- `TailIndex` stores the index of the next free space in the queue
- `NumberInQueue` stores the number of items in the queue.

(i) The function `Enqueue()` takes the value to be added to the queue as a parameter. The function returns `TRUE` if the item was added, or `FALSE` if the queue is full.

Complete the **pseudocode** for the function `Enqueue()`.

```
FUNCTION Enqueue(BYVALUE DataToInsert : INTEGER) RETURNS BOOLEAN

  IF NumberInQueue > ..........................................

    THEN

     RETURN FALSE

    ELSE

     MyNumbers[..........................................] ← ..........................................

     TailIndex ← ..........................................

     IF TailIndex > 9

       THEN

        TailIndex ← ..........................................

     ENDIF

     NumberInQueue ← NumberInQueue + 1

     RETURN TRUE

  ENDIF

ENDFUNCTION
```

[5]

(ii) The function `Dequeue()` returns the value at the head of the queue, or −1 if the queue is empty.

Complete the **pseudocode** for the function `Dequeue()`.

```
FUNCTION Dequeue() RETURNS INTEGER
```

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

```
ENDFUNCTION
```

[5]

5 The following procedure performs an insertion sort on the global array `TheArray` that has 10 elements.

Complete the pseudocode for the procedure `InsertionSort()`.

```
PROCEDURE InsertionSort()

  DECLARE Count : INTEGER

  DECLARE Counter : INTEGER

  DECLARE Temp : INTEGER

  Count ← ..........................................

  WHILE Count < 10

    Temp ← TheArray[Count]

    Counter ← Count .............................................

    WHILE ......................... >= 0 AND TheArray[Counter] > ...................................

    TheArray[Counter + 1] ← TheArray[Counter]

    Counter ← Counter - 1

    ENDWHILE

    TheArray[............................................] ← Temp

    Count ← Count + 1

  ENDWHILE

ENDPROCEDURE
```

[5]

6 A social networking website only allows people who are over 16 years old to join.

To create an account, the user must enter:

- their age
- a unique username, which is compared to others in the database
- a password that must be at least 8 characters long, with at least one upper case letter, one lower case letter, one symbol and one digit.

If the user is not old enough to join the network, the statement "Too young" is displayed.
If the user is old enough, but the username is already taken, the statement "Choose another username" is displayed.
If the user is old enough, but the password does not meet the requirements, the statement "Password does not meet requirements" is displayed.

(a) Complete the decision table for the social networking website.

Conditions	Available username	N	Y	N	Y	N	Y	N	Y
	Suitable password	N	N	Y	Y	N	N	Y	Y
	Age > 16	N	N	N	N	Y	Y	Y	Y
Actions	"Too young"								
	"Choose another username"								
	"Password does not meet requirements"								

[4]

(b) Simplify the decision table by removing the redundancies.

Conditions	Available username								
	Suitable password								
	Age > 16								
Actions	"Too young"								
	"Choose another username"								
	"Password does not meet requirements"								

[3]

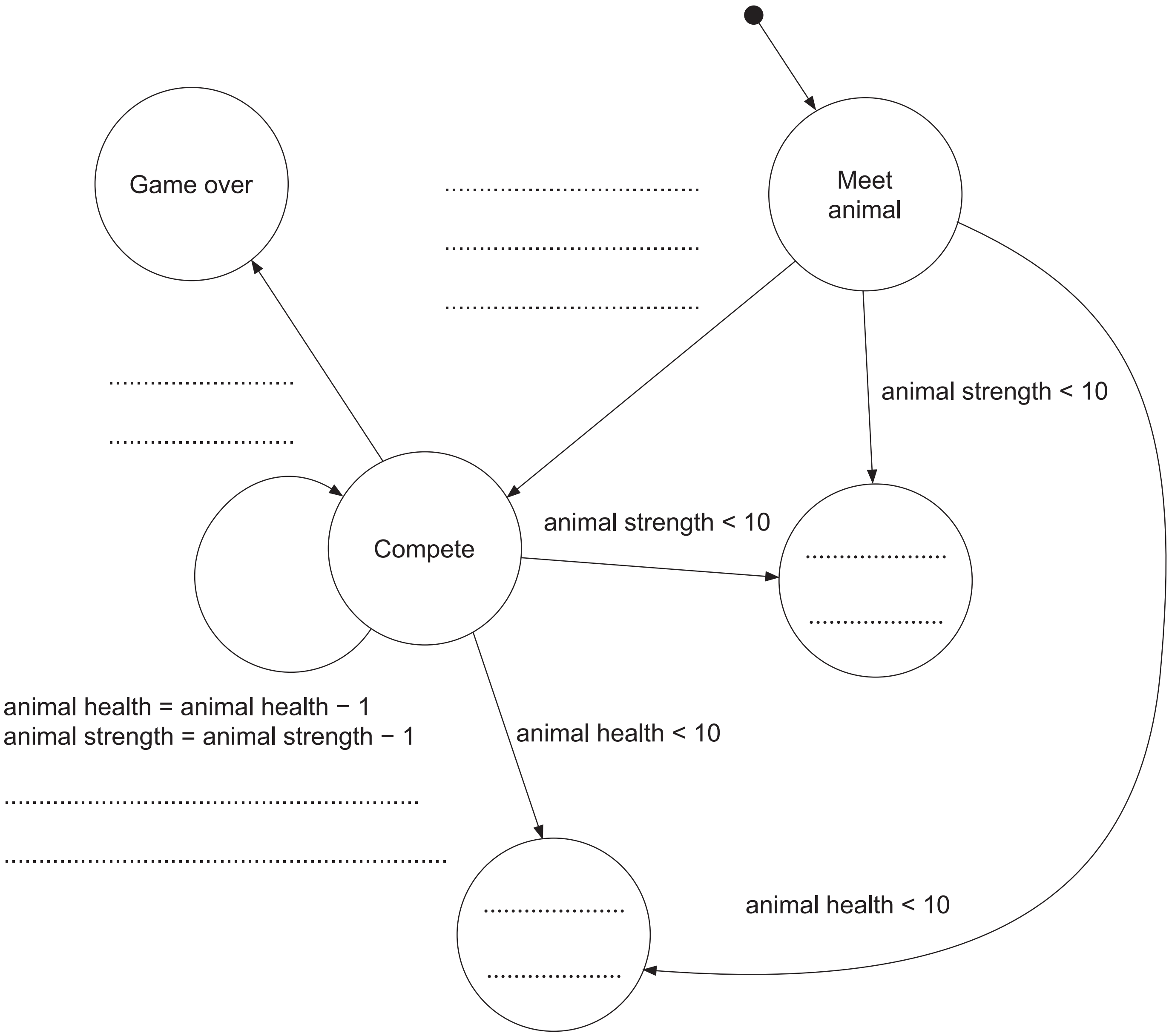
7 Anika is designing a computer game. The user controls a character that moves around a virtual world.
When the character meets an animal:

- if the animal's strength is less than 10, the animal runs away
- if the animal's health is less than 10, it is caught by the character
- if the animal's strength and health are both 10 or more, the character and the animal compete.

When the character and animal compete, the animal's health, animal's strength and character's health are decreased by 1. This is repeated until one of the following conditions is met:

- the character's health goes to 0, the game is over
- the animal's strength goes below 10, the animal runs away
- the animal's health goes below 10, the animal is caught.

Complete the state-transition diagram for this part of the program.



[5]

8 The table shows assembly language instructions for a processor that has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

	Instruction		**Explanation**
Label	**Op code**	**Operand**	
	`LDM`	`#n`	Immediate addressing. Load the number n to ACC
	`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC
	`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents of this calculated address to ACC
	`LDR`	`#n`	Immediate addressing. Load the number n to IX
	`STO`	`<address>`	Store the contents of ACC at the given address
	`ADD`	`<address>`	Add the contents of the given address to ACC
	`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX)
	`CMP`	`#n`	Compare the contents of ACC with number n
	`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False
	`LSL`	`#n`	Bits in ACC are shifted n places to the left. Zeroes are introduced on the right hand end
	`LSR`	`#n`	Bits in ACC are shifted n places to the right. Zeroes are introduced on the left hand end
	`OUT`		Output to the screen the character whose ASCII value is stored in ACC
	`END`		Return control to the operating system
`<label>:`	`<op code>`	`<operand>`	Labels an instruction
`<label>:`	`<data>`		Gives a symbolic address <label> to the memory location with contents <data>

An algorithm stores a 3-character word. It takes each character in turn, multiplies its value by 2 and outputs the new character.

Complete the following assembly language program for the algorithm using the instruction set provided on the previous page.

Instruction			Comment
Label	**Op code**	**Operand**	
			// initialise Index Register to 0
	LDX	character	// load character and multiply by 2
	OUT		// output the new character
	INC	IX	// increment the Index Register
	LDD	count	// loop 3 times
	STO	count	
	CMP	#3	
		LOOP	
	END		// end program
count:	0		
character:	B01000001		// the 3-character stored word
	B10001110		
	B01000100		

[5]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6258787983

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **October/November 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (SLM) 220617

1 **(a)** Numbers are stored in a computer using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and exponent.

(i) Write the normalised floating-point representation of the following unsigned binary number using this system.

1011100.011001

Working ....

....

....

....

....

Mantissa **Exponent**

[2]

(ii) State the consequence of storing the binary number in **part (a)(i)** as a floating-point number in this system. Justify your answer.

Consequence ....

....

Justification ....

....

....

[2]

(b) Explain the reason why binary numbers are stored in normalised form.

....

....

....

....

....

....

.... [3]

2 Draw **one** line from each programming paradigm to its **most appropriate** description.

Programming paradigm	**Description**
	Programs using the instruction set of a processor
Declarative	Programs based on events such as user actions or sensor outputs
Imperative	Programs using the concepts of class, inheritance, encapsulation and polymorphism
Low-level	Programs with an explicit sequence of commands that update the program state, with or without procedure calls
Object-oriented	Programs that specify the desired result rather than how to get to it

[4]

3 Enumerated and pointer are two non-composite data types.

(a) Write **pseudocode** to create an enumerated type called `Parts` to include these parts sold in a computer shop:

`Monitor, CPU, SSD, HDD, LaserPrinter, Keyboard, Mouse`

....

....

.... [2]

(b) Write **pseudocode** to create a pointer type called `SelectParts` that will reference the memory location in which the current part name is stored.

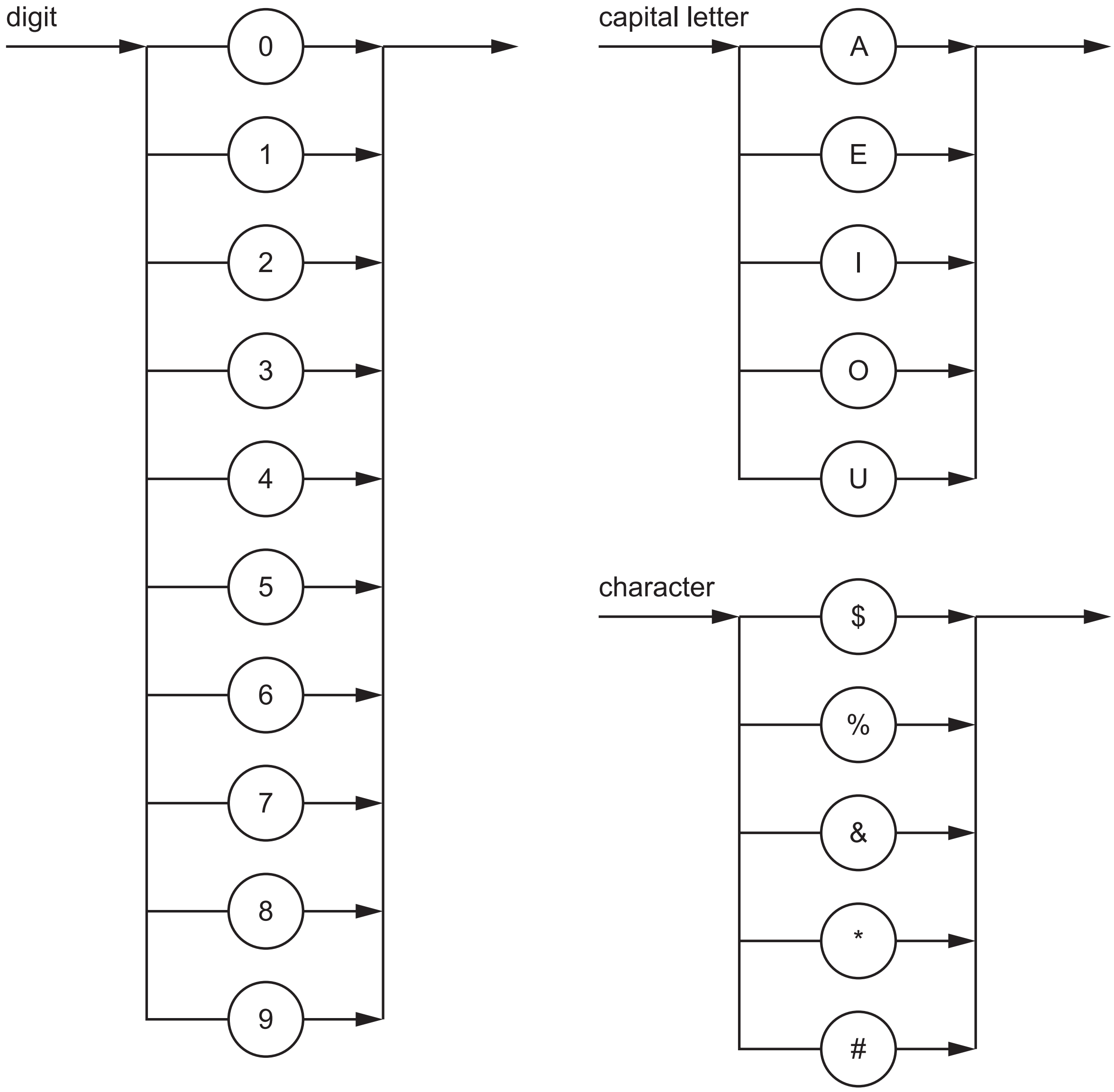
....

....

.... [2]

4 The following syntax diagrams for a particular programming language show the syntax of:

- a digit
- a capital letter
- a character.



(a) Write the Backus-Naur Form (BNF) notation of the syntax diagram for character.

....

....

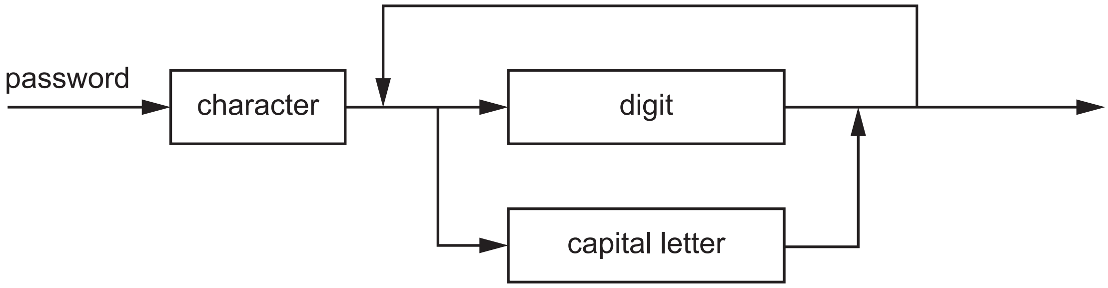
.... [2]

(b) A password must begin with a character and be followed by one or more digits or capital letters.

(i) State an example of a valid password.

.... [1]

(ii) A valid password is represented by the syntax diagram:



Write the BNF notation of the syntax diagram for password.

....

....

....

....

....

....

....

.... [4]

5 **(a)** Compare sequential and serial methods of file organisation.

....

....

....

....

....

....

....

....

.... [4]

(b) State the most suitable method of file access when a record is referenced by a unique address on a disk-type storage medium.

.... [1]

(c) State the most suitable method of file access when a bank stores its data records in ascending order of account number.

.... [1]

6 (a) Explain how packet switching is used to transfer messages across the internet.

....

....

....

....

....

....

....

....

....

.... [5]

(b) Outline the function of a router in packet switching.

....

....

....

....

....

.... [3]

7 **(a)** Write the Boolean expression that corresponds to the given truth table as a sum-of-products.

INPUT				**OUTPUT**
A	**B**	**C**	**D**	**Z**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Z = ...

.... [3]

(b) **(i)** Complete the Karnaugh map (K-map) for the given truth table.

CD \ AB	00	01	11	10
00				
01				
11				
10				

[2]

(ii) Draw loop(s) around appropriate group(s) of 1s in the K-map to produce an optimal sum-of-products. [2]

(iii) Write the Boolean expression from your answer to **part b(ii)** as a simplified sum-of-products.

Z = ...

.... [2]

(iv) Write the simplified Boolean expression for your answer to **part b(iii)**.

Z = ...

.... [1]

8 **(a)** Describe the purpose of the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.

....

....

....

....

.... [2]

(b) Explain how SSL/TLS protocols are used when a client-server communication is initiated.

....

....

....

....

....

....

....

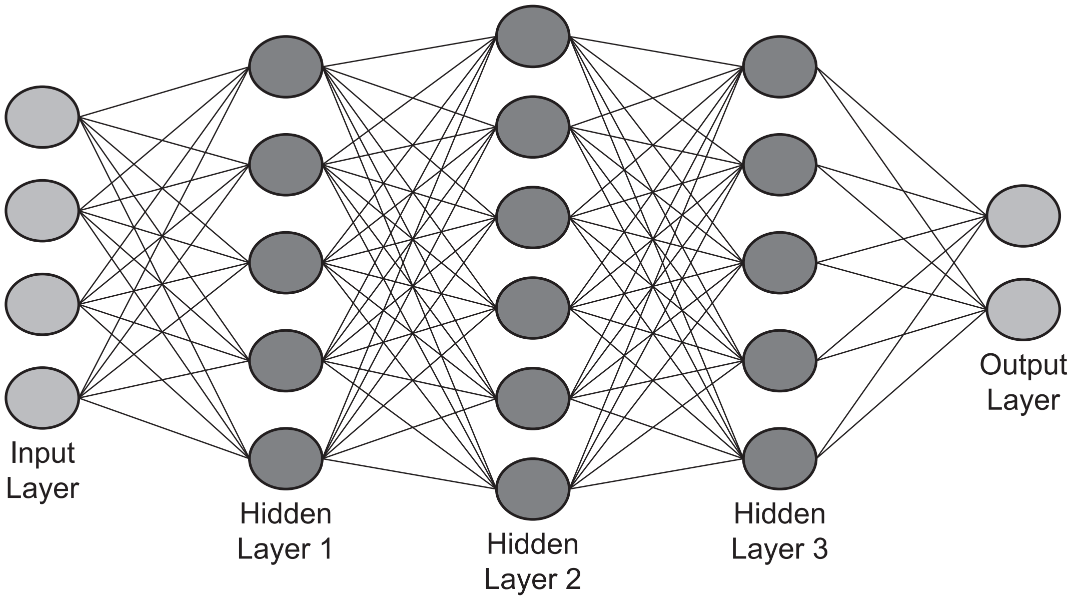
....

....

....

.... [4]

9 **(a)** The diagram shown represents an artificial neural network.



(i) State the reason for having multiple hidden layers in an artificial neural network.

....

.... [1]

(ii) Explain how artificial neural networks enable machine learning.

....

....

....

....

....

....

....

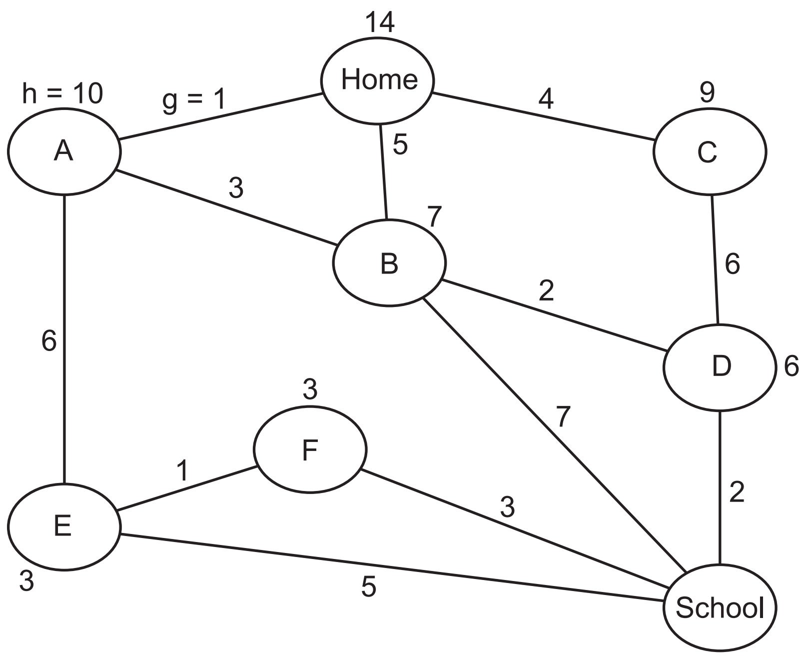
....

....

.... [4]

(b) Find the shortest path between the Home and School nodes using the A* algorithm. Show your working in the table provided.

The first two rows in the table have been completed.



Node	Cost from Home node (g)	Heuristic (h)	Total (f = g + h)
Home	0	14	14
A	1	10	11

Final path	

[5]

10 **(a)** State **three** essential features of **recursion**.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) Explain the reasons why a stack is a suitable Abstract Data Type (ADT) to implement recursion.

....

....

....

....

....

.... [3]

(c) Identify **two** ADTs other than a stack.

1 ....

2 ....
[2]

(d) The function `StackFull()` checks whether a stack is full.

The function uses the variable `TopOfStack` to represent the pointer to the most recent position used on the stack, and the variable `Max` to represent the maximum size of the stack. Assume `TopOfStack` and `Max` are global variables.

```
FUNCTION StackFull() RETURNS BOOLEAN
    IF TopOfStack = Max THEN
       RETURN TRUE
    ELSE
       RETURN FALSE
    ENDIF
ENDFUNCTION
```

An algorithm `AddInteger` is required to add a new integer data element to a stack.

The stack is implemented as an array `ArrayStack`.

The function `AddInteger()` calls `StackFull()` and returns an appropriate message.

Complete the pseudocode for the function `AddInteger()`.

```
FUNCTION AddInteger(NewInteger : INTEGER) RETURNS STRING
```

....

....

....

....

....

....

....

....

....

....

....

....

```
ENDFUNCTION
```

[5]

BLANK PAGE

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME						
CENTRE NUMBER					CANDIDATE NUMBER	

* 0 4 7 4 7 0 1 1 7 5 *

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2017**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **14** printed pages and **2** blank pages.

DC (ST/SW) 158024/4

1 **(a)** Consider the following pseudocode user-defined data type:

```
TYPE MyContactDetail
   DECLARE Name        : STRING
   DECLARE HouseNumber : INTEGER
ENDTYPE
```

(i) Write a pseudocode statement to declare a variable, `NewFriend`, of type `MyContactDetail`.

....[1]

(ii) Write a pseudocode statement that assigns 129 to the `HouseNumber` of `NewFriend`.

....[1]

(b) The user-defined data type `MyContactDetail` needs to be modified by:

- adding a field called `Area` which can take three values, uptown, downtown or midtown
- amending the field `HouseNumber` so that house numbers can only be in the range 1 to 499.

Write the updated version of `MyContactDetail`.

....

....

....

....

....

....

....[3]

(c) A pointer is a variable that stores the address of a variable of a particular type.

Consider the pseudocode on page 3, which uses the following identifiers:

Identifier	Data type	Description
`IPointer`	`^INTEGER`	pointer to an integer
`Sum`	`INTEGER`	an integer variable
`MyInt1`	`INTEGER`	an integer variable
`MyInt2`	`INTEGER`	an integer variable

```
Sum ← 91               // assigns the value 91 to the integer variable Sum
IPointer ← @Sum        // assigns to IPointer the address of the
                       // integer variable Sum
MyInt1 ← IPointer^     // assigns to variable MyInt1 the value at an
                       // address pointed at by IPointer
IPointer^ ← MyInt2     // assigns the value in the variable MyInt2 to
                       // the memory location pointed at by IPointer
```

The four assignment statements are executed. The diagram shows the memory contents after execution.

Variable	**Memory Address**	**Contents**
	...	
	5848	
	5847	
IPointer	5846	4402
	5845	
	...	
	4403	
Sum	4402	33
	4401	
	...	
	3428	
MyInt1	3427	91
MyInt2	3426	33
	3425	
	...	

Use the diagram to state the current values of the following expressions:

(i) `IPointer` ....[1]

(ii) `IPointer^` ....[1]

(iii) `@MyInt1` ....[1]

(iv) `IPointer^ = MyInt2` ....[1]

(d) Write pseudocode statements that will achieve the following:

(i) Place the address of `MyInt2` in `IPointer`.

....[1]

(ii) Assign the value `33` to the variable `MyInt1`.

....[1]

(iii) Copy the value in `MyInt2` into the memory location currently pointed at by `IPointer`.

....[1]

2 The following incomplete table shows descriptions and terms relating to malware.

(a) Complete the table with appropriate description and terms.

	Description	Term	
(i)	Malicious code is installed on a personal computer so that the user is misdirected to a fraudulent web site without their knowledge.	....	[1]
(ii)	An attempt to acquire sensitive information, often for malicious reasons, by trying to deceive the user through the contents of an email.	....	[1]
(iii)	....	Worm	[2]

(b) State **two** vulnerabilities that the malware in **part (a)(i)** or **part (a)(ii)** can exploit.

Vulnerability 1 ....

....

Vulnerability 2 ....

....
[2]

(c) Digital certificates are used in internet communications. A Certificate Authority (CA) is responsible for issuing a digital certificate.

The digital certificate contains a digital signature produced by the CA.

(i) Name **three** additional data items present in a digital certificate.

1 ....

2 ....

3 ....
[3]

(ii) Describe how the digital signature is produced by the CA.

....

....

....

....

....

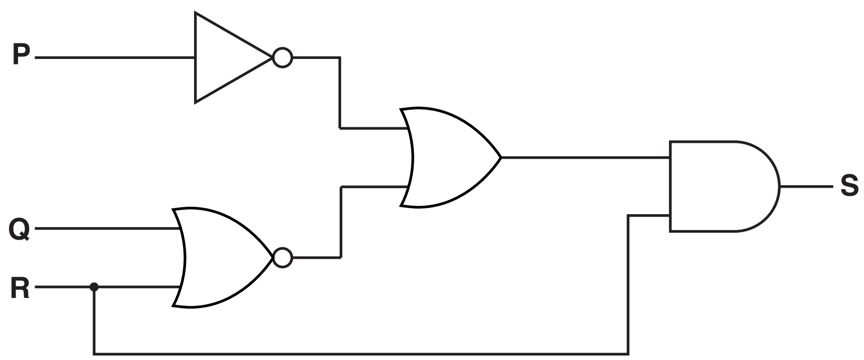
....[3]

(iii) Give the reason for including a digital signature in the digital certificate.

....

....[1]

3 A logic circuit is shown:



(a) Write the Boolean algebraic expression corresponding to this logic circuit:

S = ...[4]

(b) Complete the truth table for this logic circuit:

P	Q	R	Working space	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

[2]

(c) **(i)** Complete the Karnaugh Map (K-map) for the truth table in **part (b)**.

		PQ			
		00	**01**	**11**	**10**
R	**0**				
	1				

[1]

The K-map can be used to simplify the function in **part (a)**.

(ii) Draw loop(s) around appropriate groups to produce an optimal sum-of-products. [1]

(iii) Write a simplified sum-of-products expression, using your answer to **part (ii)**.

S = ...[1]

(d) One Boolean identity is:

(A + B) . C = A . C + B . C

Simplify the expression for S in **part (a)** to the expression for S in **part (c)(iii)**.

You should use the given identity and De Morgan's Laws.

....

....

....

....

....

....

....

....[3]

4 **(a)** Three file organisation methods and two file access methods are shown below.

Draw lines to link each file organisation method to its appropriate file access method(s).

File organisation method	**File access method**
random	sequential
serial	direct
sequential	

[4]

(b) An energy company supplies electricity to a large number of customers. Each customer has a meter that records the amount of electricity used. Customers submit meter readings using their online account.

The company's computer system stores data about its customers.

This data includes:

- account number
- personal data (name, address, telephone number)
- meter readings
- username and encrypted password.

The computer system uses three files:

File	Content	Use
A	Account number and meter readings for the current month.	Each time a customer submits their reading, a new record is added to the file.
B	Customer's personal data.	At the end of the month to create a statement that shows the electricity supplied and the total cost.
C	Usernames and encrypted passwords.	When customers log in to their accounts to submit meter readings.

For each of the files A, B and C, state an appropriate file organisation method for the use given in the table.

All three file organisation methods must be different.

Justify your choice.

(i) File A organisation ....

Justification ....

....

....

....[3]

(ii) File B organisation ....

Justification ....

....

....

....[3]

(iii) File C organisation ....

Justification ....

....

....

....[3]

5 The TCP/IP protocol suite can be viewed as a stack with four layers.

(a) Complete the stack by inserting the names of the three missing layers.

Application layer

[3]

(b) BitTorrent is a protocol used at the Application layer for the exchange of data.

(i) State the network model used with this protocol.

....[1]

(ii) State the use of BitTorrent.

....[1]

(iii) Explain how the exchange of data is achieved using BitTorrent.

....

....

....

....

....

....

....

....[4]

(c) State **two** additional protocols that are also used at the Application layer for the exchange of data.

For each protocol, give an example of an appropriate exchange of data.

Protocol 1 ....

Example ....

....

Protocol 2 ....

Example ....

....
[4]

6 A large office building has many floors. On each floor there are security sensors and security cameras. There is the same number of sensors on each floor. The building has a single security room.

The images from the security cameras are output on monitors (one monitor for each floor) placed in the security room.

The data from the sensors are read and processed by a computer system. Sensor readings and warning messages can be displayed on the monitors.

(a) **(i)** State the name given to the type of system described.

....[1]

(ii) Explain your answer to **part (i)**.

....

....[1]

(iii) State **two** sensors that could be used in this system.

Sensor 1 ....

Sensor 2 ....
[2]

(b) A software routine:

- checks the readings from the sensors
- outputs readings and warning messages to the monitors
- loops continuously.

The routine uses the following pseudocode variables:

Identifier	Data type	Description
`FloorCounter`	`INTEGER`	Loop counter for number of floors
`SensorCounter`	`INTEGER`	Loop counter for number of sensors
`NumberOfFloors`	`INTEGER`	Stores the number of floors
`NumberOfSensors`	`INTEGER`	Stores the number of sensors
`ForEver`	`BOOLEAN`	Stores value that ensures continuous loop

(i) Complete the following pseudocode algorithm for the routine.

```
01 ForEver ← ..............................................................................................
02 REPEAT
03   FOR FloorCounter ← 1 TO NumberOfFloors
04     FOR SensorCounter ← 1 TO ...................................................................
05       READ Sensor(SensorCounter)on Floor(FloorCounter)
06       IF Sensor value outside range
07          THEN
08             OUTPUT "Problem on Floor ", FloorCounter
09       ENDIF
10     ENDFOR
11   ENDFOR
12   //
13   // Delay loop
14   // Delay loop
15   //
16 UNTIL ..........................................................................................................
```

[3]

(ii) A delay needs to be introduced before the loop is processed again.

Write a `FOR` loop, in pseudocode, to replace lines `13` and `14`.

....

....[1]

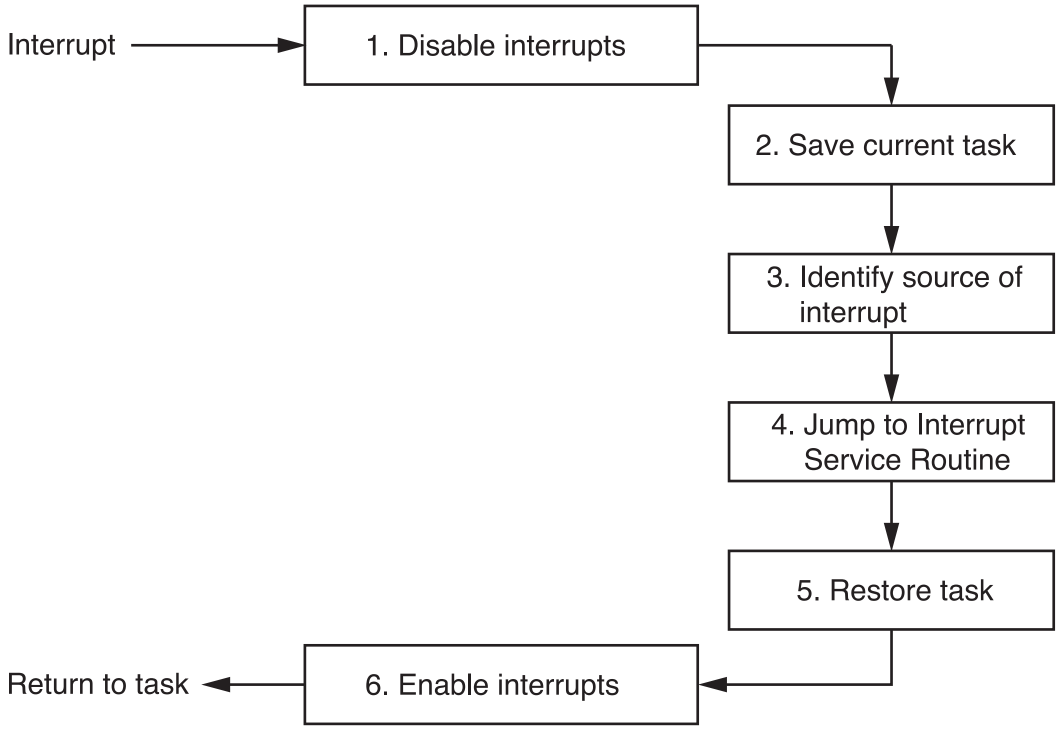
(iii) Give a reason for this delay in the system.

....

....[1]

(c) An alternative method of reading and processing sensor data is to use interrupts. Each sensor is connected so that it can send an interrupt signal to the processor if its value changes.

On receipt of an interrupt signal, the processor carries out a number of steps as shown in the following diagram.



(i) State the purpose of step 3.

....

....[1]

(ii) Explain what happens at step 4.

....

....

....

....[2]

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

Cambridge International AS & A Level

CANDIDATE NAME	
CENTRE NUMBER	
CANDIDATE NUMBER	

2326710888

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2019**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

DC (ST/CB) 171622/2

Cambridge Assessment International Education

1 Real numbers are stored using floating-point representation in a computer system.

This representation uses:

- 8 bits for the mantissa, followed by
- 4 bits for the exponent.

Two's complement form is used for both the mantissa and the exponent.

(a) **(i)** A real number is stored as a 12-bit normalised binary number as follows:

Mantissa

0	1	0	1	0	0	1	0

Exponent

0	0	1	0

Calculate the denary value for this binary number. Show your working.

Working ....

....

....

Denary value .... [3]

(ii) Calculate the normalised binary number for –3.75. Show your working.

Mantissa

Exponent

Working ....

....

....

.... [3]

(b) The number of bits available to represent a real number is increased to 16.

State the effect of increasing the size of the exponent by 4 bits.

....

.... [1]

(c) State why some binary representations can lead to rounding errors.

....

.... [1]

(d) Complete the following descriptions by inserting the **two** missing terms.

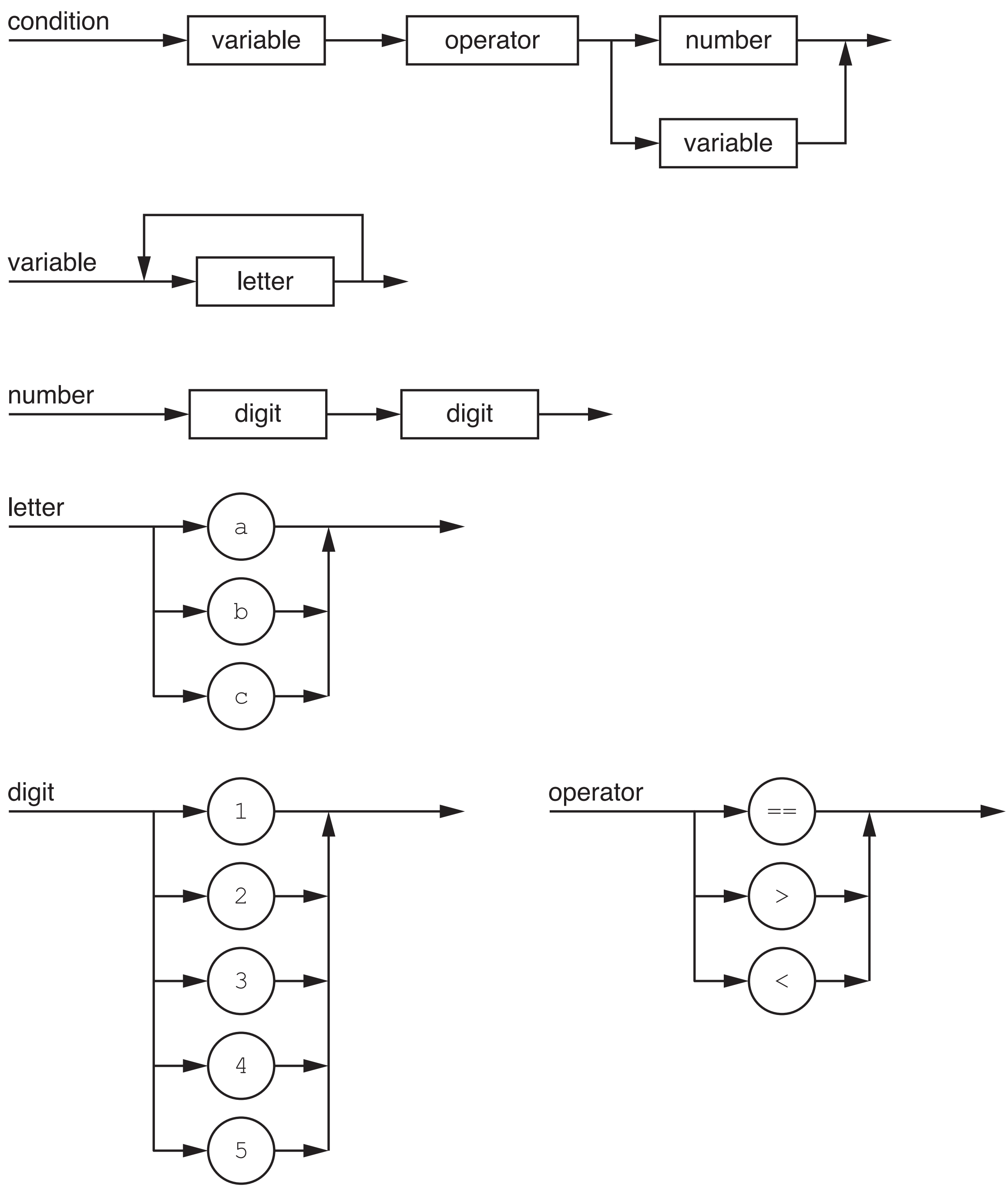
.... can occur in the exponent of a floating-point number, when the exponent has become too large to be represented using the number of bits available.

A calculation results in a number so small that it cannot be represented by the number of bits available. This is called ....

[2]

2 The following syntax diagrams for a programming language show the syntax of:

- a condition
- a variable
- a number
- a letter
- a digit
- an operator



(a) The following conditions are invalid.

Give the reason in each case.

(i) `35 > 24`

Reason ....

.... [1]

(ii) `abc := cba`

Reason ....

.... [1]

(iii) `bc < 49`

Reason ....

.... [1]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagram.

`<operator> ::=` ....

....

`<number> ::=` ....

....

`<variable> ::=` ....

....

`<condition> ::=` ....

....

[6]

3 Protocols are essential for communication between computers.

(a) Explain why protocols are essential for communication between computers.

....

....

....

....

.... [2]

(b) A protocol used in bus networks is CSMA/CD.

Explain what is meant by **CSMA/CD**.

....

....

....

....

....

....

....

.... [4]

4 A Boolean expression produces the following truth table.

INPUT			OUTPUT
A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(a) Write the Boolean expression for the truth table as a sum-of-products.

X = ... [2]

(b) Complete the Karnaugh Map (K-map) for the truth table above.

AB

		00	**01**	**11**	**10**
C	**0**				
	1				

[1]

The K-map can be used to simplify the expression in **part (a)**.

(c) Draw loops around appropriate groups in the K-map in **part (b)** to produce an optimal sum-of-products. [2]

(d) Write, using your answer to **part (c)**, a simplified sum-of-products expression for the truth table.

X = ... [2]

5 **(a)** Explain why user-defined data types are necessary.

....

....

....

.... [2]

(b) An organisation stores data about its employees.

- Employee ID is a five-digit number, for example, 01234.
- Employee name is a string, for example, 'Kiri Moana'.
- Department is one of three values: Sales, Technical, Customer services.
- Salary is an integer value in the range 25 000 to 150 000.

(i) Complete the following **pseudocode** definition of a user-defined data type to store the employee data.

```
TYPE Employee
  DECLARE EmployeeID   : ...............................................
  DECLARE EmployeeName : STRING
  DECLARE Department   : ( .............................................
                         ...............................................)
  DECLARE Salary       : 25000..150000
..........................................................................
```

[4]

(ii) Write a **pseudocode** statement to declare a variable, `NewEmployee` of data type `Employee`.

....

.... [1]

(iii) Write a **pseudocode** statement that assigns `02244` to the `EmployeeID` of `NewEmployee`.

....

.... [1]

(iv) `Employee` is an example of a record that is a composite data type.

State **two** other composite data types.

1 ....

2 ....

[2]

6 (a) An operating system (OS) uses a memory management technique called paging.

Explain what is meant by the following terms.

Page ....

....

....

Page frame ....

....

....

Page table ....

....

....

[3]

(b) Explain why an operating system needs to use scheduling algorithms.

....

....

....

....

....

.... [3]

(c) State what is meant by an **interrupt**.

....

.... [1]

(d) For a computer system using multi-programming, the low-level scheduler decides which process will get next use of the processor.

One algorithm could be a round-robin, which means every process gets use of the processor in sequence for a fixed amount of time (time-slice).

For a round-robin algorithm, five processes are currently loaded and get the use of the processor in the sequence:

JOB21 – JOBSS – JOBPT – JOB32 – JOB42, then return to JOB21

Process JOB32 has just completed its time-slice.

The following paragraph describes what happens next. Complete the paragraph by inserting the missing processes.

Interrupt received from the low-level scheduler. Save all register contents for

....

Copy the saved registers for .... to the CPU.

The processor will now process ....

[3]

7 **(a)** Identify the **four** layers of the TCP/IP protocol suite.

1 ....

2 ....

3 ....

4 ....
[4]

(b) The TCP/IP protocol suite is responsible for transmitting data across the Internet using packet switching.

(i) Explain why packet switching is used when sending data across the Internet.

....

....

....

.... [2]

(ii) Each packet requires a header.

Describe the purpose of a packet header.

....

....

....

.... [2]

(iii) Identify **three** items that should be contained in a packet header.

Item 1 ....

....

Item 2 ....

....

Item 3 ....

....
[3]

8 Digital certificates are used in internet communications. A Certificate Authority (CA) is responsible for issuing a digital certificate.

(a) Identify **two** data items present in a digital certificate.

1 ....

2 ....
[2]

(b) The following paragraph describes how a digital signature is produced. Complete the paragraph by inserting an appropriate term in each space.

A .... algorithm is used to generate a message digest from the plain text message. The message digest is .... with the sender's ....

[3]

9 **(a)** The following incomplete table shows descriptions relating to computer architectures.

Complete the table by inserting the appropriate terms.

	Description	Term
A	• There are several processors. • Each processor executes different sets of instructions on one set of data at the same time.	....
B	• The processor has several ALUs. • Each ALU executes the same set of instructions on different sets of data at the same time.	....
C	• There is only one processor. • The processor executes one set of instructions on one set of data.	....
D	• There are several processors. • Each processor executes a different set of instructions. • Each processor operates on different sets of data.	....

[4]

(b) State **three** characteristics of massively parallel computers.

1 ....

....

2 ....

....

3 ....

....

[3]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6690157298

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2019**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **11** printed pages and **1** blank page.

DC (ST) 163542/3

Cambridge Assessment International Education

1 **(a)** A computer stores real numbers using floating-point representation. The floating-point numbers have:

- eight bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both stored in two's complement format.

(i) Calculate the denary value of the following floating-point number.

Show your working.

Mantissa

0	0	1	1	0	1	1	1

Exponent

0	1	0	1

Working ....

....

....

....

....

Answer ....

[3]

(ii) State why the floating-point number in **part (a)(i)** is **not** normalised.

....

.... [1]

(iii) Give the floating-point number in **part (a)(i)** in normalised two's complement format.

Mantissa

Exponent

[2]

(b) (i) Convert the denary number +11.625 into a normalised floating-point number.

Show your working.

Working ....

....

....

....

....

....

Mantissa							

Exponent			

[3]

(ii) Convert the denary number –11.625 into a normalised floating-point number.

Show your working.

Working ....

....

....

....

....

....

Mantissa							

Exponent			

[3]

(c) A student enters the following into an interpreter:

```
OUTPUT(0.2 * 0.4)
```

The student is surprised to see that the interpreter outputs the following:

```
0.08000000000000002
```

Explain why the interpreter outputs this value.

....

....

....

....

....

.... [3]

2 Packet switching can be used to transmit data across the Internet.

Packet switching is not always the most appropriate method of transferring data.

(a) Name an alternative method of transferring data across the Internet.

.... [1]

(b) Give an example of a situation where the method you identified in **part (a)** is more appropriate.

Justify your choice.

Example ....

....

Justification ....

....

....

....
[3]

3 **(a)** A Boolean algebraic expression produces the following truth table.

INPUT			OUTPUT
A	**B**	**C**	**X**
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

(i) Complete the Karnaugh Map (K-map) for the truth table.

		AB			
		00	**01**	**11**	**10**
C	**0**				
	1				

[1]

The K-map can be used to simplify the expression that produced the truth table in **part (a)**.

(ii) Draw loops around appropriate groups of 1s in the K-map to produce an optimal sum-of-products. [2]

(iii) Write the simplified sum-of-products Boolean expression for the truth table.

X = ... [2]

(b) A logic circuit with four inputs produces the following truth table.

INPUT				OUTPUT
A	**B**	**C**	**D**	**X**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

(i) Complete the K-map for the truth table.

AB

CD				

[4]

(ii) Draw loops around appropriate groups of 1s in the K-map to produce an optimal sum-of-products. [2]

(iii) Write the simplified sum-of-products Boolean algebraic expression for the truth table.

X = ... [2]

4 **(a)** Describe the main steps in the evaluation of a Reverse Polish Notation (RPN) expression using a stack.

....

....

....

....

....

....

....

.... [4]

(b) The infix expression `8 * (5 – 2) – 30 / (2 * 3)` converts to:

`8 5 2 – * 30 2 3 * / –`

in Reverse Polish Notation (RPN).

Show the changing contents of the stack as this RPN expression is evaluated.

[4]

5 Sanjeet is a member of the public, and he wants to send a private message to a government department.

(a) Explain how asymmetric encryption is used to ensure that the message remains private.

....

....

....

.... [2]

(b) When the government department replies to Sanjeet, it needs to send a verified message.

Explain how asymmetric encryption can be used to ensure that it is a verified message.

....

....

....

....

....

.... [2]

(c) The government's computer systems are vulnerable to malware.

(i) Describe **two** vulnerabilities that malware can exploit in computer systems.

1 ....

....

....

....

2 ....

....

....

....

[4]

(ii) Identify **one** method that can be used to restrict the effect of malware.

....

.... [1]

6 A company sells plant watering systems that automatically turn on water sprinklers when the soil becomes too dry.

The plant watering system has a processor and connecting cables.

Identify **two** other hardware devices that are required in this system. State the purpose of each device.

Device 1 ....

Purpose ....

....

Device 2 ....

Purpose ....

.... [4]

7 **(a)** RISC (Reduced Instruction Set Computing) and CISC (Complex Instruction Set Computing) are two types of processor.

Tick (✓) **one** box in each row to show if the statement applies to RISC or CISC processors.

Statement	RISC	CISC
Larger instruction set		
Variable length instructions		
Smaller number of instruction formats		
Pipelining is easier		
Microprogrammed control unit		
Multi-cycle instructions		

[3]

(b) In parallel processing, a computer can have multiple processors running in parallel.

(i) State the **four** basic computer architectures used in parallel processing.

1 ....

2 ....

3 ....

4 ....
[4]

(ii) Describe what is meant by a **massively parallel computer**.

....

....

....

....

....

.... [3]

8 (a) A computer process can be in one of three states.

Identify **and** describe **two** of these states.

State 1 ....

Description ....

....

....

State 2 ....

Description ....

....

....
[6]

(b) One of the main tasks of an operating system is resource management.

Describe how an operating system can maximise the use of resources.

Primary memory ....

....

....

....

....

....

Disk ....

....

....

....

....

....
[6]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME						
CENTRE NUMBER						
CANDIDATE NUMBER						

9196178439

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2020**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Blank pages are indicated.

DC (JC) 188582/1

1 In a particular computer system, real numbers are stored using floating-point representation, with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the denary value for the following floating-point number. Show your working.

Mantissa													Exponent			
0	1	0	1	0	0	0	0	0	0	0	0		0	1	1	0

Working ....

....

....

....

Denary value ....

[3]

(b) A new operating system has been installed that has changed the way the floating-point numbers are used. The order of the exponent and the mantissa are reversed.

(i) Calculate the new denary value for the following floating-point number that has the same bit pattern as the number in **part (a)**. Show your working.

Exponent					Mantissa											
0	1	0	1		0	0	0	0	0	0	0	0	0	1	1	0

Working ....

....

....

....

Denary value ....

[3]

(ii) Identify **two** problems that can occur due to the change in the representation of the floating-point number.

Problem 1 ....

....

Problem 2 ....

....
[2]

2 Data types can be classified as composite or non-composite.

(a) Draw **one** line from each data type to its correct classification.

Data type	Classification
Pointer	
Record	Composite
Set	
Class	Non-composite
Integer	

[2]

(b) A user-defined data type, `timeOfDay`, is declared using the following pseudocode.

```
TYPE timeOfDay = (morning, afternoon, evening, night)
```

(i) Identify the type of user-defined data type declared **and** state its classification.

Type ....

Classification ....
[2]

(ii) Write pseudocode to declare the variable `session` of type `timeOfDay`.
Assign the value `afternoon` to the variable `session`.

....

....

....

.... [2]

3 The use of protocols is essential for successful communication between computers.

(a) Define the term **communication protocol**.

....

....

.... [2]

(b) Identify **two** protocols that are used in the transfer of emails **and** state the purpose of each protocol.

Protocol 1 ....

Purpose ....

....

Protocol 2 ....

Purpose ....

....
[4]

(c) Manav and Miora want to have a video conversation over the Internet using a dedicated connection.

(i) Identify **and** describe the switching method used to implement this connection.

Method ....

Description ....

....

....

....

....
[3]

(ii) State **one** benefit and **one** drawback of the method you identified in **part (c)(i)**.

Benefit ....

....

Drawback ....

....
[2]

4 The following truth table represents a logic circuit with three inputs and two outputs.

INPUT			OUTPUT	
A	**B**	**C**	**X**	**Y**
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

(a) Write the Boolean expressions for the truth table as sum-of-products.

X = ...

Y = ...

[3]

(b) Complete the Karnaugh Maps (K-maps) for the truth table.

OUTPUT X

AB

		00	**01**	**11**	**10**
C	**0**				
	1				

OUTPUT Y

AB

		00	**01**	**11**	**10**
C	**0**				
	1				

[2]

(c) The K-maps can be used to simplify **one** of the expressions in **part (a)**.

(i) Draw loop(s) around appropriate group(s) of 1s to produce an optimal sum-of-products for the single output table that can be simplified in **part (b)**. [2]

(ii) Write the simplified sum-of-products expressions for this output from **part (c)(i)**.

.... [2]

5 Complete these statements about flip-flops.

A flip-flop is a ....

It has .... stable states.

A flip-flop is used for ....

There are different types of flip-flop, for example .... and

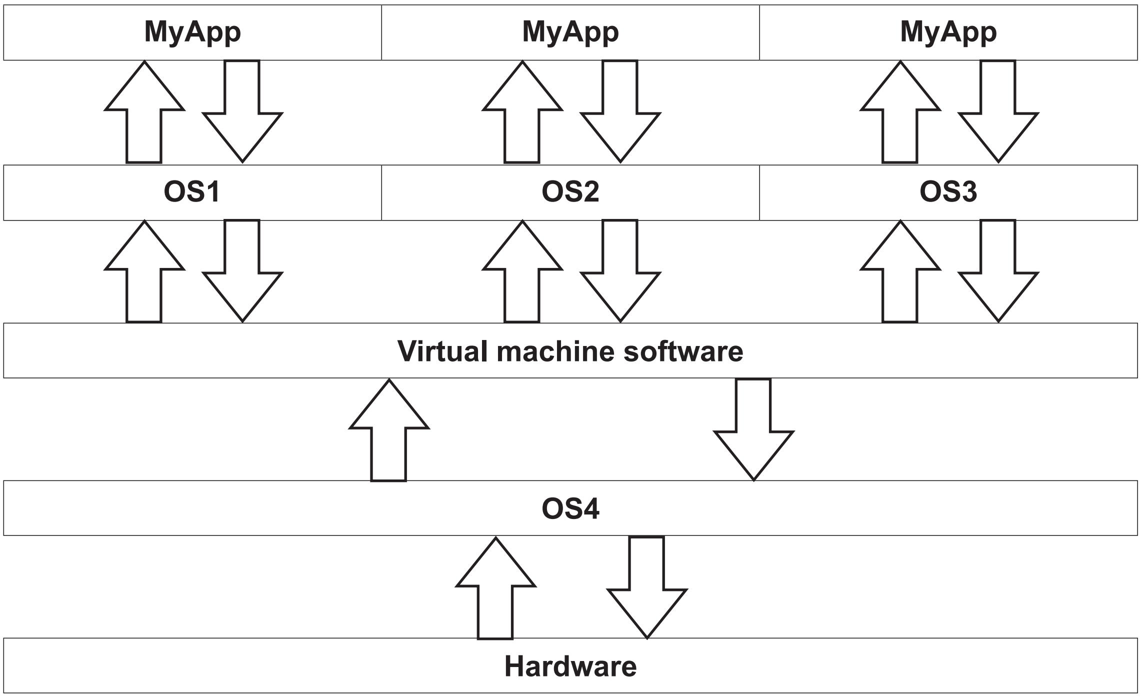
....

[5]

6 Mahmoud is developing a new application, MyApp, that needs to work with three different operating systems (OS1, OS2 and OS3).

He has decided to use virtual machine software to test MyApp with these three different operating systems.

The diagram shows MyApp running on three virtual machines.



(a) Describe the roles of the **four** operating systems (OS1, OS2, OS3 and OS4) shown in the diagram.

....

....

....

....

.... [3]

(b) Describe the role of the virtual machine software in the testing of MyApp.

....

....

....

....

....

.... [3]

(c) Explain **one** benefit and **one** drawback of this approach to testing MyApp.

Benefit ....

....

....

....

Drawback ....

....

....

....
[4]

7 Sam wants to send confidential data to an organisation. He has already received the organisation's digital certificate. The organisation has asked him to make sure that the message containing the confidential data is encrypted and is sent with a digital signature.

(a) Explain the process the organisation followed to obtain its digital certificate.

....

....

....

....

....

.... [3]

(b) Identify **two** items included in the organisation's digital certificate that will be used when sending the message. Give a reason why each item is required.

Item 1 ....

Reason ....

....

Item 2 ....

Reason ....

....
[4]

(c) Identify **two** other items included in the organisation's digital certificate.

....

....

....

.... [2]

(d) Explain how the digital signature for Sam's message is produced.

....

....

....

....

....

....

....

....

.... [4]

8 A car monitoring system provides information to the driver about the car's performance and alerts the driver to possible problems.

- Data about the car's performance is stored in three memory locations with addresses `601` to `603`.
- Location `601` contains the distance travelled in kilometres for the current trip as a binary integer.
- Location `602` contains the quantity of fuel used in litres for the current trip, as a fixed-point binary number with 5 places before the binary point and three places after the binary point.
- The four least significant bits of location `603` are flags used to identify problems with the car, for example, the fuel is low. A flag is set to 1 if there is a problem, or 0 if not. These problems are:
  - Bit 0 - high engine temperature
  - Bit 1 - low oil pressure
  - Bit 2 - low battery
  - Bit 3 - low fuel
  - Bits 4 to 7 are not used

(a) The current contents of addresses `601` to `603` are:

	Most significant							Least significant
601	0	0	1	0	1	1	0	0
602	0	0	1	0	1	0	0	1
603	0	0	0	0	0	1	0	0

State the information that the current contents of addresses `601` to `603` will provide to the driver.

....

....

....

.... [3]

(b) A car has low oil pressure and low fuel. It has travelled 80 kilometres and used 7.25 litres of fuel.

Complete the contents of the addresses to record this information.

601								
602								
603								

[3]

(c) The following table shows the assembly language instructions for the car performance monitoring system. There is one general purpose register, the Accumulator (ACC).

Table 8.1

Instruction			Explanation
Label	**Op code**	**Operand**	
	LDM	#n	Load the number n to ACC
	LDD	<address>	Load the contents of the location at the given address to ACC
	STO	<address>	Store the contents of ACC at the given address
	AND	#n	Bitwise AND operation of the contents of ACC with the numeric operand
	CMP	#n	Compare the contents of ACC with the number n
	JPE	<address>	Following a compare instruction, jump to <address> or <label> if the compare was True
	JMP	<address>	Jump to <address> or <label>
<label>:	<op code>	<operand>	Labels an instruction

Note:

denotes immediate addressing
B denotes a binary number, for example B01001010
& denotes a hexadecimal number, for example &4A

(i) Write **assembly language** instructions to set the contents of addresses 601 and 602 to zero, and set all four least significant bits of the contents of address 603 to one.
Use the instruction set from **Table 8.1**.

....

....

....

....

....

.... [3]

(ii) A program continuously checks the flags. If a flag is set, the program moves to the error-handling routine at the specified label. For example, if the engine temperature is high, the program jumps to the label for the error-handling routine `HIGHTEMP`. The error-handling routine instructions have not been provided.

A programmer has written most of the instructions for the program in the following table. There are four missing operands.

Complete the assembly language program by writing the **four** missing operands.

Label	Op code	Operand
CHECKFLAGS:	LDD	603
	AND	&0F
	STO	TEMP
	AND	&01
	CMP	&01
	JPE	HIGHTEMP
	LDD	TEMP
	AND	&02
	CMP	
	JPE	LOWOIL
	LDD	TEMP
	AND	
	CMP	&04
	JPE	LOWBATT
	LDD	
	AND	&08
	CMP	&08
	JPE	LOWFUEL
	JMP	
TEMP:		

[4]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME										
CENTRE NUMBER						CANDIDATE NUMBER				

2258633649

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 2018**

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (SC) 148677/4

CAMBRIDGE
International Examinations

[Turn over

1 A declarative language is used to represent the following facts and rules about animals.

```
01 feature(dog, drinks_milk).

02 feature(dog, has_lungs).

03 feature(horse, has_lungs).

04 feature(tuna, lives_in_water).

05 feature(tuna, has_gills).

06 feature(crab, lives_in_water).

07 mammal(drinks_milk).

08 mammal(has_lungs).

09 fish(lives_in_water).

10 fish(has_gills).

11 is_a_mammal(X) IF (feature(X, Y) AND mammal(Y)) AND (feature(X, Z)
   AND mammal(Z)).
```

These clauses are explained in the following table.

Clause	Explanation
`01`	A dog has the feature, drinks milk
`07`	A mammal drinks milk
`11`	`X` is a mammal, if: • `X` has the feature `Y` and a mammal has a feature `Y`, **and** • `X` has the feature `Z` and a mammal has the feature `Z`

(a) More facts are to be included.

(i) A bird has wings, and a bird lays eggs.

Write the additional clauses to record these facts.

`12` ....

`13` ....
[2]

(ii) An eagle has all the features of a bird.

Write the additional clauses to record this fact.

`14` ....

`15` ....
[2]

(b) **(i)** Using the variable `B`, the goal

```
feature(B, drinks_milk)
```

returns

```
B = dog
```

Write the result returned by the goal

```
feature(B, lives_in_water)
```

`B =` .... [2]

(ii) Write a goal, using the variable `C`, to find the feature(s) of tuna.

.... [2]

(c) An animal is a bird if it lays eggs **and** it has wings.

Complete the following rule.

`is_a_bird(X) IF` ....

.... [3]

(d) Declarative programming and object-oriented programming are two examples of programming paradigms.

(i) Define the term **programming paradigm**.

....

.... [1]

(ii) Give **two** examples of programming paradigms, other than declarative and object-oriented programming.

1 ....

2 ....

[2]

2 Kendra collects books. She is writing a program to store and analyse information about her books.

Her program stores information about each book as a record. The following table shows the information that will be stored about each book.

Field name	Description
Title	The title of the book
Author	The first listed author of the book
ISBN	A 13-digit code that uniquely identifies the book, for example: "0081107546738"
Fiction	If the book is fiction (TRUE) or non-fiction (FALSE)
LastRead	The date when Kendra last read the book

(a) Write **pseudocode** to declare an Abstract Data Type (ADT) named Book, to store the information in the table.

....

....

....

....

....

....

....

....

.... [4]

(b) The records are stored in a random access file.

The function, `Hash()`, takes as a parameter the ISBN and returns the hash value.

The disk address of the record in the hash table is calculated as: ISBN modulus 2000 plus 1.

Write **program code** for the function `Hash()`.

Programming language ....

Program code ....

....

....

....

....

....

....

.... [4]

(c) The random access file, `MyBooks.dat`, stores the data about the books in the format:

```
<Title>
<Author>
<ISBN>
<Fiction>
<LastRead>
```

A procedure, `FindBook()`:

- prompts the user to input the ISBN of a book until the ISBN contains 13 numeric digits
- uses the function `Hash()` to calculate the disk address of the record
- reads the record for that book from `MyBooks.dat` into a variable of type `Book`
- outputs all the data about the book.

Use **pseudocode** to write the procedure `FindBook()`.

You can assume that the record exists at the disk address generated.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[8]

3 Joseph is taking a toy apart. Each time he removes an item from the toy, he writes the name of the item at the bottom of a paper list. When he rebuilds the toy, he puts the items back together working from the bottom of the list.

Joseph writes a computer program to create the list using a stack, `Parts`.

(a) Describe a stack structure.

....

.... [1]

(b) The stack is represented as an array in the program, the first element in the array is `[0]`.

The current contents of the stack, `Parts`, and its pointer, `StackPointer` are shown.

StackPointer | 5 |

	StackContents
0	"Screw 1"
1	"Screw 2"
2	"Back case"
3	"Screw 3"
4	"Engine outer"
5	
6	
7	

(i) Describe the purpose of the variable `StackPointer`.

....

.... [1]

(ii) The procedure `POP()` removes an item from the stack. The procedure `PUSH(<identifier>)` adds an item to the stack.

The current contents of the stack, `Parts`, and its pointer, `StackPointer` are shown.

StackPointer | 5 |

StackContents

0	"Screw 1"
1	"Screw 2"
2	"Back case"
3	"Screw 3"
4	"Engine outer"
5	
6	
7	

Use the table below to show the contents of the stack, `Parts`, and its pointer after the following code is run.

```
POP()
POP()
PUSH("Light 1")
PUSH("Light 2")
PUSH("Wheel 1")
POP()
POP()
```

StackPointer | |

StackContents

0	
1	
2	
3	
4	
5	
6	
7	

[2]

(c) A 1D array, `Parts`, is used to implement the stack. `Parts` is declared as:

```
DECLARE Parts : ARRAY[0 : 19] OF STRING
```

(i) The procedure `POP` outputs the last element that has been pushed onto the stack and replaces it with a `'*'`.

Complete the **pseudocode** for the procedure `POP`.

```
PROCEDURE POP

   IF .................................. = ..................................

      THEN

         OUTPUT "The stack is empty"

      ELSE

         StackPointer ← ............................................

         OUTPUT .....................................................................

         Parts[StackPointer] ← ..................................

   ENDIF

ENDPROCEDURE
```

[5]

(ii) The procedure `PUSH()` puts the parameter onto the stack.

Complete the **pseudocode** for the procedure `PUSH()`.

```
PROCEDURE PUSH(BYVALUE Value : String)

   IF StackPointer > .....................................

      THEN

         OUTPUT "Stack full"

      ELSE

         ............................................. ← .......................................

         StackPointer ← ...............................................................

   ENDIF

ENDPROCEDURE
```

[4]

4 The recursive algorithm for the `Calculate()` function is defined as follows:

```
01 FUNCTION Calculate(BYVALUE Number : INTEGER) RETURNS INTEGER
02     IF Number = 0
03        THEN
04           Calculate ← -10
05        ELSE
06           Calculate ← Number * Calculate(Number - 1)
07     ENDIF
08 ENDFUNCTION
```

(a) **(i)** State what is meant by a **recursive algorithm**.

....

.... [1]

(ii) State the line number in `Calculate()` where the recursive call takes place.

.... [1]

Question 4(b) begins on the next page.

(b) The function is called with `Calculate(3)`.

Dry run the function **and** complete the trace table below. State the final value returned. Show your working.

```
FUNCTION Calculate(BYVALUE Number : INTEGER) RETURNS INTEGER
    IF Number = 0
       THEN
          Calculate ← -10
       ELSE
          Calculate ← Number * Calculate(Number - 1)
    ENDIF
ENDFUNCTION
```

Working ....

....

....

....

Trace table:

Call number	**Function call**	**`Number = 0 ?`**	**Return value**

Final return value ....

[6]

(c) A recursive algorithm within a subroutine can be replaced with an iterative algorithm.

(i) Describe **one** problem that can occur when running a subroutine that has a recursive algorithm.

....

....

....

.... [2]

(ii) Rewrite the `Calculate()` function in **pseudocode**, using an **iterative algorithm**.

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

5 A game uses a set of cards. Each card has a number (between `0` and `9` inclusive) and a shape (`"square"`, `"triangle"` or `"circle"`).

The game is written using object-oriented programming.

The class, `Cards`, has the private properties:

- `Number`
- `Shape`

and the methods:

- `Constructor()`
- `GetNumber()`
- `GetShape()`

The purpose of each method in the class `Cards` is given in the following table.

Method	**Purpose**
`Constructor()`	Takes a number and a shape as parameters Checks that the number and the shape are valid and: • either assigns the parameters to `Number` and `Shape` • or reports an error.
`GetNumber()`	A public method that returns the number for that card.
`GetShape()`	A public method that returns the shape for that card.

(a) Explain why the properties are private.

....

....

....

....

.... [2]

(b) Write **program code** for the `Constructor()` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

.... [5]

(c) Write **program code** for the `GetNumber()` method.

Programming language ....

Program code

....

....

....

....

.... [2]

(d) A card, `OneS`, has the value `1` for `Number` and the value `"square"` for `Shape`.

Write **program code** to instantiate an instance of `Cards` for `OneS`.

Programming language ....

Program code

....

....

.... [2]

(e) The game has a function, `Compare()` that takes two cards as parameters and compares them.

If the cards are identical, the function outputs `"SNAP"` and returns `-1`. If they are not identical, and the card numbers are different, it returns the `Number` of the card with the higher value or the `Number` for the cards if they are the same.

Write **program code** for the `Compare()` function.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

4461388712

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 2019**

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 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.
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.

The maximum number of marks is 75.

This document consists of **17** printed pages and **3** blank pages.

Cambridge Assessment International Education

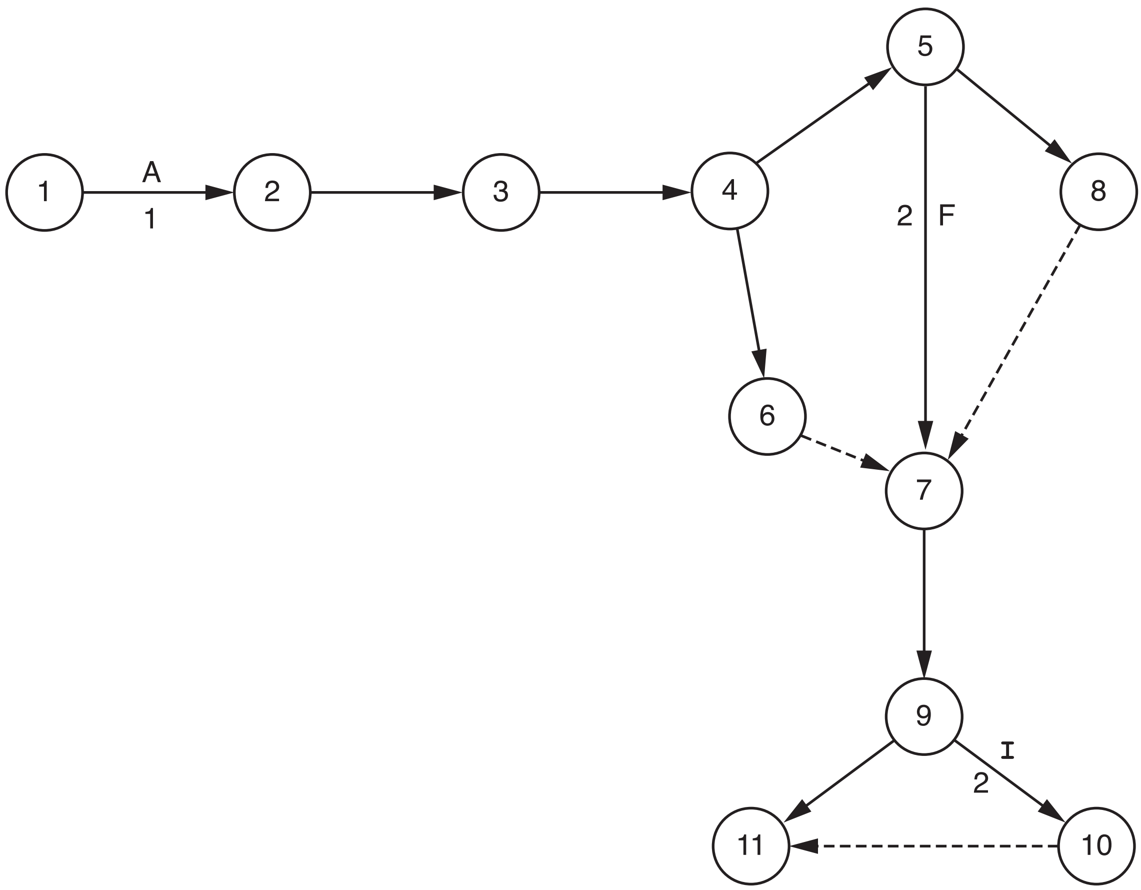
1 A technology company needs software to calculate how much each employee should be paid.

(a) Developing the software will involve the following activities:

Activity	Description	Time to complete (weeks)	Predecessor
A	Identify requirements	1	–
B	Observe current system	1	A
C	Create algorithm design	3	B
D	Write code	10	C
E	Test modules	7	C
F	White box testing	2	D
G	Black box testing	3	D
H	Install software	1	E, F, G
I	Acceptance testing	2	H
J	Create user documentation	2	H

(i) Add the correct activities and times to the following Program Evaluation Review Technique (PERT) chart for the software development.

Three of the activities and times have been done for you.



[7]

(ii) The dashed line connecting nodes 10 and 11 indicates a dummy activity.

State the purpose of a dummy activity.

....

.... [1]

(b) A bonus payment may be added to an employee's salary. A pension payment may also be subtracted from an employee's salary.

The company needs to assess what additions and subtractions should be made to the salary of each employee. There are three conditions to check:

- If the employee has worked a public holiday, they receive a 3% bonus payment.
- If the employee has worked 160 or more hours in a month, they receive an additional 5% bonus payment.
- If the employee pays into a pension, the company subtracts 4% for the pension payment.

Complete the decision table to show the additions and subtractions.

		Rules							
Conditions	**Public holiday**	Y	Y	Y	Y	N	N	N	N
	Hours >= 160	Y	Y	N	N	Y	Y	N	N
	Pension	Y	N	Y	N	Y	N	Y	N
Actions	**3% bonus payment**								
	5% bonus payment								
	4% pension payment								

[3]

(c) The company decides to implement a program for the software using object-oriented programming (OOP).

Each employee has a unique employee ID, name, address and date of birth. There are two types of employee: salary and apprenticeship.

Salaried employees are paid a fixed monthly payment. The hours a salary employee works in a month are recorded to calculate bonus payments. They may receive bonus payments and make pension payments (given in **part(b)**).

Apprenticeship employees are paid weekly. They receive an hourly rate of pay. Apprenticeship employees do not receive bonus payments or make pension payments.

(i) Complete the following class diagram for the program.

Employee
EmployeeID : STRING Name : STRING Address : STRING DateOfBirth : Date
Constructor() GetEmployeeID() GetName() GetAddress() GetDateOfBirth() SetEmployeeID() SetName() SetAddress() SetDateOfBirth()

SalaryEmployee
MonthlyPayment : CURRENCY HoursThisMonth: REAL PublicHoliday : BOOLEAN Pension : BOOLEAN
Constructor() GetMonthlyPayment() GetHoursThisMonth() GetPublicHoliday() GetPension() SetMonthlyPayment() SetHoursThisMonth() SetPublicHoliday() SetPension()

ApprenticeshipEmployee
....
.... GetHourlyRate() GetHoursThisWeek() SetHourlyRate()

[3]

(ii) Write **program code** for the `Constructor()` in the `Employee` class.

All properties are sent as parameters.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

.... [4]

(iii) Write **program code** for the `GetEmployeeID()` method in the `Employee` class.

The get method returns the value of the `EmployeeID` property.

Programming language ....

Program code

....

....

....

....

....

.... [2]

(iv) Write **program code** for the `SetEmployeeID()` method in the `Employee` class.

The set method takes the new value as its parameter.

Programming language ....

Program code

....

....

....

....

....

.... [2]

(v) Write **program code** for the `SetPension()` method in the `SalaryEmployee()` class.

- The method takes a new value for `Pension` as a parameter.
- If the parameter's value is valid (it is `TRUE` or `FALSE`), the method returns `TRUE` and sets the parameter's value.
- Otherwise the method returns `FALSE` and does not set `Pension`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

.... [4]

Question 1 continues on the next page.

(vi) A `SalaryEmployee` is paid a fixed monthly payment.

- If the employee has worked a public holiday, they receive a 3% bonus payment. This is calculated from their `MonthlyPayment`.
- If the employee has worked 160 or more hours in a month, they receive an additional 5% bonus payment, calculated from their `MonthlyPayment`.
- If the employee pays into a pension, 4% will be subtracted from their `MonthlyPayment`.

Monthly salary is the final payment the employee receives.

For example, Chris is a `SalaryEmployee`. His fixed `MonthlyPayment` is $1000. He has worked a public holiday. He has worked 165 hours this month. He pays into a pension.

- The public holiday bonus is $30 (3% of $1000)
- The hours worked bonus payment is $50 (5% of $1000)
- The pension payment is $40 (4% of $1000)

Chris's monthly salary is calculated as ($1000 + $30 + $50) – $40 = $1040

The function `CalculateMonthlySalary()` is used to calculate the monthly salary. It:

- takes a `SalaryEmployee` as a parameter
- calculates the bonus payments and pension payment
- outputs the pension payment and total bonus payment
- calculates and returns the monthly salary.

Write **program code** for the function `CalculateMonthlySalary()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

[8]

(d) Noona describes an example of a feature of object-oriented programming (OOP). She says:

"One method exists in the parent class but is overwritten in the child class, to behave differently."

Identify the feature Noona has described.

.... [1]

2 The number of cars that cross a bridge is recorded each hour. This number is placed in a circular queue before being processed.

(a) The queue is stored as an array, `NumberQueue`, with eight elements. The function `AddToQueue` adds a number to the queue. `EndPointer` and `StartPointer` are global variables.

Complete the following **pseudocode** algorithm for the function `AddToQueue`.

```
FUNCTION AddToQueue(Number : INTEGER) RETURNS BOOLEAN

   DECLARE TempPointer : INTEGER

   CONSTANT FirstIndex = 0

   CONSTANT LastIndex = ......................................................

   TempPointer ← EndPointer + 1

   IF ...................................................... > LastIndex

       THEN

          TempPointer ← ......................................................

   ENDIF

   IF TempPointer = StartPointer

       THEN

          RETURN ......................................................

       ELSE

          EndPointer ← TempPointer

          NumberQueue[EndPointer] ← ......................................................

          RETURN TRUE

   ENDIF

ENDFUNCTION
```

[5]

(b) Describe how a number is removed from the circular queue to be processed.

....

....

....

....

....

....

....

.... [4]

(c) A queue is one example of an Abstract Data Type (ADT).

Identify **three other** Abstract Data Types.

1 ....

2 ....

3 ....

[3]

3 A company wants to test a program to check that it works. They can use different types of test data to do this.

(a) Identify **three** different types of test data that the company can use.

1 ....

2 ....

3 ....
[3]

(b) The programmer will make use of debugging features, when building and testing a program.

(i) Two debugging features are described in the table.

Write the correct name for **each** debugging feature.

Description	**Name of debugging feature**
A point where the program can be halted to see if the program works to this point.	....
One statement is executed and then the program waits for input from the programmer to move on to the next statement.	....

[2]

(ii) Identify **and** describe **one other** debugging feature.

Debugging feature ....

Description ....

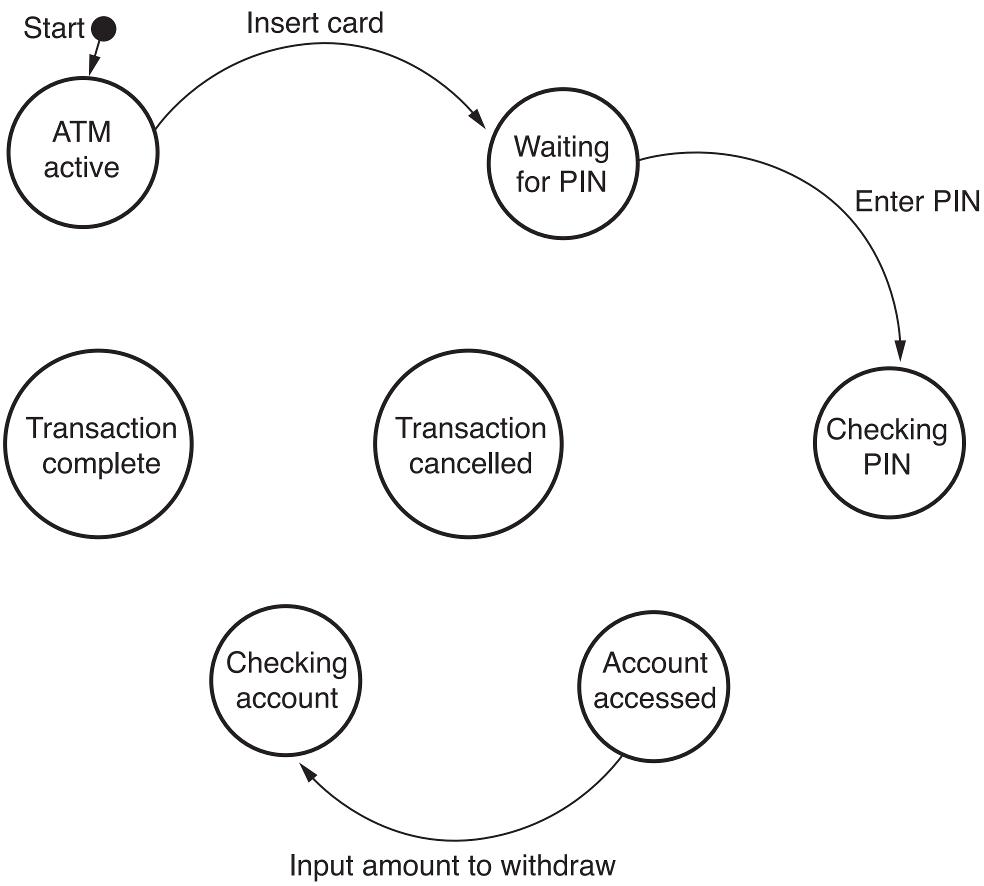
....
[2]

4 A bank wants to analyse how an automated teller machine (ATM) deals with transactions.

The following state-transition table shows the transitions from one state to another for a transaction.

Current state	Event	Next state
ATM active	Insert card	Waiting for PIN
Waiting for PIN	Enter PIN	Checking PIN
Waiting for PIN	Cancel selected	Transaction cancelled
Checking PIN	PIN valid	Account accessed
Checking PIN	PIN invalid	Waiting for PIN
Account accessed	Cancel selected	Transaction cancelled
Account accessed	Input amount to withdraw	Checking account
Checking account	Funds available	Transaction complete
Transaction complete	Return card and dispense cash	ATM active
Checking account	Funds not available	Account accessed
Transaction cancelled	Return card	ATM active

Complete the state-transition diagram to correspond with the table.



[8]

5 The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC) and an Index Register (IX).

Instruction		Explanation
Op code	**Operand**	
LDM	#n	Immediate addressing. Load the number n to ACC.
LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC.
LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents of this calculated address to ACC.
LDR	#n	Immediate addressing. Load the number n to IX.
STO	<address>	Store the contents of ACC at the given address.
STX	<address>	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents from ACC to this calculated address.
ADD	<address>	Add the contents of the given address to the ACC.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
JMP	<address>	Jump to the given address.
CMP	<address>	Compare the contents of ACC with the contents of <address>.
CMP	#n	Compare the contents of ACC with number n.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
AND	#n	Bitwise AND operation of the contents of ACC with the operand.
AND	<address>	Bitwise AND operation of the contents of ACC with the contents of <address>.
XOR	#n	Bitwise XOR operation of the contents of ACC with the operand.
XOR	<address>	Bitwise XOR operation of the contents of ACC with the contents of <address>.
OR	#n	Bitwise OR operation of the contents of ACC with the operand.
OR	<address>	Bitwise OR operation of the contents of ACC with the contents of <address>. <address> can be an absolute address or a symbolic address.
LSL	#n	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
LSR	#n	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
IN		Key in a character and store its ASCII value in ACC.
OUT		Output to the screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

(a) A programmer needs a program that multiplies a binary number by 4.

The programmer has started to write the program in the following table. The comment column contains explanations for the missing program instructions.

Write the program using the given instruction set.

Label	Instruction		Comment
	Op code	Operand	
			`// load contents of NUMBER`
			`// perform shift to multiply by 4`
			`// store contents of ACC in NUMBER`
			`// end program`
`NUMBER:`	`B00110110`		

[5]

Note:

`#` denotes immediate addressing
`B` denotes a binary number, e.g. `B01001010`
`&` denotes a hexadecimal number, e.g. `&4A`

(b) A programmer needs a program that counts the number of lower case letters in a string.

The programmer has started to write the program in the following table. The comment column contains explanations for the missing program instructions.

Complete the program using the given instruction set. A copy of the instruction set is provided on the opposite page.

Label	Instruction		Comment
	Op code	Operand	
	LDR	#0	// initialise Index Register to 0
START:			// load the next value from the STRING
			// perform bitwise AND operation with MASK
			// check if result is equal to MASK
			// if FALSE, jump to UPPER
			// increment COUNT
UPPER:	INC	IX	// increment the Index Register
			// decrement LENGTH
			// is LENGTH = 0 ?
			// if FALSE, jump to START
	END		// end program
MASK:	B00100000		// if bit 5 is 1, letter is lower case
COUNT:	0		
LENGTH:	5		
STRING:	B01001000		// ASCII code for 'H'
	B01100001		// ASCII code for 'a'
	B01110000		// ASCII code for 'p'
	B01110000		// ASCII code for 'p'
	B01011001		// ASCII code for 'Y'

[8]

Instruction		Explanation
Op code	Operand	
LDM	#n	Immediate addressing. Load the number n to ACC.
LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC.
LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents of this calculated address to ACC.
LDR	#n	Immediate addressing. Load the number n to IX.
STO	<address>	Store the contents of ACC at the given address.
STX	<address>	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents from ACC to this calculated address.
ADD	<address>	Add the contents of the given address to the ACC.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
JMP	<address>	Jump to the given address.
CMP	<address>	Compare the contents of ACC with the contents of <address>.
CMP	#n	Compare the contents of ACC with number n.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
AND	#n	Bitwise AND operation of the contents of ACC with the operand.
AND	<address>	Bitwise AND operation of the contents of ACC with the contents of <address>.
XOR	#n	Bitwise XOR operation of the contents of ACC with the operand.
XOR	<address>	Bitwise XOR operation of the contents of ACC with the contents of <address>.
OR	#n	Bitwise OR operation of the contents of ACC with the operand.
OR	<address>	Bitwise OR operation of the contents of ACC with the contents of <address>. <address> can be an absolute address or a symbolic address.
LSL	#n	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
LSR	#n	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
IN		Key in a character and store its ASCII value in ACC.
OUT		Output to the screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

2692392368

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **May/June 2023**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

DC (KS) 338128/2

1 Numbers are stored in two different computer systems by using floating-point representation.

System 1 uses:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

System 2 uses:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised floating-point representation of 113.75 **and** show how it would be represented in each of these two systems.

Show your working.

System 1

Mantissa

Exponent

System 2

Mantissa

Exponent

Working ....

....

....

....

....

....

[4]

(b) Explain the problem that occurred in **part (a)** when representing the number in **system 2**.

....

....

....

.... [2]

2 **(a)** Draw **one** line from each machine learning category to its **most appropriate** description.

Machine learning category	**Description**
Supervised learning	simulates the data-processing capabilities of the human brain to make decisions
Reinforcement learning	enables learning by mapping an input to an output based on example input–output pairs
Deep learning	enables information related to errors produced by the neural network to be transmitted
Unsupervised learning	enables learning in an interactive environment by trial and error using its own experiences
	enables learning by allowing the process to discover patterns on its own that were previously undetected

[4]

(b) Describe the purpose of both the A* algorithm and Dijkstra's algorithm.

....

....

....

.... [2]

3 **(a)** A hashing algorithm is used to calculate storage locations for records in a random access file. It calculates hash values by using the function modulus 3.

The function modulus gives the remainder after integer division.
For example, 1030 modulus 3 = 1. Therefore, the record key 1030 gives a hash value of 1.

Complete the table to show the remaining hash values.

Record key	**Hash value**
`1030`	`1`
`1050`	
`1025`	

[2]

(b) Describe what happens, in relation to the storage or retrieval of a record in the file, when the calculated hash value is a duplicate of a previously calculated hash value for a different record key.

....

....

....

....

....

....

....

.... [4]

4 Two descriptions of user-defined data types are given.

Give appropriate type declaration statements for each, including appropriate names.

(a) A data type to hold a set of prime numbers below 20. These prime numbers are:

2, 3, 5, 7, 11, 13, 17, 19

....

....

....

.... [2]

(b) A data type to point to a day in the week, for example Monday.

....

....

....

.... [2]

5 **(a)** State, with a reason, where it would be appropriate to use circuit switching.

....

....

....

.... [2]

(b) Give **two** benefits and **two** drawbacks of circuit switching.

Benefit 1 ....

....

Benefit 2 ....

....

Drawback 1 ....

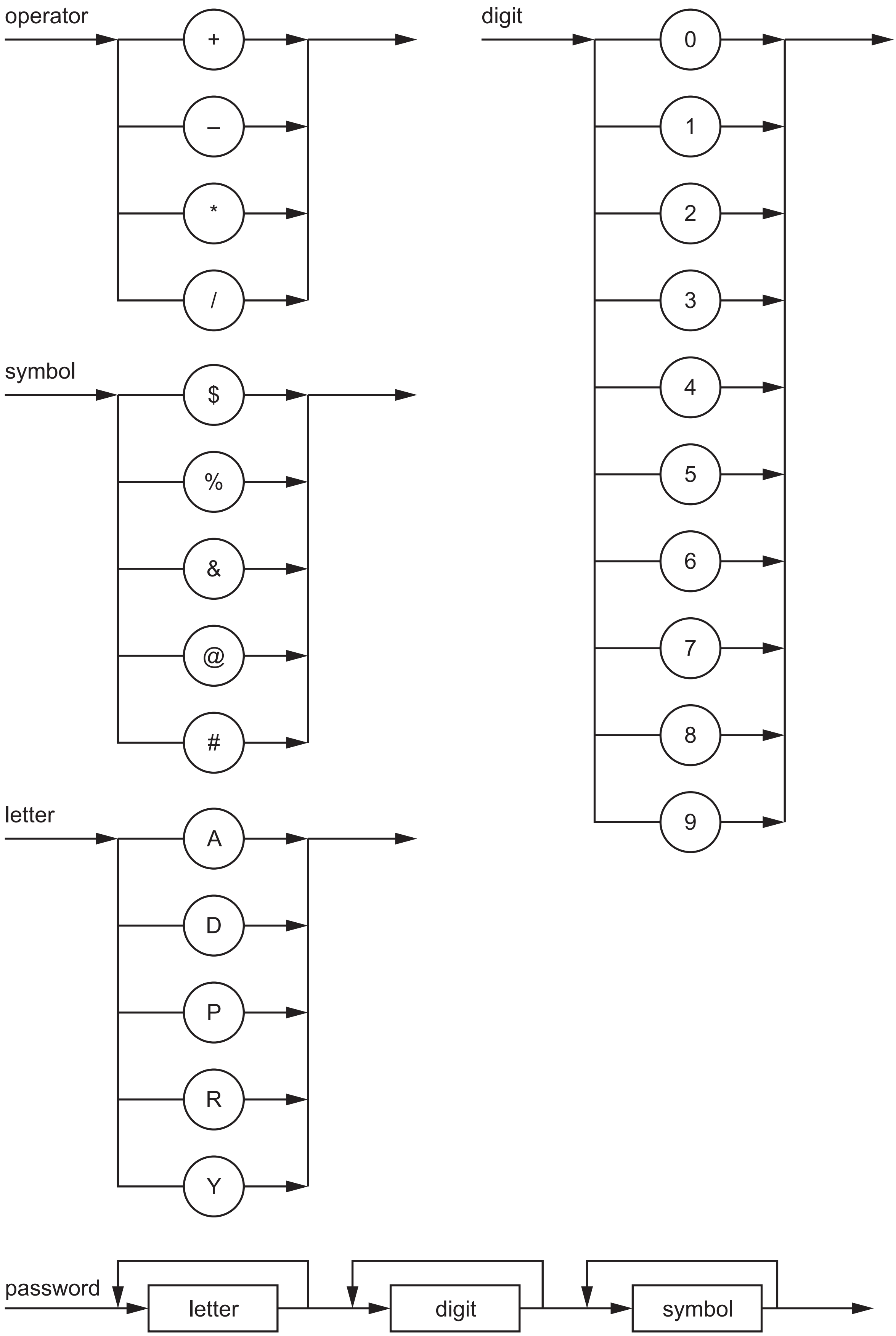
....

Drawback 2 ....

....

[4]

6 Several syntax diagrams are shown.



(a) State whether each of the following passwords is valid or invalid and give a reason for your choice.

`DPAD99$` ....

Reason ....

....

`DAD#95` ....

Reason ....

....

`ADY123?` ....

Reason ....

.... [3]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.

`<symbol> ::=` ....

....

`<letter> ::=` ....

.... [1]

(c) An identifier begins with one or more letters, followed by zero digits or one digit or more digits.

Valid letters and digits are shown in the syntax diagrams on page 6.

Draw a syntax diagram for an identifier.

[4]

7 **(a)** Complete the Karnaugh map (K-map) for the following Boolean expression.

$$\mathbf{Z = \overline{A}.\overline{B}.\overline{C}.\overline{D} + \overline{A}.\overline{B}.\overline{C}.D + \overline{A}.B.\overline{C}.\overline{D} + \overline{A}.B.\overline{C}.D + A.B.\overline{C}.\overline{D} + A.B.\overline{C}.D}$$

CD \ AB	00	01	11	10
00				
01				
11				
10				

[2]

(b) Draw loop(s) around appropriate group(s) in the K-map to produce an optimal sum-of-products. [2]

(c) Write the Boolean logic expression from your answer to **part (b)** as a simplified sum-of-products.

Z = ...

.... [2]

(d) Use Boolean algebra to give your answer to **part (c)** in its simplest form.

Z = ... [1]

8 Outline the characteristics of massively parallel computers.

....

....

....

....

....

.... [3]

9 **(a)** Encryption is used to alter data into a form that makes it meaningless if intercepted.

Describe the purpose of asymmetric key cryptography.

....

....

....

.... [2]

(b) Identify **two** benefits and **two** drawbacks of quantum cryptography.

Benefit 1 ....

....

Benefit 2 ....

....

Drawback 1 ....

....

Drawback 2 ....

....
[4]

10 The pseudocode algorithm shown copies an active accounts text file `ActiveFile.txt` to an archive accounts text file `ArchiveFile.txt`, one line at a time. Any blank lines found in the active accounts text file are replaced with the words "`Account not present`" in the archive accounts text file.

Complete this file-handling pseudocode.

```
DECLARE Account : STRING

..............................................................................

OPENFILE "ArchiveFile.txt" FOR WRITE

WHILE NOT ....................................................................
   READFILE "ActiveFile.txt", Account
   IF Account = "" THEN

      WRITEFILE "ArchiveFile.txt", " ......................................"
   ELSE

      WRITEFILE "ArchiveFile.txt", .........................................
   ENDIF
ENDWHILE

..............................................................................
CLOSEFILE "ArchiveFile.txt"
```

[5]

11 Pseudocode is to be written to implement a queue Abstract Data Type (ADT) with items of the string data type. This will be implemented using the information in the table.

Identifier	Data type	Description
`FrontPointer`	`INTEGER`	points to the start of the queue
`RearPointer`	`INTEGER`	points to the end of the queue
`Length`	`INTEGER`	the current size of the queue
`Queue`	`STRING`	1D array to implement the queue

A constant, with identifier `MaxSize`, limits the size of the queue to 60 items.

(a) Write the pseudocode to declare `MaxSize`, `FrontPointer`, `RearPointer`, `Length` and `Queue`.

....

....

....

....

....

.... [3]

(b) Complete the following pseudocode for the function `Dequeue` to remove the front item from the queue.

```
FUNCTION Dequeue RETURNS STRING
   DECLARE Item : STRING

   ............................................................ > 0 THEN

      Item ← ..........................................................
      FrontPointer ← FrontPointer + 1

      ..................................................................
      IF Length = 0 THEN
         CALL Initialise // reset the pointers
      ELSE
         IF FrontPointer > MaxSize THEN

         ........................................................... ← 1
         ENDIF
      ENDIF
   ELSE
      OUTPUT "The print queue was empty - error!"
      Item ← ""
   ENDIF
   RETURN Item
ENDFUNCTION
```

[4]

(c) Explain how a new element can be added to the queue if it is implemented using two stacks.

....

....

....

....

....

.... [4]

12 **(a)** Describe what is meant by recursion.

....

....

....

.... [2]

(b) A Fibonacci sequence is a series of numbers formed by adding together the two preceding numbers, for example:

0, 1, 1, 2, …

This function calculates and returns values in the Fibonacci sequence and uses recursion.

```
FUNCTION Fib(Number : INTEGER) RETURNS INTEGER
   IF Number <= 1 THEN
      Result ← Number
   ELSE
      Result ← Fib(Number – 1) + Fib(Number – 2)
   ENDIF
   RETURN Result
ENDFUNCTION
```

Complete the trace table for the function when it is called as `Fib(5)`.

Call number	Function call	Number	Result	Return value
		5		

[5]

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

4135444915

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 2017**

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (NH/JG) 129977/4

CAMBRIDGE International Examinations

[Turn over

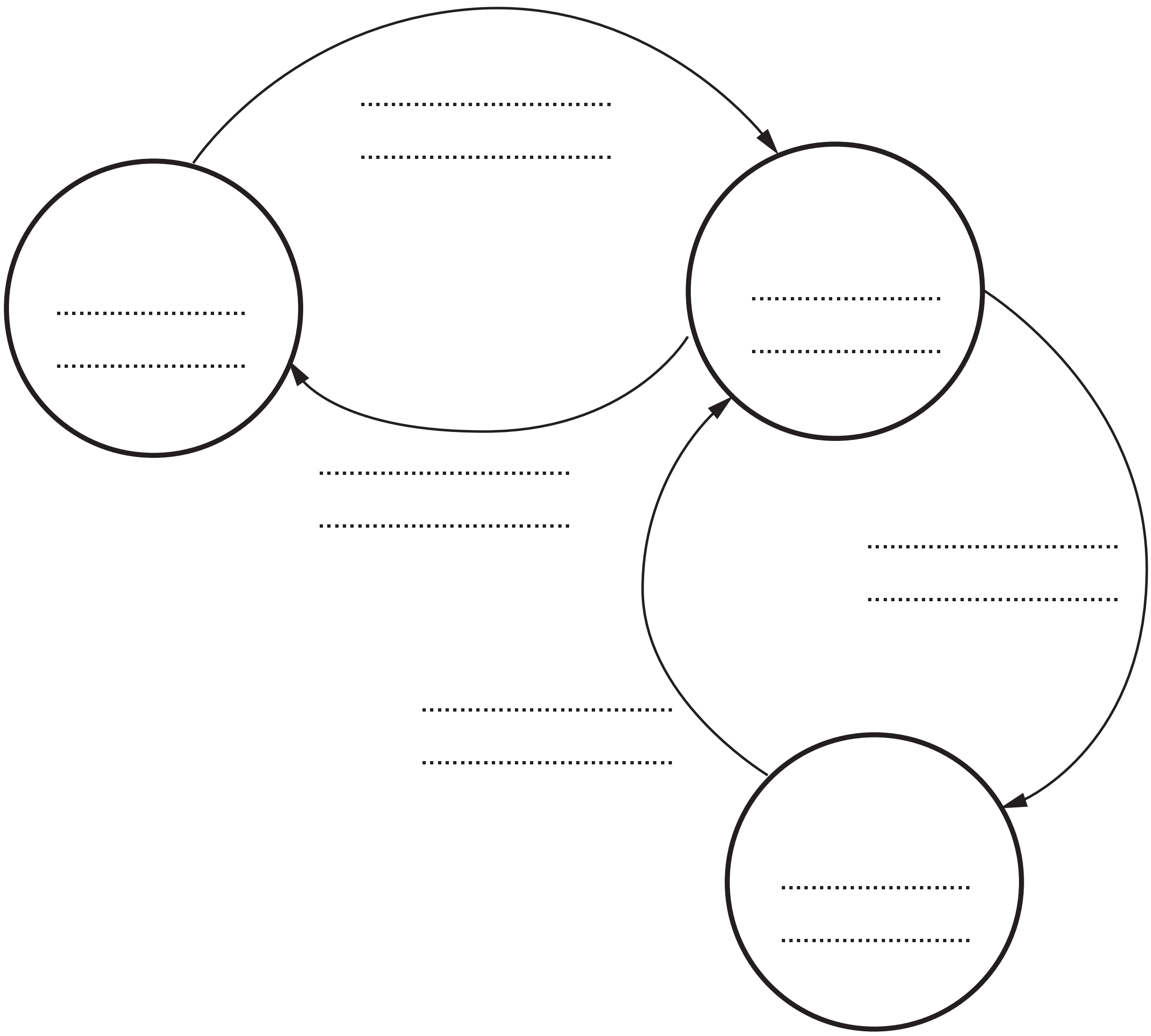
1 A greenhouse has a window that automatically opens and closes depending on the internal temperature.

If the temperature rises above 20 °C, the window half opens. If the temperature rises above 30 °C, the window fully opens. If the temperature drops below 25 °C, the window returns to being half open. If the temperature drops below 15 °C, the window fully closes.

The window has three possible states: **Closed**, **Half Open** and **Fully Open**.

Current state	Event	Next state
Closed	Temperature rises above 20 °C	Half Open
Half Open	Temperature drops below 15 °C	Closed
Half Open	Temperature rises above 30 °C	Fully Open
Fully Open	Temperature drops below 25 °C	Half Open

Complete the state-transition diagram for the window:



[7]

2 **(a)** **(i)** State how repetition is shown in a Jackson Structured Programming (JSP) structure diagram.

....

....[1]

(ii) State how selection is shown in a JSP structure diagram.

....

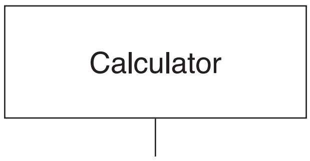
....[1]

(b) A simple calculator is to be created.

The calculator is to be used as follows:

- User inputs 2 numbers (x and y).
- User inputs an operator (`+, −, * or /`).
- The calculator computes the answer.
- The calculator displays the answer.

Draw a JSP diagram for the calculator. The first element is provided.



[5]

3 A declarative programming language is used to represent the following knowledge base:

```
01 person(jane).
02 person(ahmed).
03 person(caroline).
04 person(stuart).
05 food(chocolate).
06 food(sushi).
07 food(pizza).
08 food(chilli).
09 likes(jane, pizza).
10 likes(ahmed, chocolate).
11 likes(ahmed, pizza).
12 likes(jane, chilli).
13 likes(stuart, sushi).
14 dislikes(stuart, chocolate).
15 dislikes(jane, sushi).
16 dislikes(caroline, pizza).
```

These clauses have the following meanings:

Clause	Explanation
01	Jane is a person
05	Chocolate is a food
09	Jane likes pizza
14	Stuart dislikes (does not like) chocolate

(a) Mimi is a person who likes chocolate but does not like sushi or lettuce.

Write additional clauses to represent this information.

17 ....

18 ....

19 ....

20 ....

21 ....

[5]

(b) Using the variable `PersonName`, the goal:

```
likes(PersonName, pizza).
```

returns:

```
PersonName = jane, ahmed.
```

Write the result that is returned by the goal:

```
likes(ahmed, FoodItem).
```

`FoodItem =` ....

....[2]

(c) B might like A, if B is a person, A is a food and B does not dislike A.

Write this as a rule.

`might_like(`.. , ...`)`

`IF` ....

....

....[6]

4 The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		Explanation
Op code	**Operand**	
LDM	#n	Immediate addressing. Load the number n to ACC.
LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC.
LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
LDR	#n	Immediate addressing. Load the number n to IX.
STO	<address>	Store the contents of ACC at the given address.
STX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents from ACC to this calculated address.
ADD	<address>	Add the contents of the given address to the ACC.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
JMP	<address>	Jump to the given address.
CMP	<address>	Compare the contents of ACC with the contents of <address>.
CMP	#n	Compare the contents of ACC with number n.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
LSL	#n	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
LSR	#n	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
IN		Key in a character and store its ASCII value in ACC.
OUT		Output to the screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

(a) A program stores a letter. The user is allowed nine attempts to guess the stored letter. The program outputs "?" and the user guesses a letter. If the user guesses the letter, the program outputs "*".

The following is pseudocode for this program.

```
REPEAT
   OUTPUT '?'
   INPUT GUESS
   IF GUESS = LETTERTOGUESS
      THEN
        OUTPUT '*'
        BREAK
   ELSE
      ATTEMPTS ← ATTEMPTS + 1
   ENDIF
UNTIL ATTEMPTS = 9
```

Write this program. Use the op codes from the instruction set provided.

Label	Op code	Operand	Comment
START:	LDM	#63	// load ASCII value for '?'
			// OUTPUT '?'
			// input GUESS
			// compare with stored letter
			// if correct guess, go to GUESSED
			// increment ATTEMPTS
			// is ATTEMPTS = 9 ?
			// if out of guesses, go to ENDP
			// go back to beginning of loop
GUESSED:	LDM	#42	// load ASCII for '*'
			// OUTPUT '*'
ENDP:	END		// end program
ATTEMPTS:	0		
LETTERTOGUESS:	'a'		

[11]

(b) Five numbers are stored, starting in the location labelled NUMBERS. A program is needed to multiply each of the numbers by 4 and store them back in their original location.

Write this program. Use the op codes from the instruction set on the opposite page.

Label	Op code	Operand	Comment
START:			// initialise the Index Register
			// load the value from NUMBERS
			// multiply by 4
			// store the new value in NUMBERS
			// increment the Index Register
			// increment COUNT
			// is COUNT = 5 ?
			// repeat for next number
ENDP:	END		
COUNT:	0		
NUMBERS:	22		
	13		
	5		
	46		
	12		

[10]

Instruction		Explanation
Op code	**Operand**	
LDM	#n	Immediate addressing. Load the number n to ACC.
LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC.
LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
LDR	#n	Immediate addressing. Load the number n to IX.
STO	<address>	Store the contents of ACC at the given address.
STX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents from ACC to this calculated address.
ADD	<address>	Add the contents of the given address to the ACC.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
JMP	<address>	Jump to the given address.
CMP	<address>	Compare the contents of ACC with the contents of <address>.
CMP	#n	Compare the contents of ACC with number n.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
LSL	#n	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
LSR	#n	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
IN		Key in a character and store its ASCII value in ACC.
OUT		Output to the screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

5 Large development projects require careful resource management.

(a) **(i)** Name an appropriate project management tool that helps the manager to work out the estimated length of time it takes for the project to complete.

....

....[1]

(ii) Explain how, during the planning stage of the project, the manager would use the tool you named in **part (a)(i)**.

....

....

....

....

....

....[3]

(b) **(i)** Different programmers have been writing independent modules. The modules now need to be combined to create the final system.

Name the type of testing required at this stage.

....

....[1]

(ii) Name the final testing stage required before the system becomes operational.

....

....[1]

6 A programmer wants to create a computer simulation of animals searching for food in a desert. The desert is represented by a 40 by 40 grid. Each position in the grid is represented by a pair of coordinates. `'A'` represents an animal and `'F'` represents food. At the start of the simulation, the grid contains 5 animals and 1 food source.

The following is an example of part of the grid.

	0	1	2	3	4	...	37	38	39
0	A					..			
1			F			..			
2						..		A	
3				A		..			
...	..	..	..	..	..	..	..	..	..
38				A		..	A		
39						..			

A timer is used. In each time interval, each animal randomly moves 0 or 1 position in a random direction. The program generates this movement by computing two random numbers, each of which can be –1, 0 or 1. The program adds the first random number to the across number and the second random number to the down number representing the animal's position.

For example:

- if 0 and 1 are generated, the across value does not change, the down value increases by 1
- if –1 and 1 are generated, the across value decreases by 1, and the down value increases by 1.

Each animal has an individual score. If the animal moves to a position in the grid with food (`'F'`):

- the animal's score increases by 1
- the food disappears
- one new animal (`'A'`) is randomly generated and added to the grid (to a maximum of 20 animals)
- one new food (`'F'`) is randomly generated and added to the grid.

The simulation is to be implemented using object-oriented programming.

The programmer has designed two classes, `Desert` and `Animal`.

The `Desert` class consists of:

- attributes
  - `Grid`
  - `StepCounter`
  - `AnimalList`
  - `NumberOfAnimals`
- methods
  - `Constructor`
  - `IncrementStepCounter`
  - `GenerateFood`
  - `DisplayGrid`

Each attribute consists of a value and a get and set method that allow access to the attributes.

The following table describes the attributes and methods for the `Animal` class.

Identifier	Data type	Description
`Constructor()`		Instantiate an object of the `Animal` class • Generate a pair of random numbers between 0 and 39. • Place animal at that random position. • Initialise the animal's score to 0.
`EatFood()`		• Delete the food. • Increase the score of the animal that called the method. • Call the `GenerateFood` method of the `Desert` class. • Call the `Constructor` method of the `Animal` class.
`Move()`		• Call the `GenerateChangeInCoordinate` method for each coordinate (across or down number) of the animal's position. • Moves the animal to the new space. • If there is food in the new position, call the `EatFood` method.
`Score`	`INTEGER`	Initialised to 0
`Across`	`INTEGER`	The across value, between 0 and 39
`Down`	`INTEGER`	The down value, between 0 and 39

(a) Write **program code** to declare the attributes and constructor for the `Animal` class.

You only need to write the set and get methods for the attribute `Across`.

You should also write:

- the constructor for the class
- set and get methods for the `Across` attribute only.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

(b) The `Constructor` method of the `Desert` class:

- initialises an empty grid
- creates 5 animal objects which are added to the `AnimalList` (an array of animal objects currently on the grid)
- generates one food
- sets the `StepCounter` to 0.

Write **program code** for the `Constructor` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(c) (i) The function `GenerateChangeInCoordinate`:

- receives a coordinate (across or down number) as a parameter
- checks whether the coordinate's value is at a boundary of the grid
- returns a random change (–1, 0 or 1) that will keep the animal's position within the grid.

Write **program code** for the `GenerateChangeInCoordinate` function.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

(ii) The `Move` method uses the `GenerateChangeInCoordinate` function to calculate the new `Across` and `Down` values for an animal. If there is food in the new position in the grid, the animal eats the food.

Write **program code** for the `Move` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

(d) The programmer plans to add a graphic display to the program. The programmer will make use of a program library.

Explain what is meant by a program library.

....

....

....

....[2]

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6250551904

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **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 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.
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.

The maximum number of marks is 75.

This document consists of **19** printed pages and **1** blank page.

DC (LM/SG) 124303

CAMBRIDGE International Examinations

1 A linked list abstract data type (ADT) is to be used to store and organise surnames.

This will be implemented with a 1D array and a start pointer. Elements of the array consist of a user-defined type. The user-defined type consists of a data value and a link pointer.

Identifier	Data type	Description
`LinkedList`	`RECORD`	User-defined type
`Surname`	`STRING`	Surname string
`Ptr`	`INTEGER`	Link pointers for the linked list

(a) (i) Write **pseudocode** to declare the type `LinkedList`.

....

....

....

....[3]

(ii) The 1D array is implemented with an array `SurnameList` of type `LinkedList`.

Write the **pseudocode** declaration statement for `SurnameList`. The lower and upper bounds of the array are 1 and 5000 respectively.

....[2]

(b) The following surnames are organised as a linked list with a start pointer `StartPtr`.

`StartPtr: 3`

	1	2	3	4	5	6		5000
Surname	Liu	Yang	Chan	Wu	Zhao	Huang	...	
Ptr	4	5	6	2	0	1	...	

State the value of the following:

(i) `SurnameList[4].Surname` ....[1]

(ii) `SurnameList[StartPtr].Ptr` ....[1]

(c) Pseudocode is to be written to search the linked list for a surname input by the user.

Identifier	**Data type**	**Description**
`ThisSurname`	`STRING`	The surname to search for
`Current`	`INTEGER`	Index to array `SurnameList`
`StartPtr`	`INTEGER`	Index to array `SurnameList`. Points to the element at the start of the linked list

(i) Study the pseudocode in **part (c)(ii)**.

Complete the table above by adding the missing identifier details. [2]

(ii) Complete the pseudocode.

```
Current ← .......................................................................
IF Current = 0
   THEN
      OUTPUT ....................................................................
   ELSE
      IsFound ← .................................................................
      INPUT ThisSurname
      REPEAT
         IF ................................................... = ThisSurname
            THEN
               IsFound ← TRUE
               OUTPUT "Surname found at position ", Current
            ELSE
               // move to the next list item
               ..........................................................
         ENDIF
      UNTIL IsFound = TRUE OR ...............................................
      IF IsFound = FALSE
         THEN
            OUTPUT "Not Found"
      ENDIF
ENDIF
```

[6]

2 **(a)** **(i)** State what is meant by a recursively defined procedure.

....

....[1]

(ii) Write the line number from the pseudocode shown in **part (b)** that shows the procedure X is recursive. .... [1]

(b) The recursive procedure X is defined as follows:

```
01 PROCEDURE X(Index, Item)
02    IF MyList[Index] > 0
03       THEN
04          IF MyList(Index) >= Item
05             THEN
06                MyList[Index] ← MyList[Index + 1]
07          ENDIF
08          CALL X(Index + 1, Item)
09    ENDIF
10 ENDPROCEDURE
```

An array MyList is used to store a sorted data set of non-zero integers. Unused cells contain zero.

	1	2	3	4	5	6	7	8	9	10
MyList	3	5	8	9	13	16	27	0	0	0

(i) Complete the trace table for the dry-run of the pseudocode for the procedure `CALL X(1, 9)`.

		MyList									
Index	**Item**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	9	3	5	8	9	13	16	27	0	0	0

[4]

(ii) State the purpose of procedure `X` when used with the array `MyList`.

....

....[1]

3 A car hire company hires cars to customers. Each time a car is hired, this is treated as a transaction.

For each transaction, the following data are stored.

For the customer:

- customer name
- ID number

For the hire:

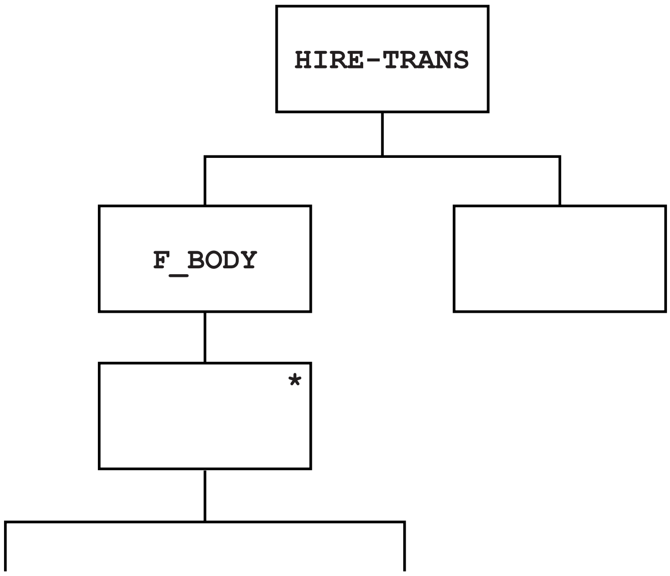
- car registration
- hire start date
- number of days hired

The transaction data are stored in a text file `HIRE-TRANS`. The file is made up of a file body, `F_BODY`, and a file trailer, `F_TRAILER`.

`F_BODY` has one transaction, `TRANS`, on each line.

(a) The first step in Jackson Structured Programming (JSP) design is to produce a JSP data structure diagram.

Complete the following JSP data structure diagram.



[7]

(b) The computer system will produce many printed reports.

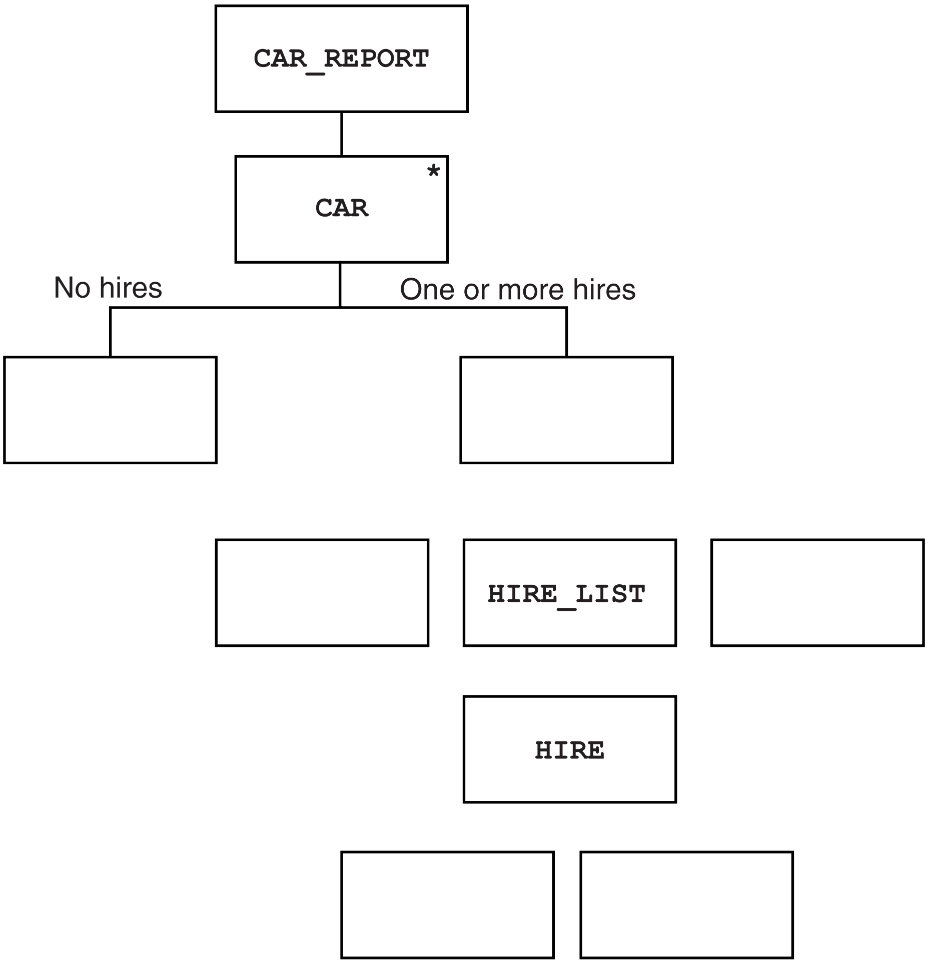
One report is `CAR_REPORT`. This displays all hire data for all cars.

For each car, the following data are displayed:

- the car data
- a list of all the hires
- the total number of hires

A car with zero hires is not included on the report.

Complete the following `CAR_REPORT` JSP data structure diagram.



[5]

4 When a car reaches a certain age, a safety assessment has to be carried out. A car's brakes and tyres must be tested. The tyre test result and the brakes test result for each car are recorded. If the car passes the assessment, a safety certificate is issued.

Cars have a unique three-character registration.

The following knowledge base is used:

```
01 car(a05).
02 car(h04).
03 car(a03).
04 car(h07).
05 car(a23).
06 car(p05).
07 car(b04).
08 carRegYear(a05, 2015).
09 carRegYear(h04, 2013).
10 carRegYear(a03, 2008).
11 carRegYear(h07, 2011).
12 carRegYear(a23, 2008).
13 carRegYear(p05, 2014).
14 carRegYear(b04, 2014).
15 testBrakes(h07, pass).
16 testTyres(h07, fail).
17 testBrakes(a03, fail).
18 testTyres(a03, fail).
19 testBrakes(a23, pass).
20 testTyres(a23, pass).
21 carAssessmentDue if carRegYear(Car, RegYear)
                                       and RegYear <= DeadlineYear.
22 issueCertificate(Car) if testTyres(Car, Result) and
                          testBrakes(Car, Result) and Result = pass.
```

(a) (i) `DeadlineYear` is assigned value `2011`.

Identify the car registrations for cars which are due to be tested.

....[1]

(ii) State how clause `22` determines whether or not a safety certificate will be issued.

....

....[1]

(b) If a car fails one of the two tests, a retest is allowed.

Write a new rule for this.

`retestAllowed(`............................`) if` ....

....

....[3]

(c) Logic programming uses a data structure called a list.

A new fact is added to the knowledge base.

`23 carList = [a03,p05,b04,h04,h07,a23].`

The following notation and operators are to be used with a list:

> `[X|Y]` denotes a list with:
>
> - `X` the first list element
> - `Y` the list consisting of the remaining list elements
>
> `[]` denotes an empty list

(i) The list `[a07,p03]` is denoted by `[A|B]`

State the value of `A` and `B`.

`A` = ...

`B` = ... [2]

(ii) The lists `[c03,d02,n05|C]` and `[c03,d02,n05,p05,m04]` are identical.

State the value of `C`.

`C` = ... [1]

(iii) The list `[a06,a02]` is denoted by `[D,E|F]`

State the value of `F`.

`F` = ... [1]

(d) The predicate `conCatCompare` is defined as a rule and returns `TRUE` or `FALSE` as follows:

`conCatCompare(X, Y, Z)` Concatenates the lists `X` and `Y` and compares the new list with list `Z`. If equal, the clause evaluates to `TRUE`, otherwise `FALSE`.

Consider the clause:

```
conCatCompare(X, Y, [a7,b6,c4])
```

If:

- the clause evaluates to `TRUE`
- and `Y` represents the list `[a7, b6, c4]`

State the value of `X`.

`X` = ...[1]

5 **(a)** A program calculates the exam grade awarded from a mark input by the user. The code is written as a function `CalculateGrade`.

The function:

- has a single parameter `Mark` of `INTEGER` data type
- returns the grade awarded `Grade` of `STRING` data type

The logic for calculating the grade is as follows:

Mark	**Grade**
Under 40	FAIL
40 and over and under 55	PASS
55 and over and under 70	MERIT
70 and over	DISTINCTION

The programmer designs the following table for test data:

Mark	**Description**	**Expected result (Grade)**
	Normal	
	Abnormal	
	Extreme/Boundary	

(i) Complete the table above. [3]

(ii) State why this table design is suitable for black box testing.

....

....[1]

(b) When designing and writing program code, explain what is meant by:

- an exception
- exception handling

....

....

....

....

....

....[3]

(c) A program is to be written to read a list of exam marks from an existing text file into a 1D array.

Each line of the file stores the mark for one student.

State **three** exceptions that a programmer should anticipate for this program.

1 ....

....

2 ....

....

3 ....

....[3]

(d) The following pseudocode is to read two numbers:

```
01 DECLARE Num1   : INTEGER
02 DECLARE Num2   : INTEGER
03 DECLARE Answer : INTEGER
04 TRY
05    OUTPUT "First number..."
06    INPUT Num1
07    OUTPUT "Second number..."
08    INPUT Num2
09    Answer ← Num1 / (Num2 – 6)
10    OUTPUT Answer
11 EXCEPT ThisException : EXCEPTION
12    OUTPUT ThisException.Message
13 FINALLY
14    // remainder of the program follows

...

29
30 ENDTRY
```

The programmer writes the corresponding program code.

A user inputs the number 53 followed by 6. The following output is produced:

```
First number...53
Second number...6
Arithmetic operation resulted in an overflow
```

(i) State the pseudocode line number which causes the exception to be raised.

.... [1]

(ii) Explain the purpose of the pseudocode on lines 11 and 12.

....

....

....

....

....

....[3]

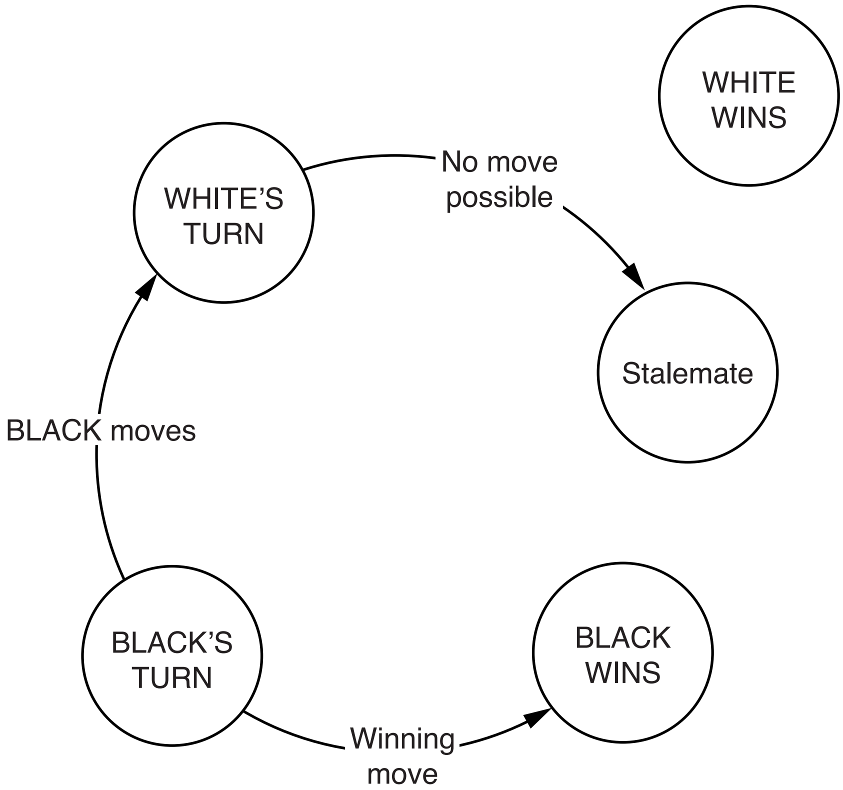
6 In a board game, one player has white pieces and the other player has black pieces. Players take alternate turns to move one of their pieces. White always makes the first move.

The game ends if:

- a player is unable to make a move when it is their turn. In this case, there is no winner. This is called ‘stalemate’.
- a player wins the game as a result of their last move and is called a ‘winner’.

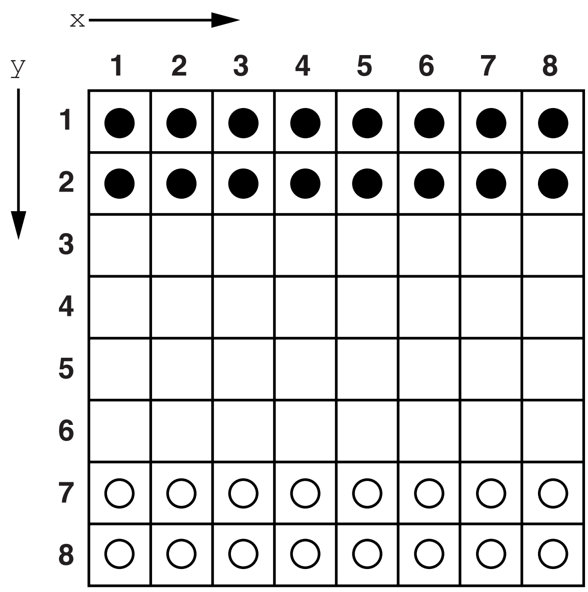
(a) A state-transition diagram is drawn to clarify how the game is played.

Complete the following state-transition diagram.



[4]

(b) The layout of the board at the start of the game is shown below:



The programmer decides to use a 2D array to represent the board. The index numbering to be used is as shown.

Each square on the board is either occupied by one piece only, or is empty.

The data stored in the array indicate whether or not that square is occupied, and if so, with a black piece or a white piece.

(i) Write **program code** to initialise the contents of the array to represent the board at the start of the game. Use characters as follows for each square:

- `'B'` represents a black piece ●
- `'W'` represents a white piece ○
- `'E'` represents an empty square

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

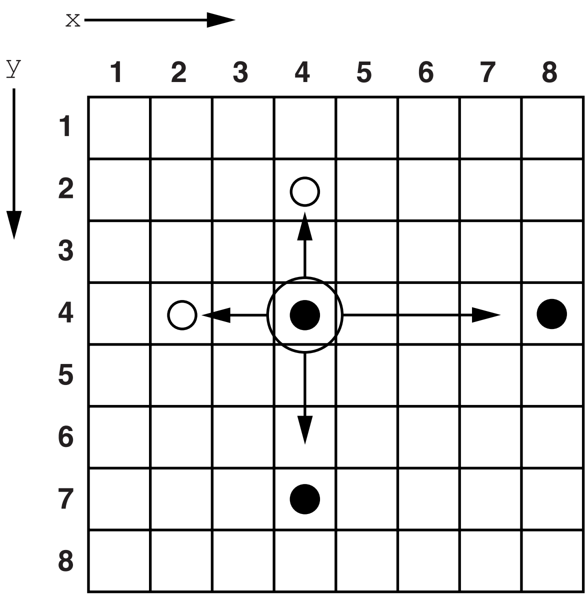
(ii) When a piece is to be moved, a procedure will calculate and output the possible destination squares for the moving piece.

A piece can move one or more squares, in the `x` or `y` direction, from its current position.

This will be a move:

- either to an empty square, with no occupied squares on the way
- or to a square containing a piece belonging to another player, with no occupied squares on the way. The other player's piece is then removed.

For example, for the circled black piece there are nine possible destination squares. Each of the two destination squares contains a white piece which would be removed.



The program requires a procedure `ValidMoves`.

It needs three parameters:

- `PieceColour` – colour of the moving piece
- `xCurrent` – current `x` position
- `yCurrent` – current `y` position

The procedure will calculate all possible destination squares **in the x direction only**.

Example output for the circled black piece is:

```
Possible moves are:
Moving LEFT
3 4
2 4 REMOVE piece
Moving RIGHT
5 4
6 4
7 4
```

Write **program code** for procedure `ValidMoves` with the following procedure header:

```
PROCEDURE ValidMoves(PieceColour : CHAR, xCurrent : INTEGER,
                                          yCurrent : INTEGER).
```

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(c) The problem is well suited to an object-oriented design followed by object-oriented programming.

(i) Describe how classes and objects could be used in this problem.

....

....

....

....

....[2]

(ii) For a class you identified in **part(c)(i)**, state **two** properties and **two** methods.

Class ....

Properties

1 ....

2 ....

Methods

1 ....

2 ....
[2]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME		
CENTRE NUMBER		CANDIDATE NUMBER

6567705812

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (CE) 205104/3

1 Data types can be defined using pseudocode.

The data type, `BicycleRecord`, is defined in pseudocode as:

```
TYPE BicycleRecord

   DECLARE BicycleID        :  INTEGER

   DECLARE BicycleAvailable :  BOOLEAN

   DECLARE BicycleLocation  :  (RiverSide, BusStation, TrainStation,
                                TownSquare, Library)

   DECLARE DateChecked      :  DATE

ENDTYPE
```

A variable, `LoanBicycle`, is declared in pseudocode as:

```
    DECLARE LoanBicycle : BicycleRecord
```

(a) Write **pseudocode** statements to assign `567` to the `BicycleID` of `LoanBicycle` **and** `FALSE` to the `BicycleAvailable` of `LoanBicycle`.

....

.... [2]

(b) The type definition for `BicycleRecord` is changed.

(i) The definition has been extended to include borrower identification numbers, `BorrowerID`, for the last 10 people who borrowed a bicycle. Each identification number is an integer.

Write the **pseudocode** statement needed in the type definition of `BicycleRecord`.

....

.... [1]

(ii) The values for the field `BicycleID` must be between `500` and `599` inclusive.

Rewrite **one pseudocode** line from the type definition of `BicycleRecord` to implement the change.

....

.... [1]

(c) Data about all the bicycles are stored in a file that uses random file organisation.

`BicycleID` is used as the key field.

Explain how a program could add an extra record to this file.

....

....

....

....

....

.... [3]

2 The TCP/IP protocol suite can be viewed as a stack with four layers.

(a) Complete the diagram of the stack by writing the names of the **three** missing layers.

Layer

Internet / Network

[3]

(b) Describe the TCP and IP protocols.

TCP ....

....

....

....

IP ....

....

....

....

[4]

3 **(a)** The truth table for a logic circuit with four inputs is shown.

INPUT				**OUTPUT**
P	**Q**	**R**	**S**	**X**
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(i) Write the Boolean algebraic expression for the truth table as a sum-of-products.

X = ... [2]

(ii) Complete the following Karnaugh Map (K-map) for the truth table.

		PQ			
		00	**01**	**11**	**10**
RS	**00**				
	01				
	11				
	10				

[2]

(iii) The K-map can be used to simplify the expression in **part (a)(i)**.

Draw loop(s) around appropriate groups in the K-map to produce an optimal sum-of-products. [2]

(iv) Write the simplified sum-of-products from the K-map.

X = ... [2]

(b) Simplify the expression for **X**, represented by the truth table in **part (a)**, using Boolean Algebra.

....

....

....

.... [2]

4 Babar is building a wireless Local Area Network (LAN).

(a) Identify **two** differences between a wireless network and a wired network.

1 ....

....

....

2 ....

....

....
[2]

(b) Identify the hardware device needed to connect the wireless LAN to the Internet. Justify your choice.

Device ....

Justification ....

....

....
[2]

5 **(a)** Flora has written a program that uses the variables `p`, `q`, `r` and `s`.

Part of the program contains the following calculations:

```
p = 5
q = 4
r = 1
s = p * (p - q + r)
```

(i) Write the Reverse Polish Notation (RPN) for the expression:

```
p * (p - q + r)
```

.... [2]

(ii) Show the changing contents of the stack as the value for `s` is calculated from the RPN expression.

[4]

(b) Convert this RPN expression back to its original infix form.

```
p q * p q r + - + p /
```

....

....

....

....

.... [3]

(c) Syntax analysis is one stage of the compilation of a program.

Identify **and** describe **two other** stages of compilation.

Stage 1 ....

Description ....

....

....

Stage 2 ....

Description ....

....

....
[4]

6 Lara wants to send an important message to her bank over the Internet.

(a) Explain why the bank requires a digital signature for the message.

....

....

....

.... [2]

(b) Asymmetric key cryptography is used to keep Lara's message secure during transmission over the Internet.

Describe this process of encrypting and decrypting Lara's message.

....

....

....

....

....

....

....

....

....

.... [5]

(c) Lara has received an email message that appears to be from her bank. She is not sure whether it is authentic.

State **two** problems that could occur if Lara opens and responds to this suspicious email message.

1 ....

....

2 ....

....
[2]

7 A virtual machine is the software emulation of a computer system using another computer system.

Describe **two** benefits and **two** limitations of using a virtual machine.

Benefit 1 ....

....

....

Benefit 2 ....

....

....

Limitation 1 ....

....

....

Limitation 2 ....

....

....
[4]

8 A large car park has 6 floors.

There is a large screen at the entrance to the car park. This screen displays the number of empty parking spaces on each floor.

- There are 256 parking spaces for cars on each floor.
- Each parking space has a sensor that detects if a car is parked on it.
- Data from the sensors are read and processed by a computer system.

(a) **(i)** Identify the type of system described.

.... [1]

(ii) Justify your answer to **part (a)(i)**.

....

....

....

.... [2]

(iii) Identify **two** types of sensor that could be used by this system.

Sensor 1 ....

Sensor 2 ....
[2]

(b) A program regularly checks each sensor's readings.

The number of empty parking spaces is displayed on the screen for each floor. If there are no empty parking spaces on a floor, a message is displayed on the screen to show that the floor is full.

The pseudocode algorithm to display this information has been written using these identifiers.

Identifier	Data type	Description
`FloorNumber`	`INTEGER`	Floor number
`SpaceNumber`	`INTEGER`	Parking space number
`SpaceAvailable`	`INTEGER`	Counts number of empty parking spaces on a floor
`ForEver`	`BOOLEAN`	Value to ensure continuous loop

The pseudocode algorithm uses the function `CheckSpace(Floor, Space)`. This function returns `TRUE` if the parking space is empty and `FALSE` otherwise.

(i) Complete the following **pseudocode** algorithm to check the number of parking spaces available.

```
ForEver ← ....................................................................................
REPEAT
  FOR FloorNumber ← 1 TO ....................................................................................
    SpaceAvailable ← 0
    FOR SpaceNumber ← 1 TO ....................................................................................
       IF CheckSpace(FloorNumber, SpaceNumber)
         THEN
           SpaceAvailable ← SpaceAvailable + 1
       ENDIF
    ENDFOR
    IF SpaceAvailable > ....................................................................................
      THEN
        OUTPUT "Floor ", ....................................................................................,
             " empty parking spaces "....................................................................................
      ELSE
        OUTPUT "Floor ", ...................................................................................., " full"
    ENDIF
  ENDFOR

// delay loop
// delay loop
UNTIL ....................................................................................
```

[6]

(ii) Write a delay loop in **pseudocode** for lines `19` and `20` of the pseudocode algorithm.

....

....

....

.... [2]

(iii) State why a delay loop is used in this system.

....

.... [1]

(c) When a car is parked in a parking space, a bit is set in the appropriate memory location.

- 32 memory locations are used for each floor with address `X01` to `X32`, where `X` is the number of the floor.
- Each location is one byte in length to hold the data for 8 parking spaces.

For example, memory location `101` is used for parking spaces 1 to 8 and memory location `102` is used for parking spaces 9 to 16.

The table shows part of floor 1 with cars parked in parking spaces 4, 11, 16 ... and 255.

Memory location	**Bits**								**Parking space number**
101	0	0	0	0	1	0	0	0	**8 – 1**
102	1	0	0	0	0	1	0	0	**16 – 9**
…	…	…	…	…	…	…	…	…	**…**
132	0	1	0	0	0	0	0	0	**256 – 249**

(i) The data in memory location `604` is shown.

604	1	0	0	0	0	0	0	1

State what this data represents.

....

....

....

.... [2]

(ii) Explain the way in which the parking space 254 on floor 5 can be checked to see if it is empty.

....

....

....

....

....

....

....

....

....

.... [5]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

4602197819

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **May/June 2022**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (EF) 311847

1 Data types can be defined using pseudocode.

The data type, `LibraryRecord`, is defined in pseudocode as:

```
TYPE LibraryRecord
   DECLARE Title : STRING
   DECLARE Fiction : BOOLEAN
   DECLARE Author : STRING
   DECLARE NumberOfCopies : INTEGER
ENDTYPE
```

A variable, `LibraryBook`, is declared in pseudocode as:

```
DECLARE LibraryBook : LibraryRecord
```

(a) Write **pseudocode** statements to assign:

- `A Level Computer Science` to `Title` of `LibraryBook`
- `FALSE` to `Fiction` of `LibraryBook`.

....

....

....

.... [2]

(b) The type definition for `LibraryRecord` is changed.

(i) The value for `NumberOfCopies` must be between `1` and `10` inclusive.

Write the updated line of **pseudocode** from the type definition of `LibraryRecord` to implement the change.

....

.... [1]

(ii) Every copy of every book is now uniquely identified by an accession number, `AccessionNumber`, as it is added to the library. Each library record will include one or more accession numbers. Each accession number is an integer.

Write the extra line of **pseudocode** needed in the type definition of `LibraryRecord`.

....

....

....

.... [2]

(c) A record is a user-defined composite data type.

Explain what is meant by a **user-defined composite data type**.
Include an example of **another** user-defined composite data type in your answer.

....

....

....

....

....

....
[3]

2 A declarative language is used to represent the following facts about cats.

```
01 type(leopard, wild).
02 type(lion, wild).
03 type(cheetah, wild).
04 type(savannah, hybrid).
05 type(persian, domestic).
06
07 hair(leopard, medium).
08 hair(lion, short).
09 hair(cheetah, medium).
10 hair(savannah, medium).
11 hair(persian, long).
12
13 spots(leopard, yes).
14 spots(lion, no).
15 spots(cheetah, yes).
16 spots(savannah, yes).
17 spots(persian, no).
```

These clauses have the following meaning:

Clause	**Meaning**
01	A leopard is a type of wild cat.
08	A lion has short hair.
16	A savannah has spots.

(a) More facts are to be included. A **caracal** is a wild cat with short hair.

Write the additional clauses to record these facts.

18 ....

19 ....
[2]

(b) Using the variable `Cat`, the goal:

```
hair(Cat, medium)
```

returns

```
Cat = leopard, cheetah, savannah
```

Write the result returned by the goal:

```
hair(Cat, long)
```

`Cat =` .... [1]

(c) **(i)** Write the goal, using the variable `Pet`, to find all the domestic cats.

....

.... [1]

(ii) Write the goal, using the variable `WildSpotty`, to find all the wild cats with spots.

....

....

....

.... [2]

3 Data can be sent over networks using either circuit switching or packet switching.

Describe both methods of data transmission. Include a different advantage and disadvantage for each method.

Circuit switching ....

....

....

....

....

Advantage ....

....

Disadvantage ....

....

Packet switching ....

....

....

....

....

Advantage ....

....

Disadvantage ....

....

[8]

4 Reduced Instruction Set Computers (RISC) and Complex Instruction Set Computers (CISC) are two types of processor.

(a) Describe what is meant by **RISC** and **CISC processors**.

RISC ....

....

....

....

CISC ....

....

....

....
[4]

(b) Identify **two** differences between RISC and CISC processors.

1 ....

....

....

2 ....

....

....
[2]

5 Part of a program's calculations uses the integer variables `j`, `k`, `m`, `n` and `p`.

```
j = 3
k = 2
m = 10
n = (j + k)/(j - k)
p = m * (m - j * k)
```

(a) Write the Reverse Polish Notation (RPN) for the expression:

```
(j + k)/(j - k)
```

.... [2]

(b) (i) Show the changing contents of the stack as the value for `p` is calculated from its RPN expression:

```
m m j k * - *
```


[4]

(ii) Describe the main steps in the evaluation of this RPN expression using a stack.

....

....

....

....

....

....

....

.... [4]

(c) State **two other** uses of a stack.

1 ....

....

2 ....

....
[2]

6 A virtual machine is used to emulate a new computer system.

Describe **two** benefits and **one** limitation of using a virtual machine for this purpose.

Benefit 1 ....

....

....

....

....

Benefit 2 ....

....

....

....

....

Limitation ....

....

....

....

....
[6]

7 A program is to be written using Object-Oriented Programming (OOP) for a shop that sells knitting yarn. There are three types of yarn: acrylic, wool or mix.

The following data are stored for each type.

- Name
- Colour
- Batch code
- Weight
- Number of balls of yarn in stock (can be edited)
- Type of yarn

The following statements apply to yarn.

- Acrylic can be soft or not soft.
- Wool can be lamb, merino or alpaca.
- Mix contains a percentage of acrylic.

Each type of yarn has a method that will display all the information about the yarn.

(a) Complete this class inheritance diagram to show the **properties**, **methods** and **inheritance**.

Yarn
`Name: STRING` `Colour: STRING` `BatchCode: STRING` `Weight: INTEGER` `NumberBalls: INTEGER` `Type: STRING`
`Constructor()` `EditNumberBalls()` `YarnInfo()`

Acrylic
....
`Constructor()` ....

Wool
....
`Constructor()` ....

Mix
`Percentage: INTEGER`
`Constructor()` `YarnInfo()`

[5]

(b) Describe what is meant by the terms **properties**, **methods** and **inheritance**.

Properties ....

....

....

....

Methods ....

....

....

....

Inheritance ....

....

....

....
[6]

8 A message is to be sent securely. Software uses a key to encrypt the message before it is sent.

(a) **(i)** Give **two** reasons for using key cryptography.

1 ....

....

2 ....

....
[2]

(ii) Give **two** methods of key cryptography that can be used.

1 ....

2 ....
[2]

(b) When there is a secure exchange of key(s), the message is sent.

The use of quantum cryptography is being considered for the secure exchange.

(i) State **two** possible benefits of using quantum cryptography.

1 ....

....

....

2 ....

....

....
[2]

(ii) State **two** possible drawbacks of using quantum cryptography.

1 ....

....

....

2 ....

....

....
[2]

BLANK PAGE

9 The table shows assembly language instructions for a processor that has one general purpose register, the Accumulator (ACC).

	Instruction		Explanation
Label	**Opcode**	**Operand**	
	`LDM`	`#n`	Load the number n to ACC
	`LDD`	`<address>`	Load the contents of the given address to ACC
	`LDI`	`<address>`	The address to be used is at the given address Load the contents of this second address to ACC
	`ADD`	`<address>`	Add the contents of the given address to the ACC
	`STO`	`<address>`	Store the contents of the ACC at the given address
`<label>:`		`<data>`	Gives a symbolic address <label> to the memory location with the contents <data> <label> can be used in place of <address>
# denotes a denary number, e.g. #123			

(a) The address `500` contains the value `100` and the address `100` contains the value `20`.

State the addressing mode and the contents of ACC after each instruction has been executed.

`LDM #500` Addressing mode ....

Contents of ACC ....

`LDD 500` Addressing mode ....

Contents of ACC ....

`LDI 500` Addressing mode ....

Contents of ACC ....

[3]

(b) Use only the given instruction set to write **assembly language** code to:

- use the constant 20 which needs to be stored
- add this constant to the value stored in the address contained in the variable Y
- store the result in variable Z.

Label	Instruction	
	Opcode	**Operand**

[7]

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME	

CENTRE NUMBER						CANDIDATE NUMBER					

5010748614

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2017**

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 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.
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.

The maximum number of marks is 75.

This document consists of **15** printed pages and **1** blank page.

DC (RW/SW) 129953/3

CAMBRIDGE International Examinations

[Turn over

1 The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		**Explanation**
Op code	**Operand**	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`STO`	`<address>`	Store the contents of ACC at the given address.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`JMP`	`<address>`	Jump to given address.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`<address>`	Bitwise XOR operation of the contents of ACC with the contents of <address>.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>.
`IN`		Key in a character and store its ASCII value in ACC.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

(a) A programmer writes a program that:

- reads a character from the keyboard (assume it will be a capital letter)
- outputs the alphabetical sequence of characters from 'A' to the character input. For example, if the character 'G' is input, the output is:

ABCDEFG

The programmer has started to write the program in the table on the following page. The Comment column contains descriptions for the missing instructions, labels and data.

Complete the following program. Use op codes from the given instruction set.

Label	Op code	Operand	Comment
START:			// INPUT character
			// store in CHAR
			// Initialise ACC (ASCII value for 'A' is 65)
			// OUTPUT ACC
			// compare ACC with CHAR
			// if equal jump to end of FOR loop
			// increment ACC
			// jump to LOOP
ENDFOR:	END		
CHAR:			

[8]

(b) The programmer now starts to write a program that:

- tests whether an 8-bit two's complement integer stored at address NUMBER is positive or negative
- outputs 'P' for a positive integer and 'N' for a negative integer.

Complete the following program. Use op codes from the given instruction set.
Show the required value of MASK in binary.

Label	Op code	Operand	Comment
START:			
		MASK	// set to zero all bits except sign bit
			// compare with 0
			// if not equal jump to ELSE
THEN:			// load ACC with 'P' (ASCII value 80)
	JMP	ENDIF	
ELSE:			// load ACC with 'N' (ASCII value 78)
ENDIF:			
	END		
NUMBER:	B00000101		// integer to be tested
MASK:			// value of mask in binary

[7]

2 A hash table has these associated operations:

- create hash table
- insert record
- search hash table

A hash table is to be used to store customer records.

Each record consists of a unique customer ID, the record key, and other customer data.

(a) The following pseudocode declares a customer record structure.

```
TYPE CustomerRecord
   CustomerID : INTEGER
   Data : STRING
ENDTYPE
```

The hash table is to be implemented as a 1D array `Customer` with elements indexed 0 to 199. The procedure to create a hash table will declare and initialise the array by storing 200 records with the `CustomerID` field in each record set to 0.

Complete the **pseudocode**.

```
PROCEDURE CreateHashTable()

   ..................................................................................

   ..................................................................................

   ..................................................................................

   ..................................................................................

ENDPROCEDURE
```

[4]

(b) A hashing function `Hash` exists, which takes as a parameter the customer ID and returns an integer in the range 0 to 199 inclusive.

(i) The procedure, `InsertRecord`, takes as a parameter the customer record to be inserted into the hash table.

The procedure makes use of the function `Hash`. Collisions will be managed using open hashing. This means a collision is resolved by storing the record in the next available location. The procedure will generate an error message if the hash table is full.

Complete the **pseudocode** for the procedure.

```
PROCEDURE InsertRecord(BYVALUE NewCustomer : CustomerRecord)
   TableFull ← FALSE
   // generate hash value
   Index ← .....................................................................
   Pointer ← Index   // initialise Pointer variable to hash value

   // find a free table element
   WHILE .......................................................................
      Pointer ← ................................................................
      // wrap back to beginning of table if necessary
      IF .......................................................................
         THEN
            ....................................................................
      ENDIF
      // check if back to original index
      IF .......................................................................
         THEN
            TableFull ← TRUE
      ENDIF
   ENDWHILE

   IF ..........................................................................
      THEN
         .......................................................................
      ELSE
         .......................................................................
   ENDIF
ENDPROCEDURE
```

[9]

(ii) The function `SearchHashTable` will search for a record in the hash table. The function takes as a parameter the customer ID to be searched for. The function will return the position in the hash table where the record has been saved. If the hash table does not contain the record, the function will return the value –1.

You can assume that there is at least one empty record in the hash table.

Complete the **pseudocode** for the function.

```
FUNCTION SearchHashTable(BYVALUE SearchID : INTEGER) RETURNS INTEGER

   // generate hash value

   Index ← ..................................................................

   // check each record from index until found or not there

   WHILE (..................................................................)

      AND (..................................................................)

       ..................................................................

   // wrap if necessary

      IF ..................................................................

         THEN

             ..................................................................

      ENDIF

   ENDWHILE

   // has customer ID been found?

   IF ..................................................................

      THEN

          ..................................................................

      ELSE

          ..................................................................

   ENDIF

ENDFUNCTION
```

[9]

(iii) A record that is no longer required is deleted.

State the problem that might be caused by this deletion.

....

....[1]

3 `NameList` is a 1D array that stores a sorted list of names. A programmer declares the array in pseudocode as follows:

```
NameList : Array[0 : 100] OF STRING
```

The programmer wants to search the list using a binary search algorithm.

The programmer decides to write the search algorithm as a recursive function. The function, `Find`, takes three parameters:

- `Name`, the string to be searched for
- `Start`, the index of the first item in the list to be searched
- `Finish`, the index of the last item in the list to be searched

The function will return the position of the name in the list, or −1 if the name is not found.

Complete the **pseudocode** for the recursive function.

```
FUNCTION Find(BYVALUE Name : STRING, BYVALUE Start : INTEGER,
                              BYVALUE Finish : INTEGER) RETURNS INTEGER

   // base case

   IF .....................................................................

      THEN

         RETURN −1

      ELSE

         Middle ← ..........................................................

         IF ................................................................

            THEN

               RETURN ......................................................

            ELSE    // general case

               IF SearchItem > .............................................

                  THEN

                     ...................................................

                  ELSE

                     ...................................................

               ENDIF

         ENDIF

   ENDIF

ENDFUNCTION
```

[7]

4 An ordered linked list Abstract Data Type (ADT) has these associated operations:

- create list
- add item to list
- output list to console

The ADT is to be implemented using object-oriented programming as a linked list of nodes.

Each node consists of data and a pointer.

(a) There are two classes, `LinkedList` and `Node`.

(i) State the term used to describe the relationship between these classes.

....[1]

(ii) Draw the appropriate diagram to represent this relationship. Do not list the attributes and methods of the classes.

[2]

(b) The design for the `Node` class consists of:

- attributes
  - `Data : STRING`
  - `Pointer : INTEGER`
- methods
  - `CreateNode(Data, Pointer)`
  - `SetData(Data)`
  - `SetPointer(Pointer)`
  - `GetData() RETURNS STRING`
  - `GetPointer() RETURNS INTEGER`

The constructor method sets the attributes to the initial values that are passed as parameters.

Write **program code** for:

- the `Node` class declaration
- the constructor.

Programming language used ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

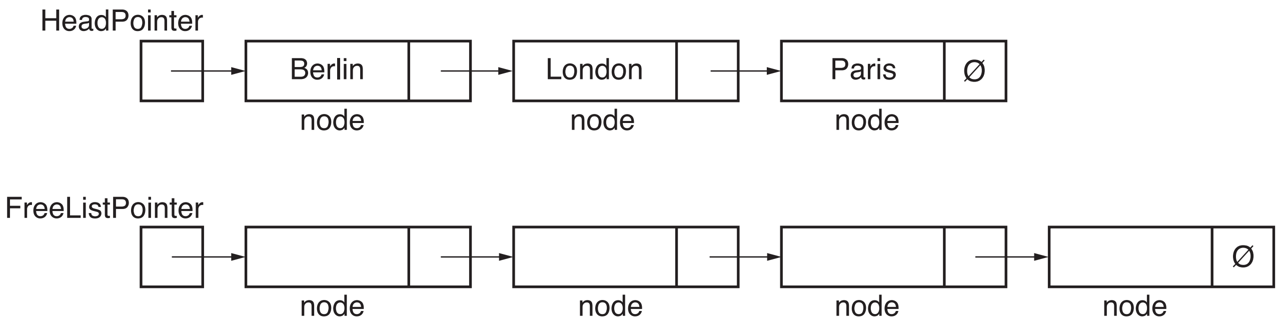
....

....[5]

(c) The identifier table for the `LinkedList` class is:

Identifier	Data type	Description
`HeadPointer`	`INTEGER`	Pointer to the first node in the ordered list.
`FreeListPointer`	`INTEGER`	Pointer to the first node in the free list.
`NodeArray`	`ARRAY[0 : 7] OF Node`	1D array stores the nodes that make the ordered linked list. The unused nodes are linked together into a free list.
`Constructor()`		Constructor instantiates an object of `LinkedList` class, initialises `HeadPointer` to be a null pointer and links all nodes to form the free list.
`FindInsertionPoint()`		Procedure that takes the new data item as the parameter `NewData` and returns two parameters: • `PreviousPointer`, whose value is: ◦ either pointer to node before the insertion point ◦ or the null pointer if the new node is to be inserted at the beginning of the list. • `NextPointer`, whose value is a pointer to node after the insertion point.
`AddToList(NewString)`		Procedure that takes as a parameter a unique string and links it into the correct position in the ordered list.
`OutputListToConsole()`		Procedure to output all the data from the list pointed to by `HeadPointer`.

The following diagram shows an example of a linked list object. This example list consists of three nodes, linked in alphabetical order of the data strings. The unused nodes are linked to form a free list.



The symbol Ø represents a null pointer.

(i) Explain the meaning of the term **null pointer**.

....

....[1]

(ii) Give an appropriate value to represent the null pointer for this design. Justify your answer.

....

....

....

....[2]

(iii) Write **program code** for the `LinkedList` class declaration **and** the constructor.

Programming language used ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[7]

(iv) Write **program code** to instantiate a linked list object with the `contacts` identifier.

Programming language used ....

Program code

....

....[1]

(v) The `OutputListToConsole` method is to output all the data stored in the linked list. `HeadPointer` points to the first list node.

Write **program code** for this method.

Programming language used ....

Program code

....

....

....

....

....

....

....

....

....

....

....[5]

Question 4 continues on page 14.

(vi) The structured English for the `AddToList(NewString)` method is as follows:

```
Make a copy of the value of free list pointer, name it NewNodePointer

Store new data item in free node pointed to by NewNodePointer

Adjust free list pointer to point to next free node

IF linked list is currently empty

   THEN

      Make this node the first node

      Set pointer of this node to null pointer

   ELSE

      Find insertion point using the FindInsertionPoint method

      // FindInsertionPoint provides

      // pointer to previous node and pointer to next node

      IF previous pointer is null pointer

         THEN

            Link this node to front of list

         ELSE

            Link this node between previous node and next node
```

The `FindInsertionPoint` method receives the new data item as the parameter `NewString`. It returns two parameters:

- `PreviousPointer`, whose value is:
  - either the pointer to the node before the insertion point
  - or the null pointer, if the new node is to be inserted at the beginning of the list.
- `NextPointer`, whose value is the pointer to the node after the insertion point.

Write **program code** for the `AddToList` method.

Programming language used ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

BLANK PAGE

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME	

CENTRE NUMBER							CANDIDATE NUMBER					

5747741459

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2015**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

DC (ST) 95543/3

CAMBRIDGE International Examinations

[Turn over

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa, followed by
- 8 bits for the exponent

Two's complement form is used for both mantissa and exponent.

(a) **(i)** A real number is stored as the following two bytes:

Mantissa

0	0	1	0	1	0	0	0

Exponent

0	0	0	0	0	0	1	1

Calculate the denary value of this number. Show your working.

....

....

....

....

....

....[3]

(ii) Explain why the floating-point number in **part (a)(i)** is not normalised.

....

....[2]

(iii) Normalise the floating-point number in **part (a)(i)**.

Mantissa

Exponent

[2]

(b) **(i)** Write the largest positive number that can be written as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(ii) Write the smallest positive number that can be written as a normalised floating-point number in this format.

Mantissa

Exponent

[2]

(iii) If a positive number is added to the number in **part (b)(i)** explain what will happen.

....

....

....

....[2]

(c) A student writes a program to output numbers using the following code:

```
X ← 0.0
FOR i ← 0 TO 1000
   X ← X + 0.1
   OUTPUT X
ENDFOR
```

The student is surprised to see that the program outputs the following sequence:

0.0   0.1   0.2   0.2999999   0.3999999 ……

Explain why this output has occurred.

....

....

....

....

....

....[3]

2 A compiler uses a keyword table and a symbol table. Part of the keyword table is shown below.

- Tokens for keywords are shown in hexadecimal.
- All the keyword tokens are in the range 00 – 5F.

Keyword	Token
←	01
+	02
=	03
IF	4A
THEN	4B
ENDIF	4C
ELSE	4D
FOR	4E
STEP	4F
TO	50
INPUT	51
OUTPUT	52
ENDFOR	53

Entries in the symbol table are allocated tokens. These values start from 60 (hexadecimal).

Study the following piece of code:

```
Counter ← 1.5
INPUT Num1
  // Check values
IF Counter = Num1
   THEN
      Num1 ← Num1 + 5.0
ENDIF
```

(a) Complete the symbol table below to show its contents after the lexical analysis stage.

Symbol	Token	
	Value	**Type**
`Counter`	60	Variable
`1.5`	61	Constant

[3]

(b) Each cell below represents one byte of the output from the lexical analysis stage.

Using the keyword table and your answer to **part (a)** complete the output from the lexical analysis.

60	01														

[2]

(c) This line of code is to be compiled:

```
A ← B + C + D
```

After the syntax analysis stage, the compiler generates object code. The equivalent code, in assembly language, is shown below:

```
LDD 234     //loads value B
ADD 235     //adds value C
STO 567     //stores result in temporary location
LDD 567     //loads value from temporary location
ADD 236     //adds value D
STO 233     //stores result in A
```

(i) Name the final stage in the compilation process that follows this code generation stage.

....[1]

(ii) Rewrite the equivalent code given above to show the effect of it being processed through this final stage.

....

....

....

....

....

....[2]

(iii) State **two** benefits of the compilation process performing this final stage.

Benefit 1 ....

....

Benefit 2 ....

....[2]

3 An email is sent from one email server to another using packet switching.

(a) State **two items** that are contained in an email packet apart from the data.

1 ....

2 ....[2]

(b) Explain the role of routers in sending an email from one email server to another.

....

....

....

....

....

....[3]

(c) Sending an email message is an appropriate use of packet switching.

Explain why this is the case.

....

....

....

....

....

....[2]

(d) Packet switching is not always an appropriate solution.

Name an alternative communication method of transferring data in a digital network.

....[1]

(e) Name an application for which the method identified in **part (d)** is an appropriate solution. Justify your choice.

Application ....

Justification ....

....

....

....

....[3]

4 **(a)** Three descriptions and two types of processor are shown below.

Draw a line to connect each description to the appropriate type of processor.

Description	**Type of processor**
Makes extensive use of general purpose registers	RISC
Many addressing modes are available	CISC
Has a simplified set of instructions	

[3]

(b) In a RISC processor three instructions (A followed by B, followed by C) are processed using pipelining.

The following table shows the five stages that occur when instructions are fetched and executed.

(i) The 'A' in the table indicates that instruction A has been fetched in time interval 1.

Complete the table to show the time interval in which each stage of each instruction (A, B, C) is carried out.

Stage	**Time interval**								
	1	2	3	4	5	6	7	8	9
Fetch instruction	A								
Decode instruction									
Execute instruction									
Access operand in memory									
Write result to register									

[3]

(ii) The completed table shows how pipelining allows instructions to be carried out more rapidly. Each time interval represents one clock cycle.

Calculate how many clock cycles are saved by the use of pipelining in the above example.

Show your working.

....

....

....

....

....

....[3]

5 **(a)** **(i)** Complete the Boolean function that corresponds to the following truth table.

INPUT			**OUTPUT**
A	**B**	**C**	**X**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$X = \overline{A} . B . C +$ ....[3]

The part to the right of the equals sign is known as the sum-of-products.

(ii) For the truth table above complete the Karnaugh Map (K-map).

		AB			
		00	**01**	**11**	**10**
C	**0**				
	1				

[1]

The K-map can be used to simplify the function in **part(a)(i)**.

(iii) Draw loop(s) around appropriate groups of 1's to produce an optimal sum-of-products. [2]

(iv) Using your answer to **part (a)(iii)**, write the simplified sum-of-products Boolean function.

X = ...[2]

(b) The truth table for a logic circuit with four inputs is given below:

INPUT				OUTPUT
A	**B**	**C**	**D**	**X**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

(i) Complete the K-map corresponding to the truth table above.

		AB		
CD				

[4]

(ii) Draw loop(s) around appropriate groups of 1's to produce an optimal sum-of-products. [2]

(iii) Using your answer to **part (b)(ii)**, write the simplified sum-of-products Boolean function.

X = ...[2]

6 A number of processes are being executed in a computer.

(a) Explain the difference between a program and a process.

....

....

....

....[2]

A process can be in one of three states: running, ready or blocked.

(b) For each of the following, the process is moved from the first state to the second state. Describe the conditions that cause each of the following changes of the state of a process:

From running to ready ....

....

....

....

From ready to running ....

....

....

....

From running to blocked ....

....

....

....[6]

(c) Explain why a process cannot be moved from the blocked state to the running state.

....

....

....

....

....

....[3]

(d) Explain the role of the high-level scheduler in a multiprogramming operating system.

....

....

....

....[2]

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6550085963

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2015**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **11** printed pages and **1** blank page.

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa, followed by
- 4 bits for the exponent

Two's complement form is used for both mantissa and exponent.

(a) **(i)** A real number is stored as the following 12-bit binary pattern:

0	1	1	0	1	0	0	0		0	0	1	1

Calculate the denary value of this number. Show your working.

....

....

....

....

....

....[3]

(ii) Give the normalised binary pattern for +3.5. Show your working.

....

....

....

....

....

....[3]

(iii) Give the normalised binary pattern for –3.5. Show your working.

....

....

....

....

....

....[3]

The number of bits available to represent a real number is increased to 16.

(b) **(i)** If the system were to use the extra 4 bits for the mantissa, state what the effect would be on the numbers that can be represented.

....

....[1]

(ii) If the system were to use the extra 4 bits for the exponent instead, state what the effect would be on the numbers that can be represented.

....

....[1]

(c) A student enters the following expression into an interpreter:

```
OUTPUT (0.1 + 0.2)
```

The student is surprised to see the following output:

0.30000000000000001

Explain why this output has occurred.

....

....

....

....

....

....[3]

2 In this question, you are shown pseudocode in place of a real high-level language. A compiler uses a keyword table and a symbol table. Part of the keyword table is shown below.

- Tokens for keywords are shown in hexadecimal.
- All the keyword tokens are in the range 00 to 5F.

Keyword	**Token**
←	01
+	02
=	03
IF	4A
THEN	4B
ENDIF	4C
ELSE	4D
FOR	4E
STEP	4F
TO	50
INPUT	51
OUTPUT	52
ENDFOR	53

Entries in the symbol table are allocated tokens. These values start from 60 (hexadecimal).

Study the following piece of code:

```
Start ← 0.1
// Output values in loop
FOR Counter ← Start TO 10
   OUTPUT Counter + Start
ENDFOR
```

(a) Complete the symbol table below to show its contents after the lexical analysis stage.

Symbol	**Token**	
	Value	**Type**
Start	60	Variable
0.1	61	Constant

[3]

(b) Each cell below represents one byte of the output from the lexical analysis stage.

Using the keyword table and your answer to **part (a)** complete the output from the lexical analysis.

60	01												

[2]

(c) The compilation process has a number of stages. The output of the lexical analysis stage forms the input to the next stage.

(i) Name this stage.

....[1]

(ii) State **two** tasks that occur at this stage.

....

....

....

....[2]

(d) The final stage of compilation is optimisation. There are a number of reasons for performing optimisation. One reason is to produce code that minimises the amount of memory used.

(i) State another reason for the optimisation of code.

....[1]

(ii) What could a compiler do to optimise the following expression?

```
A ← B + 2 * 6
```

....

....

....[1]

(iii) These lines of code are to be compiled:

```
X ← A + B
Y ← A + B + C
```

Following the syntax analysis stage, object code is generated. The equivalent code, in assembly language, is shown below:

```
LDD 436    //loads value A
ADD 437    //adds value B
STO 612    //stores result in X
LDD 436    //loads value A
ADD 437    //adds value B
ADD 438    //adds value C
STO 613    //stores result in Y
```

(iv) Rewrite the equivalent code, given above, following optimisation.

....

....

....

....

....

....[3]

3 **(a)** Explain what is meant by circuit switching.

....

....

....

....[2]

(b) There are many applications in which digital data are transferred across a network. Video conferencing is one of these.

For this application, circuit switching is preferable to the use of packet switching.

Explain why this is so.

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

(c) A web page is transferred from a web server to a home computer using the Internet.

Explain how the web page is transferred using packet switching.

....

....

....

....

....

....[3]

4 **(a)** Four descriptions and four types of computer architecture are shown below.

Draw a line to connect each description to the appropriate type of computer architecture.

Description	Computer architecture
A computer that does not have the ability for parallel processing.	SIMD
The processor has several ALUs. Each ALU executes the same instruction but on different data.	MISD
There are several processors. Each processor executes different instructions drawn from a common pool. Each processor operates on different data drawn from a common pool.	SISD
There is only one processor executing one set of instructions on a single set of data.	MIMD

[4]

(b) In a massively parallel computer explain what is meant by:

(i) Massive ....

....

....[1]

(ii) Parallel ....

....

....[1]

(c) There are both hardware and software issues that have to be considered for parallel processing to succeed.

Describe **one** hardware and **one** software issue.

Hardware ....

....

....

....

Software ....

....

....

....[4]

5 **(a)** **(i)** Complete the Boolean function that corresponds to the following truth table.

INPUT			**OUTPUT**
P	**Q**	**R**	**Z**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$Z = P \,.\, \overline{Q} \,.\, \overline{R} +$ ....[3]

The part to the right of the equals sign is known as the sum-of-products.

(ii) For the truth table above complete the Karnaugh Map (K-map).

		PQ			
		00	**01**	**11**	**10**
R	**0**				
	1				

[1]

The K-map can be used to simplify the function in **part(a)(i)**.

(iii) Draw loop(s) around appropriate groups of 1's to produce an optimal sum-of-products. [2]

(iv) Using your answer to **part (a)(iii)**, write the simplified sum-of-products Boolean function.

Z = ...[1]

(b) The truth table for a logic circuit with four inputs is given below:

INPUT				OUTPUT
P	**Q**	**R**	**S**	**Z**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

(i) Complete the K-map corresponding to the truth table above.

PQ

RS				

[4]

(ii) Draw loop(s) around appropriate groups of 1's to produce an optimal sum-of-products. [2]

(iii) Using your answer to **part (b)(ii)**, write the simplified sum-of-products Boolean function.

Z = ...[2]

6 A number of processes are being executed in a computer.

A process can be in one of three states: running, ready or blocked.

(a) For each of the following, the process is moved from the first state to the second state. Describe the conditions that cause each of the following changes of state of a process:

From blocked to ready ....

....

....

....

From running to ready ....

....

....

....[4]

(b) Explain why a process cannot move directly from the ready state to the blocked state.

....

....

....

....

....

....[3]

(c) A process in the running state can change its state to something which is neither the ready state nor the blocked state.

(i) Name this state.

....[1]

(ii) Identify when a process would enter this state.

....[1]

(d) Explain the role of the low-level scheduler in a multiprogramming operating system.

....

....

....

....[2]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME											
CENTRE NUMBER							CANDIDATE NUMBER				

2579440869

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **May/June 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

DC (LO) 213986

1 Real numbers are stored in a computer system using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- Two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised floating-point representation of –7.25 in this system.
Show your working.

Mantissa

Exponent

Working ....

....

....

....

....

....

....

.... [3]

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	0	1	1	1

Exponent

0	0	0	1	1	1

Working ....

....

....

....

....

Answer .... [3]

(c) The given binary floating-point number is not normalised.

Normalise the floating-point number. Show your working.

Mantissa | **Exponent**

0	0	0	0	0	0	0	1	1	1		1	0	0	1	1	1

Mantissa | **Exponent**

Working ....

....

....

....

....

....
[3]

(d) The denary number 513 cannot be stored accurately as a normalised floating-point number in this computer system.

(i) Explain the reason for this.

....

....

....

....

....

.... [3]

(ii) Describe an alteration to the way floating-point numbers are stored to enable this number to be stored accurately using the same total number of bits.

....

....

....

.... [2]

2 **(a)** Describe the purpose of a user-defined data type.

....

....

....

....

.... [2]

(b) Define, using pseudocode, the following enumerated data types:

(i) `SchoolDay` to hold data about the days students are usually in school.

....

.... [1]

(ii) `WeekEnd` to hold data about the days that are not school days.

....

.... [1]

(c) Define, using pseudocode, the composite data type `ClubMeet`. This will hold data about club members that includes:

- first name and last name
- the two days they attend:
  - one on a school day
  - one not on a school day.

Use the enumerated types you created in **part (b)**.

....

....

....

....

....

....

....

.... [4]

3 **(a)** Draw **one** line to connect each **Operating System (OS)** term to the **most appropriate** description about it.

OS term

- Multi-tasking
- Paging
- Interrupt handling
- Scheduling
- Virtual memory

Description

- Using secondary storage to simulate additional main memory
- Managing the processes running on the CPU
- Managing the execution of many programs that appear to run at the same time
- Locating non-contiguous blocks of data and relocating them
- Transferring control to another routine when a service is required
- Reading/writing same-size blocks of data from/to secondary storage when required

[5]

(b) Explain how an interpreter executes a program without producing a complete translated version of it.

....

....

....

....

....

....

....

.... [4]

4 (a) (i) Explain why Reverse Polish Notation (RPN) is used to carry out the evaluation of expressions.

....

....

....

.... [2]

(ii) Identify, with reasons, a data structure that could be used to evaluate an expression in RPN.

....

....

....

.... [2]

(b) Write the infix expression in RPN.

```
(a − b) * (a + c) / 7
```

....

.... [1]

(c) Write the RPN expression as an infix expression.

```
a b / 4 * a b + -
```

....

.... [1]

(d) Evaluate the RPN expression:

```
a b + c d / /
```

where `a = 17, b = 3, c = 48 and d = 12`.

Show your working.

....

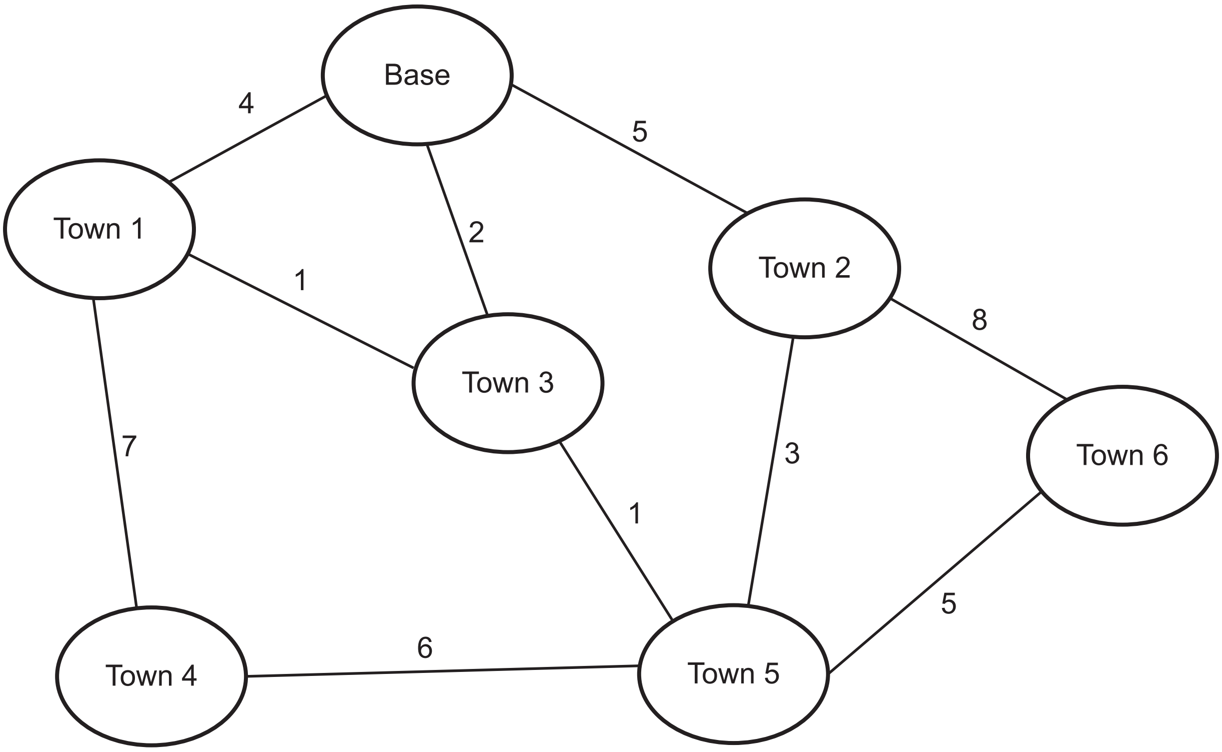
....

....

.... [2]

5 **(a)** Calculate the shortest distance between the base and each of the other towns in the diagram using Dijkstra's algorithm.

Show your working **and** write your answers in the table provided.



Working ....

....

....

....

....

....

....

....

Answers

Town 1	**Town 2**	**Town 3**	**Town 4**	**Town 5**	**Town 6**

[5]

(b) Explain the use of graphs to aid Artificial Intelligence (AI).

....

....

....

....

....

.... [3]

6 Give **two** benefits **and two** drawbacks of packet switching.

Benefit 1 ....

....

Benefit 2 ....

....

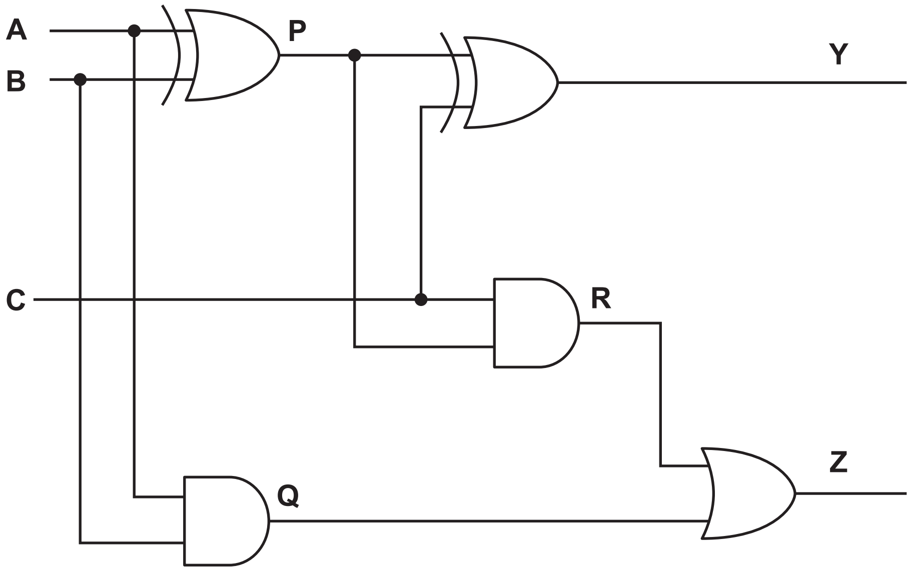
Drawback 1 ....

....

Drawback 2 ....

....
[4]

7 The diagram shows a logic circuit.



(a) Complete the truth table for the given logic circuit. Show your working.

Inputs			Working space			Outputs	
A	**B**	**C**	**P**	**Q**	**R**	**Y**	**Z**
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

[3]

(b) State the name of the logic circuit.

.... [1]

(c) Write the Boolean expressions for the two outputs **Y** and **Z** in the truth table as sum-of-products **and** state the purpose of each output.

Y = ...

Purpose ....

Z = ...

Purpose ....

[4]

8 **(a)** State **two** factors that may affect the performance of a sorting algorithm.

....

....

....

.... [2]

(b) The given algorithm is a simple bubble sort that arranges a set of scores stored in a one-dimensional array into **descending** order, and orders the corresponding students' names stored into a two-dimensional array in the same order as the scores. All the arrays are indexed from 1.

The contents of both arrays after sorting are shown.

	Score
1	98
2	97
...	
248	5
249	3

	Name	
	1	**2**
1	Smithfield	Tom
2	Johnson	Jane
...		
248	Peters	Jade
249	Allen	John

```
YearSize ← 249
Flag ← TRUE
WHILE Flag = TRUE
    Flag ← FALSE
    FOR Student ← 1 TO YearSize - 1
        IF Score[Student] < Score[Student + 1] THEN
           Temp1 ← Score[Student]
           Temp2 ← Name[Student,1]
           Temp3 ← Name[Student,2]
           Score[Student] ← Score[Student + 1]
           Name[Student,1] ← Name[Student + 1,1]
           Name[Student,2] ← Name[Student + 1,2]
           Score[Student + 1] ← Temp1
           Name[Student + 1,1] ← Temp2
           Name[Student + 1,2] ← Temp3
           Flag ← TRUE
        ENDIF
    NEXT Student
ENDWHILE
```

Write an algorithm, using pseudocode, that will perform the same task using an insertion sort.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

9 **(a)** Describe what is meant by **an imperative (procedural)** programming language.

....

....

....

.... [2]

(b) Describe what is meant by **a declarative** programming language.

....

....

....

.... [2]

(c) Identify the programming paradigm for each of these program code examples.

Program code example	**Programming paradigm**
`male(john).` `female(ethel).` `parent(john, ethel).`	
`FOR Counter = 1 TO 20` `    X = X * Counter` `NEXT Counter`	
`Start: LDD Counter` `       INC ACC` `       STO Counter`	
`public class Vehicle` `{` `    private speed;` `    public Vehicle()` `    {` `        speed = 0;` `    }` `}`	

[4]

Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

4658950581

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2019**

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 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.
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.

The maximum number of marks is 75.

This document consists of **18** printed pages and **2** blank pages.

DC (PQ) 180360

Cambridge Assessment International Education

1 **(a)** A stack contains the values `'red'`, `'blue'`, `'green'` and `'yellow'`.

`yellow`	← Top of stack
`green`	
`blue`	
`red`	

(i) Show the contents of the stack in **part(a)** after the following operations.

```
POP()

PUSH('purple')

PUSH('orange')
```


[1]

(ii) Show the contents of the stack from **part(a)(i)** after these further operations.

```
POP()

POP()

PUSH('brown')

POP()

PUSH('black')
```


[1]

(b) A queue is an alternative Abstract Data Type (ADT).

Describe a **queue**.

....

....

....

....

....

.... [3]

2 A computer games club wants to run a competition. The club needs a system to store the scores achieved in the competition.

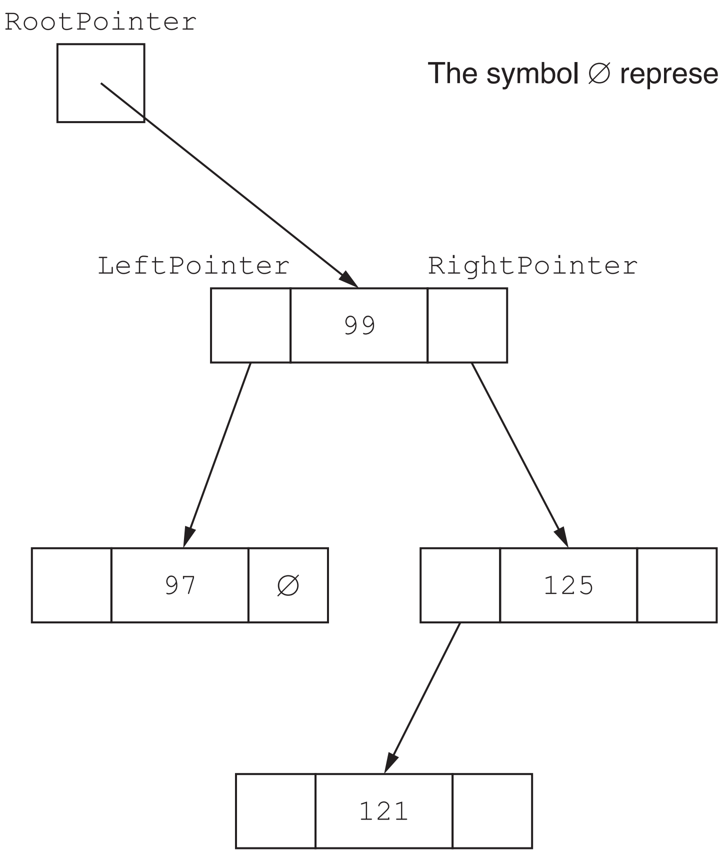
A selection of score data is as follows:

99, 125, 121, 97, 109, 95, 135, 149

(a) A linked list of nodes will be used to store the data. Each node consists of the data, a left pointer and a right pointer. The linked list will be organised as a binary tree.

(i) Complete the binary tree to show how the score data above will be organised.

The symbol ∅ represents a null pointer.



[5]

(ii) The following diagram shows a 2D array that stores the nodes of the binary tree's linked list.

Add the correct pointer values to complete the diagram, using your answer from **part (a)(i)**.

RootPointer

0

FreePointer

Index	LeftPointer	Data	RightPointer
0		99	
1		125	
2		121	
3		97	
4		109	
5		95	
6		135	
7		149	
8			

[6]

(b) The club also considers storing the data in the order in which it receives the scores as a linked list in a 1D array of records.

The following pseudocode algorithm searches for an element in the linked list.

Complete the **six** missing sections in the algorithm.

```
FUNCTION FindElement(Item : INTEGER) RETURNS ............................................

   .............................................. ← RootPointer

   WHILE CurrentPointer .............................................. NullPointer

      IF List[CurrentPointer].Data <> ..............................................

         THEN

            CurrentPointer ← List[..............................................].Pointer

         ELSE

            RETURN CurrentPointer

      ENDIF

   ENDWHILE

   CurrentPointer ← NullPointer

   ............................................ CurrentPointer

ENDFUNCTION
```

[6]

(c) The games club is looking at two programming paradigms: imperative and object-oriented programming paradigms.

Describe what is meant by the **imperative programming paradigm** and the **object-oriented programming paradigm**.

(i) Imperative ....

....

....

....

....

.... [3]

(ii) Object-oriented ....

....

....

....

....

.... [3]

(d) Players complete one game to place them into a category for the competition. The games club wants to implement a program to place players into the correct category. The programmer has decided to use object-oriented programming (OOP).

The highest score that can be achieved in the game is 150. Any score less than 50 will not qualify for the competition. Players will be placed in a category based on their score.

The following diagram shows the design for the class `Player`. This includes the properties and methods.

```
                              Player
--------------------------------------------------------------------
Score    : INTEGER // initialised to 0
Category : STRING  // "Beginner", "Intermediate",
                   // "Advanced" or "Not Qualified", initialised
                   // to "Not Qualified"
PlayerID : STRING  // initialised with the parameter InputPlayerID
--------------------------------------------------------------------
Create()         // method to create and initialise an object using
                 // language-appropriate constructor
SetScore()       // checks that the Score parameter has a valid value
                 // if so, assigns it to Score
SetCategory()    // sets Category based on player's Score
SetPlayerID()    // allows a player to change their PlayerID
                 // validates the new PlayerID
GetScore()       // returns Score
GetCategory()    // returns Category
GetPlayerID()    // returns PlayerID
```

(i) The constructor receives the parameter `InputPlayerID` to create the `PlayerID`. Other properties are initialised as instructed in the class diagram.

Write **program code** for the `Create()` constructor method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

.... [5]

(ii) Write **program code** for the following **three** get methods.

Programming language ....

`GetScore()`

Program code

....

....

....

....

`GetCategory()`

Program code

....

....

....

....

`GetPlayerID()`

Program code

....

....

....

....

[4]

(iii) The method `SetPlayerID()` asks the user to input the new player ID and reads in this value.

It checks that the length of the `PlayerID` is less than or equal to 15 characters and greater than or equal to 4 characters. If the input is valid, it sets this as the `PlayerID`, otherwise it loops until the player inputs a valid `PlayerID`.

Use suitable input and output messages.

Write **program code** for `SetPlayerID()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(iv) The method `SetScore()` checks that its `INTEGER` parameter `ScoreInput` is valid. If it is valid, it is then set as `Score`. A valid `ScoreInput` is greater than or equal to 0 and less than or equal to 150.

If the `ScoreInput` is valid, the method sets `Score` and returns `TRUE`.

If the `ScoreInput` is not valid, the method does not set `Score`, displays an error message, and it returns `FALSE`.

Write **program code** for `SetScore(ScoreInput : INTEGER)`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(v) Write **program code** for the method `SetCategory()`. Use the properties and methods in the original class definition.

Players will be placed in one of the following categories.

Category	Criteria
Advanced	Score is greater than 120
Intermediate	Score is greater than 80 and less than or equal to 120
Beginner	Score is greater than or equal to 50 and less than or equal to 80
Not Qualified	Score is less than 50

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(vi) Joanne has played the first game to place her in a category for the competition.

The procedure `CreatePlayer()` performs the following tasks.

- allows the player ID and score to be input with suitable prompts
- creates an instance of `Player` with the identifier `JoannePlayer`
- sets the score for the object
- sets the category for the object
- outputs the category for the object

Write **program code** for the `CreatePlayer()` procedure.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

(e) The programmer wants to test that the correct category is set for a player's score.

As stated in **part (d)(v)**, players will be placed in one of the following categories.

Category	**Criteria**
Advanced	Score is greater than 120
Intermediate	Score is greater than 80 and less than or equal to 120
Beginner	Score is greater than or equal to 50 and less than or equal to 80
Not Qualified	Score is less than 50

Complete the table to provide test data for each category.

Category	**Type of test data**	**Example test data**
Beginner	Normal	
	Abnormal	
	Boundary	
Intermediate	Normal	
	Abnormal	
	Boundary	
Advanced	Normal	
	Abnormal	
	Boundary	

[3]

(f) In **part (b)**, the club stored scores in a 1D array. This allows the club to sort the scores.

The following is a sorting algorithm in pseudocode.

```
NumberOfScores ← 5
FOR Item ← 1 TO NumberOfScores – 1
   InsertScore ← ArrayData[Item]
   Index ← Item – 1
   WHILE (ArrayData[Index] > InsertScore) AND (Index >= 0)
      ArrayData[Index + 1] ← ArrayData[Index]
      Index ← Index – 1
   ENDWHILE
   ArrayData[Index + 1] ← InsertScore
ENDFOR
```

(i) Give the name of this algorithm.

.... [1]

(ii) State the name of **one** other sorting algorithm.

.... [1]

(iii) Complete a dry run of the algorithm using the following trace table.

Item	NumberOfScores	InsertScore	Index	ArrayData				
				0	1	2	3	4
				99	125	121	109	115

[7]

3 Some algorithms can be written using recursion.

(a) State **two** features of recursion.

Feature 1 ....

Feature 2 ....
[2]

(b) Explain what a compiler has to do to implement recursion.

....

....

....

....

....

.... [3]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

5529018953

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 2017**

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 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.
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.

The maximum number of marks is 75.

This document consists of **14** printed pages and **2** blank pages.

DC (LK/JG) 129979/4

CAMBRIDGE International Examinations

1 Students are choosing their A Level subjects based on their IGCSE subject results.

A student can take:

- Computer Science, if they have a grade C in Maths or a grade C in Computer Science
- Maths, if they have a grade C in Maths
- Physics, if they have a grade C in Science and a grade C in Maths.

(a) Complete the decision table.

		Column							
		1	2	3	4	5	6	7	8
Conditions	Grade C in Computer Science	Y	Y	Y	Y	N	N	N	N
	Grade C in Maths	Y	Y	N	N	Y	Y	N	N
	Grade C in Science	Y	N	Y	N	Y	N	Y	N
Actions	Take Computer Science								
	Take Maths								
	Take Physics								

[4]

(b) Simplify your solution by removing redundancies.

		Column							
		S	T	U	V	W	X	Y	Z
Conditions	Grade C in Computer Science								
	Grade C in Maths								
	Grade C in Science								
Actions	Take Computer Science								
	Take Maths								
	Take Physics								

[3]

(c) Show how the columns from **part (a)** were simplified to create the columns in **part (b)**.

For example, if columns 5, 6 and 7 were simplified to create column X, then you state this in your answer.

....

....

....

....

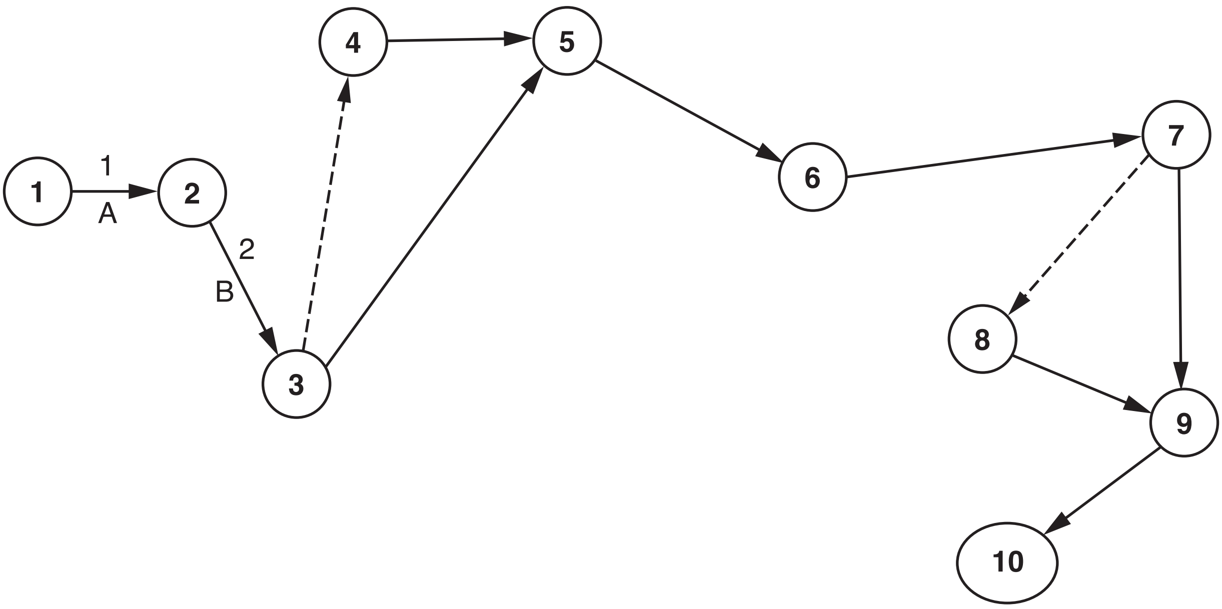
....

....[3]

2 **(a)** A project manager is planning to create a new computer game. The following table shows the activities and the estimated number of weeks to complete each activity.

Activity	Description	Weeks to complete
A	Interview end user	1
B	Produce requirements analysis	2
C	Design program structure	3
D	Design Interface	1
E	Program development	12
F	Black-box testing	2
G	Produce technical documentation	4
H	Acceptance testing	1
I	Installation	1

Complete the labelling of the Program Evaluation Review Technique (PERT) chart using the data in the table. The first two activities have been done for you.



[7]

(b) State what the dashed lines in the PERT chart represent.

....

....[1]

3 A declarative programming language is used to represent the knowledge base:

```
01      room(master_bedroom).
02      room(ensuite_bathroom).
03      room(office).
04      room(spare_bedroom).
05      room(nursery).
06      furniture(bed).
07      furniture(desk).
08      furniture(cot).
09      furniture(wardrobe).
10      furniture(computer).
11      located(bed, master_bedroom).
12      located(bed, spare_bedroom).
13      located(cot, nursery).
14      located(computer, office).
15      located(computer, master_bedroom).
```

These clauses have the following meanings:

Clause	Explanation
01	Master bedroom is a room
06	Bed is an item of furniture
11	Bed is located in the master bedroom

(a) Corridor is a room that contains a table and a lamp.

Write additional clauses to represent this information.

16 ....

17 ....

18 ....

19 ....

20 ....

[5]

(b) Using the variable `WhatItem`, the goal:

```
located(WhatItem, master_bedroom).
```

returns:

```
WhatItem = bed, computer
```

Write the result returned by the goal:

```
located(bed, WhichRoom).
```

WhichRoom = ...

....[2]

(c) **(i)** Clauses to identify rooms that are next to each other need to be stored.

The nursery is next to the master bedroom. This information is stored as:

```
21 nextTo(nursery, master_bedroom).
22 nextTo(master_bedroom, nursery).
```

Explain why both clauses are necessary.

....

....

....

....[2]

(ii) The corridor is next to the main bathroom.

Write additional clauses for this fact.

23 ....

24 ....

25 ....

[3]

(d) B can be moved into A, if B is furniture, A is a room and B is not already in A.

Write this as a rule.

canBeMovedTo(...................................... , ...)

IF ....

....[6]

4 **(a)** The array `Numbers[0 : Max]` stores numbers. An insertion sort can be used to sort these numbers into ascending order.

Complete the following **pseudocode** for the insertion sort algorithm.

```
FOR Pointer ← 1 TO (Max - 1)
   ItemToInsert ← ...........................................................................
   CurrentItem ← ...........................................................................
   WHILE (CurrentItem > 0) AND (Numbers[CurrentItem - 1] > ItemToInsert)
      Numbers[.......................................................] ← Numbers[CurrentItem - 1]
      CurrentItem ← CurrentItem - 1
   ENDWHILE
   Numbers[CurrentItem] ← .......................................................................
ENDFOR
```

[4]

(b) Identify **two** features of the array `Numbers` that would have an impact on the performance of this insertion sort algorithm.

1 ....

2 ....

[2]

5 The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		**Explanation**
Op code	**Operand**	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`LDR`	`#n`	Immediate addressing. Load the number n to IX.
`STO`	`<address>`	Store the contents of ACC at the given address.
`STX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents from ACC to this calculated address.
`ADD`	`<address>`	Add the contents of the given address to the ACC.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`JMP`	`<address>`	Jump to the given address.
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`CMP`	`#n`	Compare the contents of ACC with number n.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`LSL`	`#n`	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
`LSR`	`#n`	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
`IN`		Key in a character and store its ASCII value in ACC.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

(a) Six letters are stored, starting at the location labelled `LETTERS`. A program is needed to perform a linear search on `LETTERS` to find the letter `'x'`. The program counts the number of times `'x'` appears in `LETTERS`.

The following is the pseudocode for the program.

```
FOR COUNT ← 0 TO 5
   IF LETTERS[COUNT] = LETTERTOFIND
      THEN
         FOUND ← FOUND + 1
   ENDIF
ENDFOR
```

Write this program. Use the op codes from the given instruction set.

Label	Op code	Operand	Comment
START:	LDR	#0	// initialise Index Register
LOOP:			// load LETTERS
			// is LETTERS = LETTERTOFIND ?
			// if not, go to NOTFOUND
			// increment FOUND
NOTFOUND:			// increment COUNT
			// is COUNT = 6 ?
			// if yes, end
			// increment Index Register
			// go back to beginning of loop
ENDP:	END		// end program
LETTERTOFIND:	'x'		
LETTERS:	'd'		
	'u'		
	'p'		
	'l'		
	'e'		
	'x'		
COUNT:	0		
FOUND:	0		

[10]

(b) Six values are stored, starting at the location `VALUES`. A program is needed to divide each of the values by 8 and store them back in their original location.

Write this program. Use the op codes from the instruction set on the next page.

Label	Op code	Operand	Comment
START:			// initialise the Index Register
			// load the value from VALUES
			// divide by 8
			// store the new value in VALUES
			// increment the Index Register
			// increment REPS
			// is REPS = 6 ?
			// repeat for next value
	END		
REPS:	0		
VALUES:	22		
	13		
	5		
	46		
	12		
	33		

[10]

Instruction		Explanation
Op code	Operand	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`LDR`	`#n`	Immediate addressing. Load the number n to IX.
`STO`	`<address>`	Store the contents of ACC at the given address.
`STX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents from ACC to this calculated address.
`ADD`	`<address>`	Add the contents of the given address to the ACC.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`JMP`	`<address>`	Jump to the given address.
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`CMP`	`#n`	Compare the contents of ACC with number n.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`LSL`	`#n`	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
`LSR`	`#n`	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
`IN`		Key in a character and store its ASCII value in ACC.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

6 A bank has a range of customer accounts, which includes current accounts and savings accounts.

All accounts have:

- an account number
- a balance (amount of money in an account).

A current account has a level (bronze, silver or gold). A monthly fee ($) is taken from each account.

Savings account customers pay a regular amount ($) into their account. The payment interval is a number of weeks (for example, 4).

An object-oriented program will be written to process data about the accounts.

(a) Complete the class diagram.

Account
`AccountNumber : STRING` `Balance : CURRENCY`
`Constructor()` `GetAccountNumber()` `GetBalance()` `SetAccountNumber()` `SetBalance()`

CurrentAccount	**SavingsAccount**
....	....
`Constructor()`	.... `GetPaymentInterval()` `SetPaymentInterval()`

[3]

(b) Write **program code** to declare the `Account` class.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(c) Write **program code** to declare the `SavingsAccount` class. Do not write any get or set methods.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME										
CENTRE NUMBER						CANDIDATE NUMBER				

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2018**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **15** printed pages and **1** blank page.

DC (LEG) 164471

CAMBRIDGE
International Examinations

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- Two's complement form for both mantissa and exponent.

(a) Find the denary value for the following binary floating-point number.

Mantissa

1	0	1	1	1	0	0	1	1	0	1	0

Exponent

0	1	0	1

Show your working.

Working ....

....

....

....

....

Answer ....
[3]

(b) Calculate the normalised floating-point representation of 5.25 in this system. Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(c) The size of the mantissa is decreased and the size of the exponent is increased.

State how this affects the range and precision of the numbers that the computer system can represent.

....

....

....

....[2]

2 A programmer uses non-composite and composite data types to create a program.

(a) Define the term **non-composite data type**.

....

....[1]

(b) Describe **two** different non-composite data types.

Data type 1 ....

Description ....

....

....

Data type 2 ....

Description ....

....

....
[4]

(c) Define the term **composite data type**.

....

....[1]

(d) Describe **two** different composite data types.

Data type 1 ....

Description ....

....

....

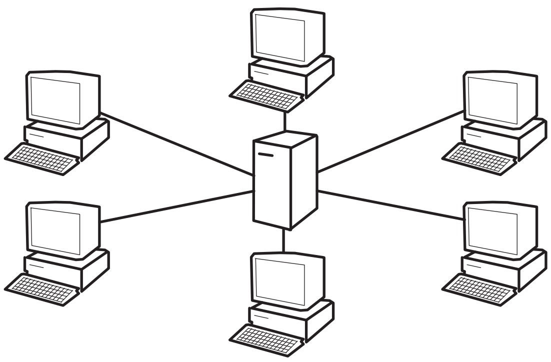
Data type 2 ....

Description ....

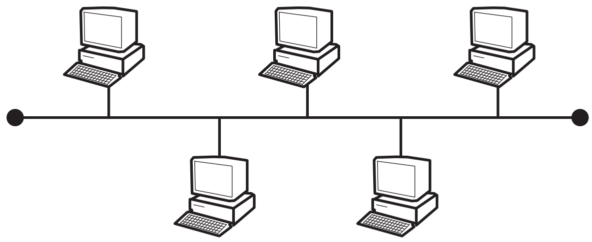
....

.... [4]

3 Star and bus are two types of topology that can be used in a Local Area Network (LAN).



Star topology



Bus topology

(a) (i) State **one** benefit and **one** drawback of the star topology.

Benefit ....

....

Drawback ....

.... [2]

(ii) State **one** benefit and **one** drawback of the bus topology.

Benefit ....

....

Drawback ....

.... [2]

(b) The sequence of steps 1 to 7 describes what happens when the LAN transmits data from Computer X to Computer Y using circuit switching. Four statements (4 to 7) are missing from the sequence.

A	Computer X sends the data.
B	The sender signals node to deallocate resources.
C	Computer Y sends a receipt signal.
D	If available, Computer X sets up path between nodes.

Write **one** letter (**A** to **D**) in the appropriate space to complete the sequence.

1 Computer X sends a connection request to Computer Y.

2 Computer Y sends ready or busy signal.

3 If busy, Computer X waits and then resends the connection request to Computer Y.

4 ....

5 ....

6 ....

7 ....

[3]

(c) **(i)** Protocols are essential for successful transmission of data over a network. The TCP/IP protocol suite operates on many layers.

State the appropriate layer for each protocol in the following table.

Protocol	**Layer**
TCP	
IP	
SMTP	

[3]

(ii) Peer-to-peer (P2P) file sharing uses the BitTorrent protocol.

Explain how the BitTorrent protocol allows files to be shared.

....

....

....

....

....

....[3]

Question 4 begins on the next page.

4 **(a)** A Boolean expression produces the following truth table.

INPUT			OUTPUT
A	**B**	**C**	**X**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

(i) Write the Boolean expression for the truth table as a sum-of-products.

X = ...[2]

(ii) Complete the Karnaugh Map (K-map) for the truth table in **part (a)(i)**.

AB

		00	**01**	**11**	**10**
C	**0**				
	1				

[1]

The K-map can be used to simplify the function in **part (a)(i)**.

(iii) Draw loop(s) around appropriate group(s) of 1s to produce an optimal sum-of-products for the table in **part (a)(ii)**. [2]

(iv) Write the simplified sum-of-products expression for your answer to **part (a)(iii)**.

X = ...[2]

(b) A logic circuit with four inputs produces the following truth table.

INPUT				**OUTPUT**
A	**B**	**C**	**D**	**X**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

(i) Complete the K-map that corresponds to the truth table.

AB

CD					

[4]

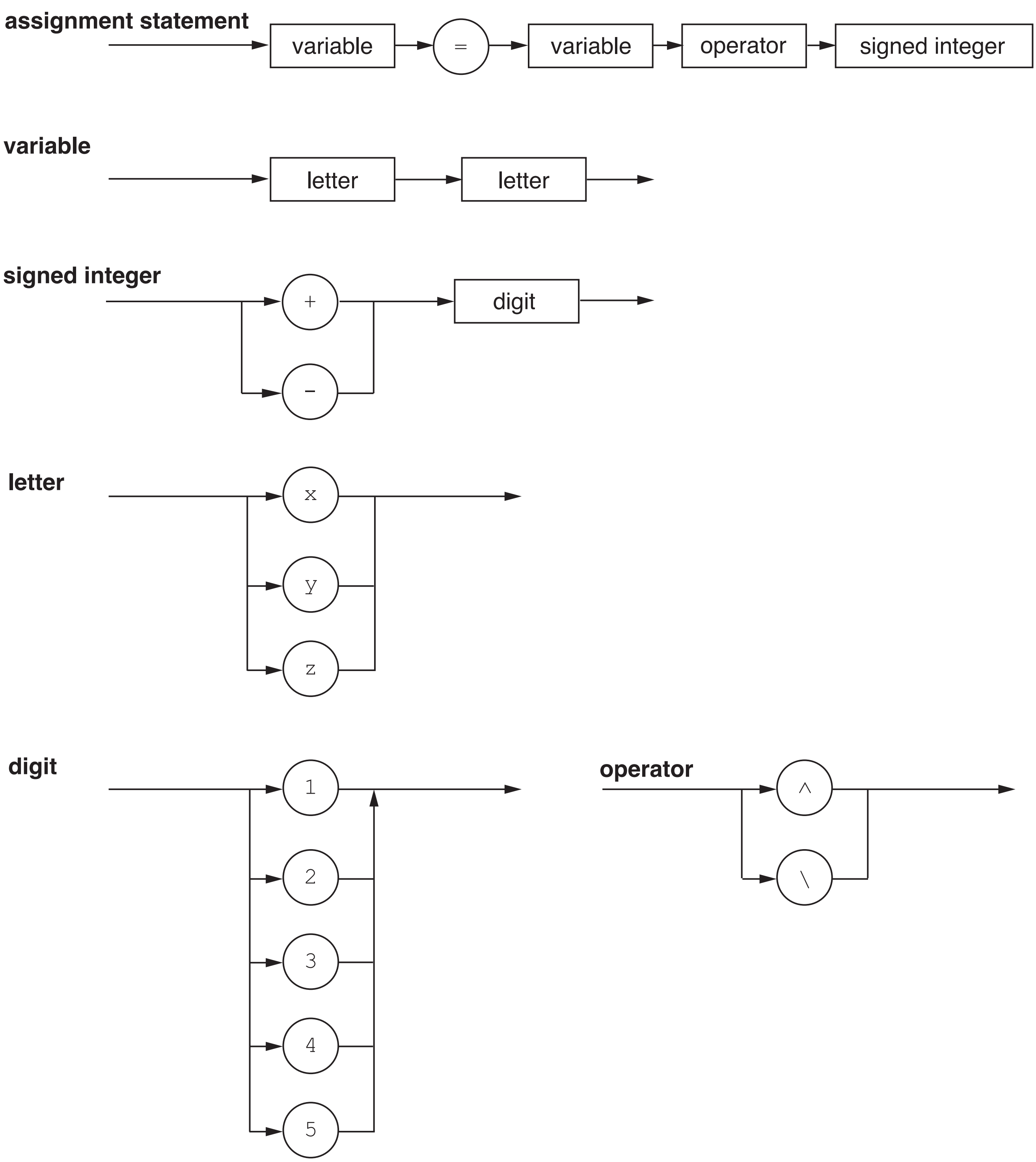
(ii) Draw loop(s) around appropriate group(s) of 1s to produce an optimal sum-of-products for the table in **part (b)(i)**. [2]

(iii) Write the simplified sum-of-products expression for your answer to **part (b)(ii)**.

X = ...[2]

5 The following syntax diagrams show the syntax of:

- an assignment statement
- a variable
- a signed integer
- a letter
- a digit
- an operator



(a) The following assignment statements are invalid.

Give the reason in each case.

(i) `xy = xy ^ c4`

Reason ....

....[1]

(ii) `zy = zy \ 10`

Reason ....

....[1]

(iii) `yy := xz ^ - 6`

Reason ....

....[1]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams on the opposite page.

`<assignment statement> ::=`

....

`<variable> ::=`

....

`<signed integer> ::=`

....

`<operator> ::=`

....

[4]

(c) Rewrite the BNF rule for a variable so that it can be any number of letters.

`<variable> ::=`

....[2]

6 A company specialises in educational software.

(a) The company is concerned that malware might disrupt their business.

(i) Add appropriate descriptions and terms in the table.

	Description	Term
A	Redirection to a bogus website that appears to be legitimate to gain confidential data.	....
B	Use email to attempt to gain a user's confidential data.	....
C	....	Spyware
D	....	Worm

[4]

(ii) A member of staff is using the Internet to carry out research. They are worried about the threat from terms **A** and **B**.

Identify **one** solution to the each of the threats.

Term **A** ....

....

Term **B** ....

....

[2]

(b) A customer downloads a new educational software package from the company.

Explain how the customer's and the company's computers use a hashing algorithm to assure the customer that:

- the software has come from the company (is authentic) and
- no one has altered it.

....

....

....

....

....

....

....

....[4]

7 A museum stores antique items that need to be kept at constant temperature.

The museum is not sure about the actual temperatures. The museum installs some equipment. This records the temperatures every hour and ensures the temperature stays within a set range.

(a) Identify the type of system described.

....[1]

(b) The system has a temperature sensor.

Identify **two** other items of hardware that the museum can use for the type of system identified.

Describe the purpose of each item.

Item 1 ....

Purpose ....

....

Item 2 ....

Purpose ....

....
[4]

(c) The equipment records the temperature in all seven rooms in the museum.

Each recording is stored as two successive bytes in memory. The format is as shown.

Temperature

Byte 1

Room

7	6	5	4	3	2	1	0

Byte 2

The room is indicated by the setting of one of the bits in **Byte 2** to 1. For example, room 7 is indicated by setting bit 7 to 1.

Bit 0 of **Byte 2** is a flag:

- The flag's initial value is zero.
- When the reading has been processed, the flag's value is set to 1.

Byte 1 contains the temperature reading as an unsigned integer.

One reading returns the following binary data.

Temperature

1	0	1	1	0	0	1	1

Byte 1

Room

7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	1

Byte 2

(i) Analyse the data contained in the two bytes.

....

....

....

....

....[3]

(ii) The system receives a temperature reading of 238 from room number 4.

Complete the bytes to show the two bytes for this recording. The reading has not yet been processed.

Byte 1

7	6	5	4	3	2	1	0

Byte 2

[2]

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

0586807379

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2017**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

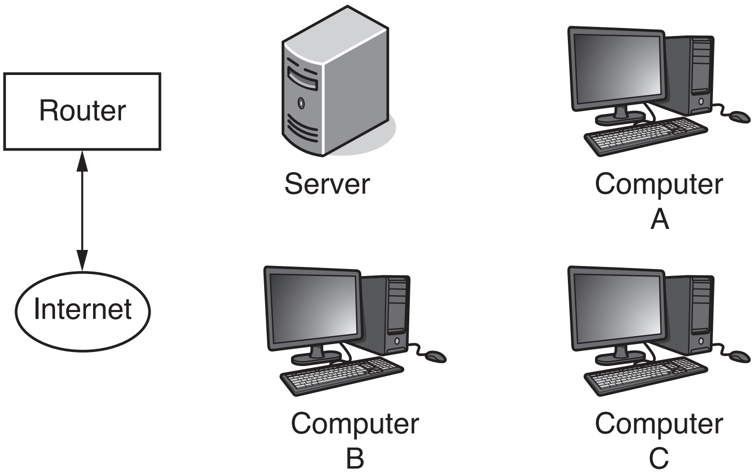
This document consists of **15** printed pages and **1** blank page.

DC (ST/FC) 135014/3

[Turn over

1 A Local Area Network (LAN) consists of three computers, one server and a router connected to the Internet. The LAN uses a bus topology.

(a) Complete the following diagram to show how the computers, the server and the router could be connected.



[2]

(b) There are four statements in the following table. For each statement, place a tick (✓) in the appropriate column to indicate whether it is true or false.

Statement	True	False
The server can send packets to Computer B and the router at the same time.		
Computer C uses the IP address of a web server to send a request for a web page on the web server.		
Computer B can read a packet sent from Computer A to Computer C.		
The server can read all incoming packets from the Internet.		

[4]

(c) The user on Computer A and the user on Computer B are both using the Internet at the same time. On a few occasions, Computer A and Computer B start transmitting packets to the router at exactly the same time. This causes a problem called a collision.

(i) Explain what is meant by a **collision** in this context.

....

....

....

....[2]

(ii) As a result of the collision, both Computer A and Computer B stop transmitting.

Computer A must carry out a number of steps to ensure the successful transmission of its packet.

Give **two** of the steps.

Step 1 ....

Step 2 ....
[2]

(d) The LAN topology is redesigned.

(i) Describe the changes that could be made to the LAN topology to overcome the problem identified in **part (c)**.

....

....

....

....[2]

(ii) Explain how the redesign has overcome the problem.

....

....

....

....[2]

2 **(a)** The following diagram shows four descriptions and two types of processor.

Draw lines to connect each description to the appropriate type of processor.

Description	**Type of processor**
It has a simplified set of instructions.	
Emphasis is on the hardware rather than the software.	CISC
It makes extensive use of general purpose registers.	RISC
Many instruction formats are available.	

[4]

(b) In a RISC processor, instructions are processed using pipelining.

(i) Explain what is meant by **pipelining**.

....

....

....

....[2]

(ii) The following table shows the five stages that occur when instructions are fetched and executed. The table also shows a number of time intervals.

Two instructions, D followed by E, are fetched and executed. The ‘E’ in the incomplete table shows that instruction E has been fetched in time interval 2.

Complete each row of the table.

Stage	**Time interval** 1	2	3	4	5	6	7	8
Fetch instruction		E						
Read registers and decode instruction								
Execute instruction								
Access operand in memory								
Write result to register								

[3]

(c) The instruction set for a RISC processor that allows pipelining includes the following instruction.

Instruction		Explanation
Op code	**Operands**	
`ADD`	`<dest>, <op1>, <op2>`	Add the integers in registers `op1` and `op2`. Place the result in register `dest`.

A program contains the following three instructions.

```
ADD r3, r2, r1

ADD r5, r4, r3

ADD r10, r9, r8
```

(i) Explain why pipelining fails for the first two instructions.

....

....

....

....[2]

(ii) The instructions were produced by a compiler after translation of a high-level language program.

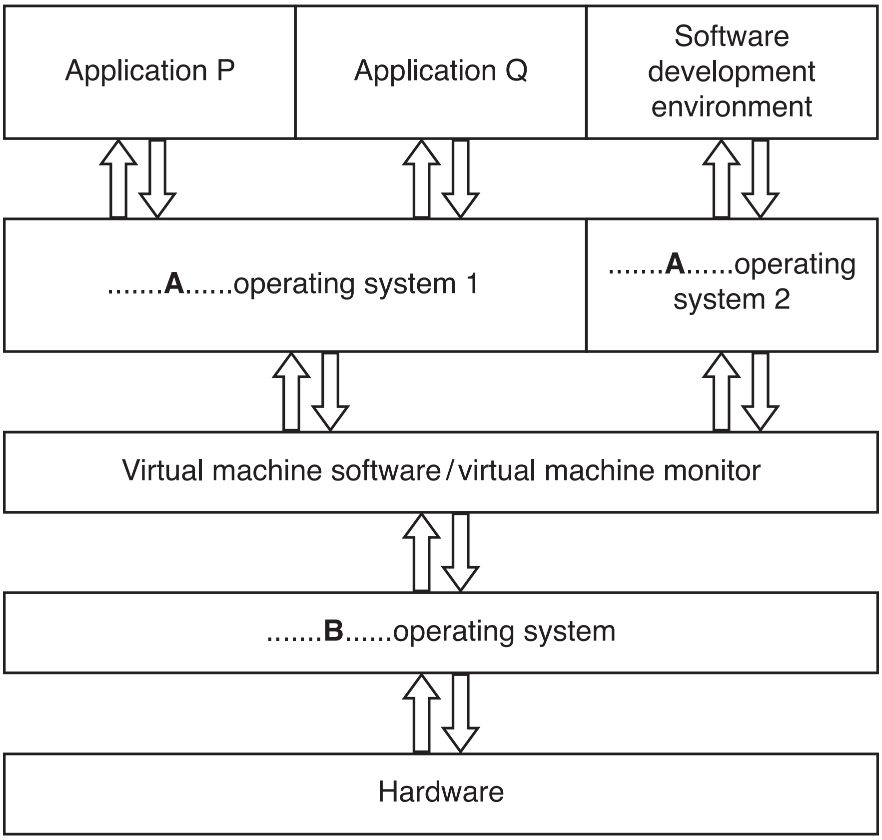
The compiler is not capable of code optimisation.

State how the code from the compiler could have been optimised to overcome the problem in **part (c)(i)**.

....

....[1]

3 **(a)** This diagram shows how applications P, Q and a software development environment can be run on a virtual machine system.



(i) State the operating systems labelled **A** and **B** in the diagram.

A ....

B ....

[2]

(ii) Application P is executing and requests data from a file.

Describe what happens after ....**A**......operating system 1 has received the data request from the application.

....

....

....

....

....

....[3]

(b) A software development company uses virtual machines to produce software.

(i) State **one** benefit to the company.

....

....[1]

(ii) Explain **two** limitations of this approach.

Limitation 1 ....

....

....

....

Limitation 2 ....

....

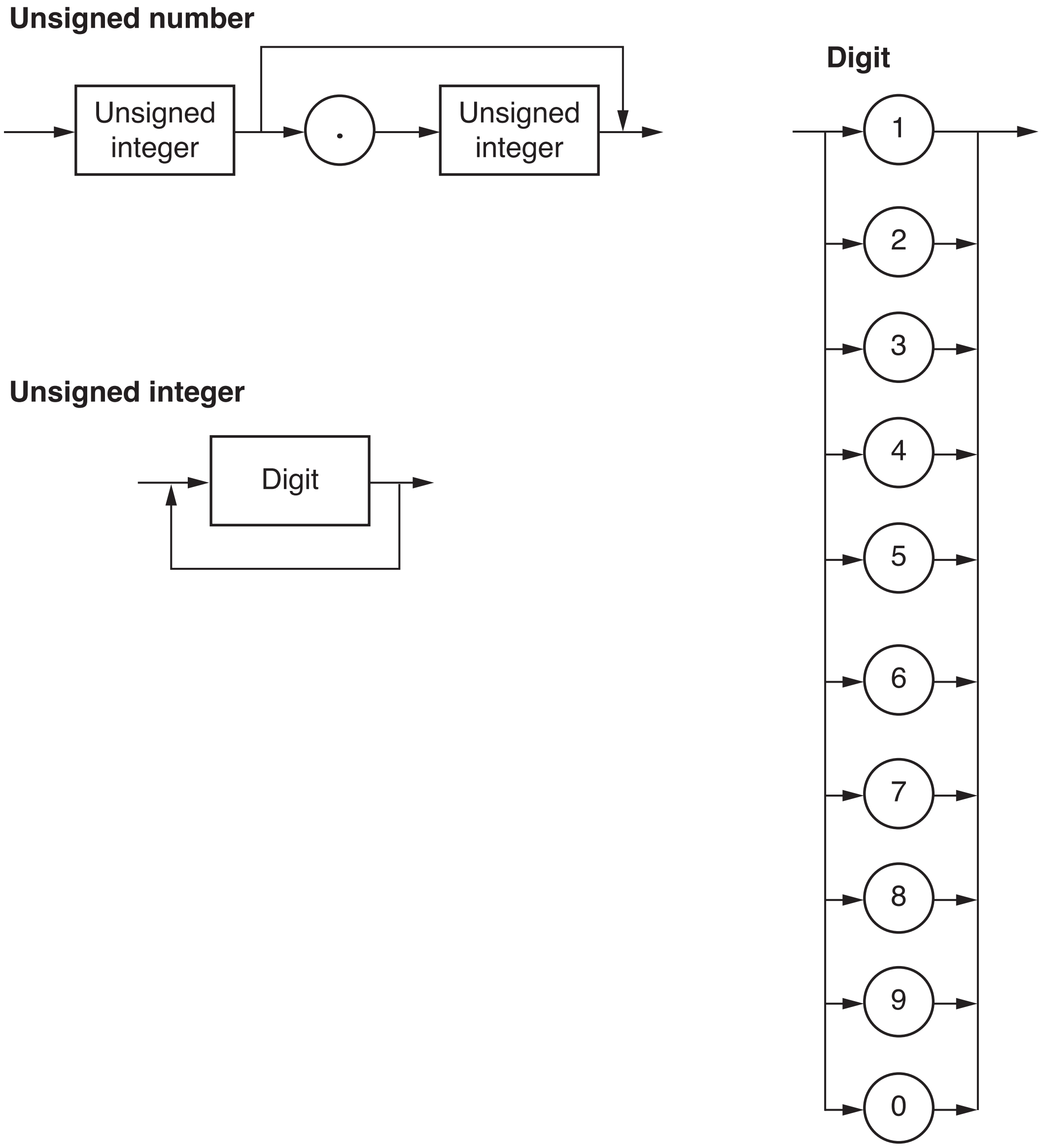
....

....

[4]

4 The following syntax diagrams for a particular programming language show the syntax of:

- an unsigned number
- an unsigned integer
- a digit.



(a) **(i)** Explain why 32 is a valid unsigned integer.

....

....

....

....[2]

(ii) Explain why 32.5 is a valid unsigned number.

....

....

....

....[2]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.

`<unsigned_number> ::=` ....

....

`<unsigned_integer> ::=` ....

....

`<digit> ::=` ....

....
[5]

The format of an unsigned number is amended to include numbers with possible exponents.

If an unsigned number has an exponent, then the exponent part:

- will start with an 'E'
- be followed by an optional '+' or '–' sign
- and be completed by an unsigned integer.

Examples of unsigned numbers with exponents include: 3E2, 3E+3, 3E–32, 3.45E–2

(c) **(i)** Redraw the syntax diagram for unsigned number to include numbers that might have exponents.

[4]

(ii) Use your syntax diagram from **part (c)(i)** to write the BNF for an unsigned number to include numbers with exponents.

<unsigned_number> ::= ....

....

....

....

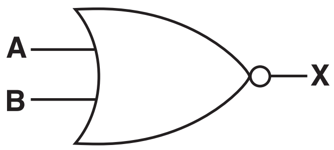
....

....

....

....[4]

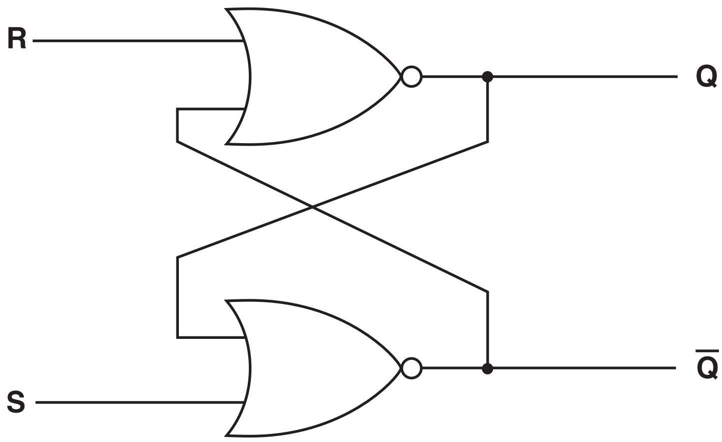
5 **(a)** Complete the truth table for this NOR gate:



A	B	X
0	0	
0	1	
1	0	
1	1	

[1]

A SR flip-flop is constructed using two NOR gates.



(b) Complete the truth table for the SR flip-flop:

	S	R	Q	$\overline{Q}$
Initially	1	0	1	0
S changed to 0	0	0		
R changed to 1	0	1		
R changed to 0	0	0		
S and R changed to 1	1	1		

[4]

Another type of flip-flop is the JK flip-flop. The JK flip-flop is an improvement on the SR flip-flop.

(c) **(i)** The JK flip-flop has three inputs. Two of the inputs are the Set (J) and the Reset (K).

State the third input.

....[1]

(ii) There are **two** problems with the SR flip-flop that the JK flip-flop overcomes.

State each problem and state why it does not occur for the JK flip-flop.

Problem 1 ....

....

....

....

Problem 2 ....

....

....

....

[4]

6 The environment in a very large greenhouse is managed by a computer system. The system uses a number of different sensors that include temperature sensors. In addition, the system controls a number of heaters, windows and sprinklers.

(a) State **one** other type of sensor that could be used with this system.

Justify your choice.

Sensor ....

Justification ....

.... [2]

(b) Describe why feedback is important in this system.

....

....

....

....

....

....[3]

(c) **(i)** The system makes use of a number of parameters. These parameters are used in the code that runs the system.

State **one** of the parameters used in controlling the temperature in the greenhouse.

....[1]

(ii) Explain how the parameter identified in **part (c)(i)** is used in the feedback process.

....

....

....

....[2]

(d) There are eight temperature sensors numbered 1 to 8. Readings from these sensors are stored in four 16-bit memory locations. The memory locations have addresses from 4000 to 4003. Each memory location stores two sensor readings as two unsigned binary integers.

Sensor 1 reading is stored in bits 8 to 15 of address 4000; Sensor 2 reading is stored in bits 0 to 7 of address 4000 and so on. The diagram shows that the current sensor 1 reading has a value of 97.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
4000	0	1	1	0	0	0	0	1	0	0	1	1	1	0	0	1
4001	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0
4002	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	1
4003	1	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1

(i) Give the denary value of the current reading for Sensor 5.

....

....

....

....[1]

(ii) The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC).

Instruction		**Explanation**
Op code	**Operand**	
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`AND`	`#n`	Bitwise AND operation of the contents of ACC with the operand.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`#n`	Bitwise XOR operation of the contents of ACC with the operand.
`XOR`	`<address>`	Bitwise XOR operation of the contents of ACC with the contents of <address>.
`OR`	`#n`	Bitwise OR operation of the contents of ACC with the operand.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>. <address> can be an absolute address or a symbolic address.
`LSL`	`#n`	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
`LSR`	`#n`	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.

The reading for Sensor 5 is used in a calculation. The calculation is carried out by two assembly language instructions.

The first instruction loads the contents of the 16-bit location that contains the value for Sensor 5.

The second instruction moves the bits in Sensor 5 so that the 16-bit value is the value of Sensor 5.

Complete the two instructions in the following code. Use the instruction set provided.

```
LDD ........................... // load the contents of the 16-bit location
                                   containing the value for Sensor 5 into the
                                   Accumulator

.................................. // move the bits in the Accumulator so that the
                                   Accumulator stores the value of Sensor 5 as an
                                   unsigned 16-bit binary integer
```

[3]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

9344667390

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 2021**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **20** pages. Any blank pages are indicated.

1 Sandy is writing a program to process data in a stack. The stack is implemented as a 1D array, `DataStack`, which has up to 100 elements.

The function `Push(Value)` stores `Value` on the stack and returns `TRUE` if `Value` was added to the stack, or `FALSE` if the stack is full.

The function `Pop()` returns the item at the top of the stack, or returns `-1` if the stack is empty.

`DataStack` and `TopPointer` are declared as global.

(a) Show the state of `DataStack` and its pointer after the following functions are executed on the current contents.

```
Pop()

Pop()

Push(19)

Pop()

Push(50)
```

TopPointer | 3 |

Index	Data
[7]	
[6]	
[5]	
[4]	
[3]	8
[2]	6
[1]	20
[0]	10

[2]

(b) Write **program code** for the function `Pop()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(c) Sandy has also used a queue in her program.

Describe the ways in which a queue differs from a stack.

....

....

....

....

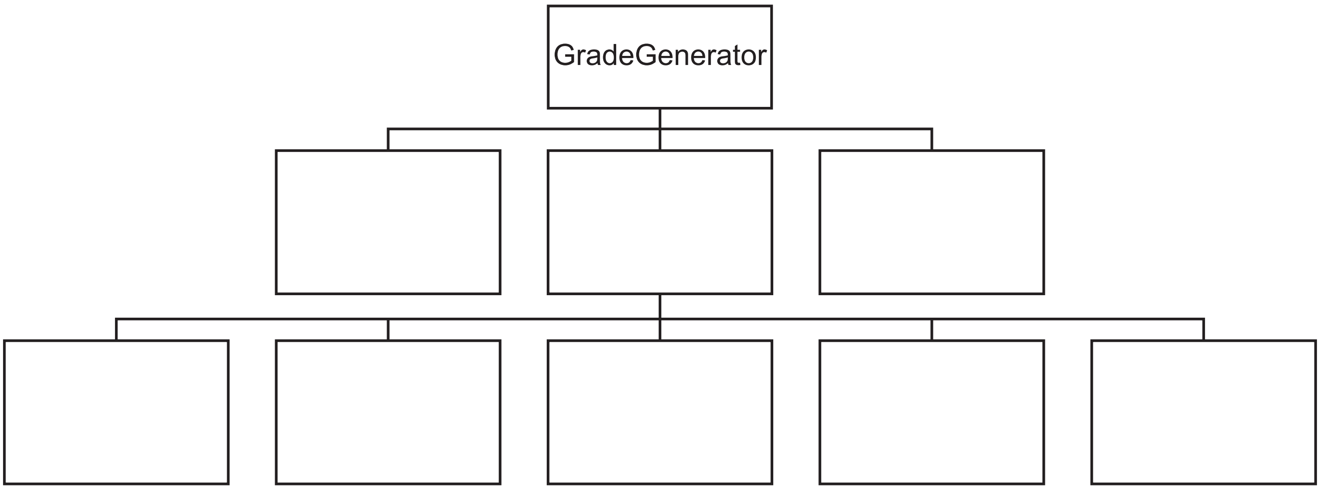
....

.... [2]

2 A grade generator program takes the mark a student obtained in a test as input.

The program calculates and outputs the grade that matches the mark. The grade is either A, B, C, D or U.

Complete the following JSP structure diagram for the grade generator program.



[4]

3 The following pseudocode algorithm performs a binary search on the sorted array `ThisArray`.

The algorithm returns either the location of `SearchItem` in the array, or `−1` if `SearchItem` is not in the array.

The function `DIV` returns the integer value of the division, for example, `11 DIV 2` returns `5`.

Complete the algorithm by writing the missing pseudocode statements.

```
FUNCTION BinarySearch(ThisArray[], LowerBound, UpperBound,
                      SearchItem : INTEGER) RETURNS INTEGER

DECLARE Flag : BOOLEAN

DECLARE Mid : INTEGER

Flag ← -2

WHILE Flag <> -1

   Mid ← LowerBound + ((UpperBound – LowerBound) DIV 2)

   IF ................................................ < ................................................

      THEN

         RETURN ................................................

      ELSE

         IF ThisArray[Mid] > SearchItem

            THEN

              UpperBound ← Mid ................................................

            ELSE

               IF ThisArray[Mid] < SearchItem

                  THEN

                     LowerBound ← Mid ................................................

                  ELSE

                     RETURN ................................................

               ENDIF

         ENDIF

   ENDIF

ENDWHILE

ENDFUNCTION
```

[6]

4 Teachers in a school may work on Mondays, Tuesdays and Wednesdays. There are three time slots on each day: time slot 1, time slot 2 and time slot 3.

A teacher is either busy or free.

The school is using a declarative language to write a program to record which teachers are busy in each time slot on each day.

The following knowledge base is used:

```
01 teacher(james).
02 teacher(jill).
03 teacher(karl).
04 teacher(kira).
05 day(monday).
06 day(tuesday).
07 day(wednesday).
08 timeSlot(1).
09 timeSlot(2).
10 timeSlot(3).
11 busy(james, monday, 1).
12 busy(james, tuesday, 2).
13 busy(karl, monday, 1).
14 busy(kira, wednesday, 3).
```

These clauses have the following meaning:

Clause	Explanation
01	James is a teacher
05	Monday is a day
08	1 is a time slot
11	James is busy in time slot 1 on Monday

(a) More facts need to be included.

Fred is a teacher who is busy in time slot 1 on Tuesday.

Write additional clauses for these facts.

15 ....

16 ....

[2]

(b) Additional clauses are needed to identify whether Jill is busy in time slot 1 on Monday, Tuesday, or Wednesday.

Write these additional clauses.

17 ....

18 ....

19 ....
[2]

(c) Write a goal, using the variable `X`, to find all the teachers who are busy in time slot 3 on Monday.

....

.... [1]

(d) Write a rule to find whether a teacher `X` is free in a specific time slot `Y` on day `Z`.

```
IsTeacherFree(X, Z, Y)

IF
```

....

....

....

.... [4]

5 The recursive algorithm for the `Recursion()` function is defined in pseudocode as follows:

```
FUNCTION Recursion(A, B : INTEGER) RETURNS INTEGER
   IF A <= 100
      THEN
         RETURN 1
      ELSE
         IF A > B
            THEN
               RETURN 5 + Recursion(A - 1, B)
            ELSE
               RETURN 10 + Recursion(A – 10, B)
         ENDIF
   ENDIF
ENDFUNCTION
```

(a) The function is called with the following pseudocode statement:

```
OUTPUT Recursion(104, 102)
```

Dry run the function and complete the trace table. Give the output the program will produce.

Trace table:

Function call	A	B	Return value

Output = ...

Working ....

....

....

....

....

....
[4]

(b) Rewrite the function `Recursion()` in **pseudocode**, using an **iterative algorithm**.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

6 Kobi is writing an application that uses a record structure to store data.

(a) **(i)** Describe what is meant by a **record structure**.

....

....

....

.... [2]

(ii) The record structure stores the unique ID number (a whole number), first name and last name of a customer.

Write a **pseudocode** declaration for the record structure `CustomerData`.

....

....

....

....

....

....

.... [2]

(b) Kobi's application stores the records in a random access file.

The function `StoreRecord()`:

- takes a customer record as a parameter
- uses the function `CustomerHash()` to calculate and return the hash value for its parameter
- stores the customer record in the returned hash value address.

Assume there are no collisions.

Complete the following pseudocode algorithm to write a new record to the random access file.

```
PROCEDURE  StoreRecord(NewData : ..................................................)

   HashValue ← CustomerHash(NewData.CustomerID)

   Filename ← "CustomerRecords.dat"

   OPENFILE Filename FOR ..................................................

   SEEK Filename, ..................................................

   PUTRECORD Filename, ..................................................

   .................................................. Filename

ENDPROCEDURE
```

[5]

(c) Identify **two** typical features of a debugger **and** describe how Kobi could use each one during the development of the application.

Feature 1 ....

....

....

....

Feature 2 ....

....

....

....
[4]

(d) Give **one** benefit and **one** drawback of Kobi using a program generator whilst developing his application.

Benefit ....

....

Drawback ....

....
[2]

7 Sonya is writing a computer program that requires a user input. The user should input an integer between 1 and 100. Sonya wants to use exception handling.

(a) Explain the reasons why Sonya should use exception handling in her program.

....

....

....

.... [2]

(b) Write **program code** to read in the number from the user and raise an exception if the data is not valid.

Programming language ....

Program code

....

....

....

....

....

....

.... [3]

(c) Give **two other** examples of where exception handling can be used in a program.

1 ....

2 ....

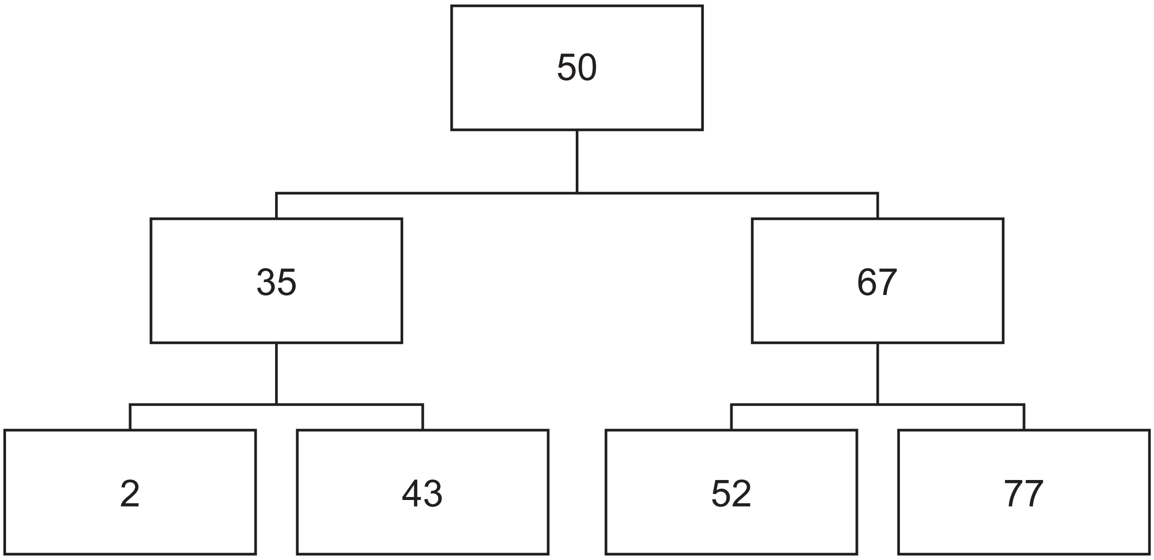
[2]

8 Data entered into a computer is stored in an ordered binary tree.

The binary tree is stored in a 2D array, `BinaryTree`.

The first element of the array is index 0.

(a) The current contents of the binary tree are:



Complete the `LeftPointer` and `RightPointer` values in the following table for the binary tree shown.

A null pointer is represented by `−1`.

RootNode | 0 |

Index	LeftPointer	Data	RightPointer
[0]		50	
[1]		67	
[2]		77	
[3]		35	
[4]		2	
[5]		43	
[6]		52	
[7]			
[8]			
[9]			
[10]			

[2]

(b) A post-order tree traversal outputs the left node, then the right node, then the root node.

In the tree given in **part (a)**, the post-order tree traversal would output:

```
2      43      35      52      77      67      50
```

Complete the following recursive pseudocode algorithm `PostOrder()`.

```
PROCEDURE PostOrder(.................................................... : INTEGER)

   IF BinaryTree[RootNode, 0] <> -1

     THEN

       .................................................... (BinaryTree[RootNode, ....................................................])

   ENDIF

   IF BinaryTree[RootNode, 2] <> -1

     THEN

       .................................................... (BinaryTree[RootNode, 2])

   ENDIF

   OUTPUT BinaryTree[RootNode, ....................................................]

ENDPROCEDURE
```

[5]

BLANK PAGE

9 A program uses a hashing algorithm to store data in the global array, `StoredData`.

The first element of the array is index 0. The array has 10 000 integer elements.

(a) Write a **pseudocode** declaration for the array `StoredData` **and** initialise each element to `−1`.

....

....

....

.... [3]

(b) The hashing algorithm calculates the remainder after dividing the data by 1000, and then adds 6 to it.

The function `AddItem()` takes the data as a parameter. It calculates the index to store the data using the hashing algorithm.

If there is a collision, the function:

- checks the next index until it finds an index that does not have data in it
- continues to search from the start of the array, if it reaches the end of the array.

The function returns `TRUE` if the item was successfully added, and `FALSE` if the array is full.

Write **program code** for the function `AddItem()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

[7]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6948841832

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2016**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

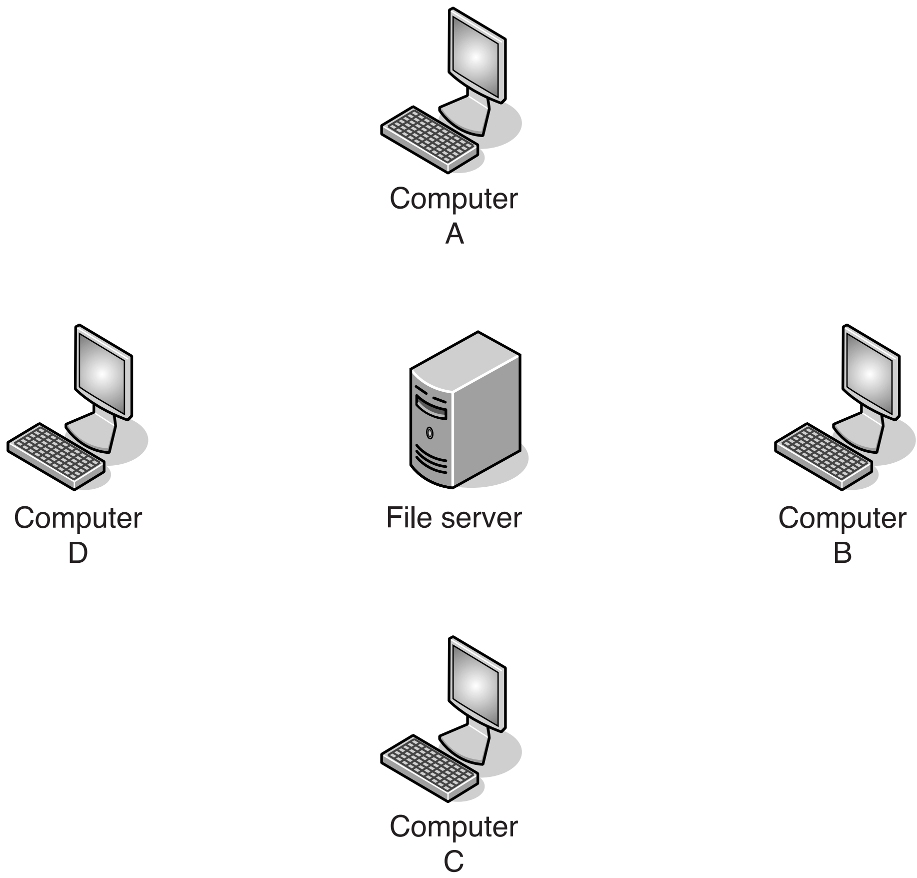
This document consists of **15** printed pages and **1** blank page.

DC (ST/AR) 126657

CAMBRIDGE International Examinations

1 A Local Area Network (LAN) consists of four computers and one server. The LAN uses a bus topology.

(a) Complete the diagram below to show how the computers and the File server could be connected.



[2]

(b) Computer C sends a data packet to Computer A.

Three statements are given below.

Tick (✓) to show whether each statement is true or false.

Statement	True	False
Computer C uses the IP address of Computer A to indicate that the packet is for Computer A.		
Computer B can read the packet sent from Computer C to Computer A.		
The File server routes the packet to Computer A.		

[3]

(c) Computer A starts transmitting a packet to Computer C. At exactly the same time, the File server starts transmitting a packet to Computer D. This causes a problem.

(i) State the name given to this problem.

....

....[1]

(ii) Give **three** steps taken by both Computer A and the File server to allow them to transmit their packets successfully.

Step 1 ....

....

Step 2 ....

....

Step 3 ....

....[3]

(d) Adding a switch to the LAN changes its topology. Explain how the use of a switch removes the problem identified in **part (c)(i)**.

....

....

....

....

....

....

....

....[4]

2 Digital certificates are used in Internet communications. A Certificate Authority (CA) is responsible for issuing digital certificates.

(a) Name **three** data items present in a digital certificate.

1 ....

2 ....

3 ....[3]

(b) The method of issuing a digital certificate is as follows:

1 A user starts an application for a digital certificate using their computer. On this computer a key pair is generated. This key pair consists of a public key and an associated private key.

2 The user submits the application to the CA. The generated .... **(i)** .... key and other application data are sent. The key and data are encrypted using the CA's .... **(ii)** .... key.

3 The CA creates a digital document containing all necessary data items and signs it using the CA's .... **(iii)** .... key.

4 The CA sends the digital certificate to the individual.

In the above method there are three missing words. Each missing word is either 'public' or 'private'.

State the correct word. Justify your choice.

(i) ....

Justification ....

....[2]

(ii) ....

Justification ....

....[2]

(iii) ....

Justification ....

....[2]

(c) Alexa sends an email to Beena.

Alexa's email program:

- produces a message digest (hash)
- uses Alexa's private key to encrypt the message digest
- adds the encrypted message digest to the plain text of her message
- encrypts the whole message with Beena's public key
- sends the encrypted message with a copy of Alexa's digital certificate

Beena's email program decrypts the encrypted message using her private key.

(i) State the name given to the encrypted message digest.

....[1]

(ii) Explain how Beena can be sure that she has received a message that is authentic (not corrupted or tampered with) and that it came from Alexa.

....

....

....

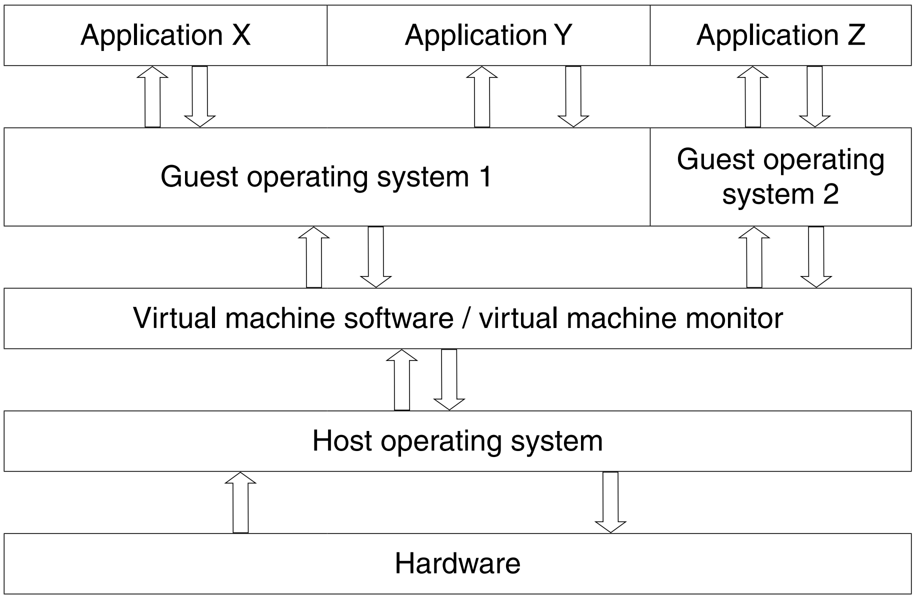
....[2]

(iii) Name **two** uses where encrypted message digests are advisable.

1 ....

2 ....[2]

3 **(a)** The following diagram shows how applications X, Y and Z can run on a virtual machine system.



(i) The virtual machine software undertakes many tasks.

Describe **two** of these tasks.

Task 1 ....

....

Task 2 ....

....[2]

(ii) Explain the difference between a **guest operating system** and a **host operating system**.

....

....

....

....[2]

(b) A company uses a computer as a web server. The manufacturer will no longer support the computer's operating system (OS) in six months' time. The company will then need to decide on a replacement OS.

The company is also considering changing the web server software when the OS is changed.

Whenever any changes are made, it is important that the web server service is not disrupted.

In developing these changes, the company could use virtual machines.

(i) Describe **two** possible uses of virtual machines by the company.

Use 1 ....

....

....

....

Use 2 ....

....

....

....[4]

The web server often has to handle many simultaneous requests.

(ii) The company uses a virtual machine to test possible solutions to the changes that they will need to make.

Explain **one** limitation of this approach.

....

....

....

....[2]

4 **(a)** Three file organisation methods and two file access methods are shown below.

Draw lines to link each file organisation method to its appropriate file access method or methods.

File organisation method	**File access method**
serial	direct
sequential	sequential
random	

[4]

(b) A bank has a very large number of customers. The bank stores data for each customer. This includes:

- unique customer number
- personal data (name, address, telephone number)
- transactions

The bank computer system makes use of three files:

- A – a file that stores customer personal data. This file is used at the end of each month for the production of the monthly statement.
- B – a file that stores encrypted personal identification numbers (PINs) for customer bank cards. This file is accessed when the customer attempts to withdraw cash at a cash machine (ATM).
- C – a file that stores all customer transaction records for the current month. Every time the customer makes a transaction, a new record is created.

For each of the files A, B and C, state an appropriate method of organisation. Justify your choice.

(i) File A organisation ....

Justification ....

....

....

....[3]

(ii) File B organisation ....

Justification ....

....

....

....[3]

(iii) File C organisation ....

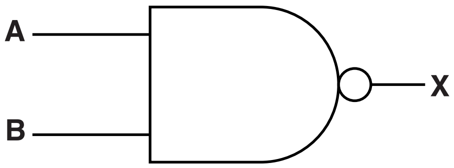
Justification ....

....

....

....[3]

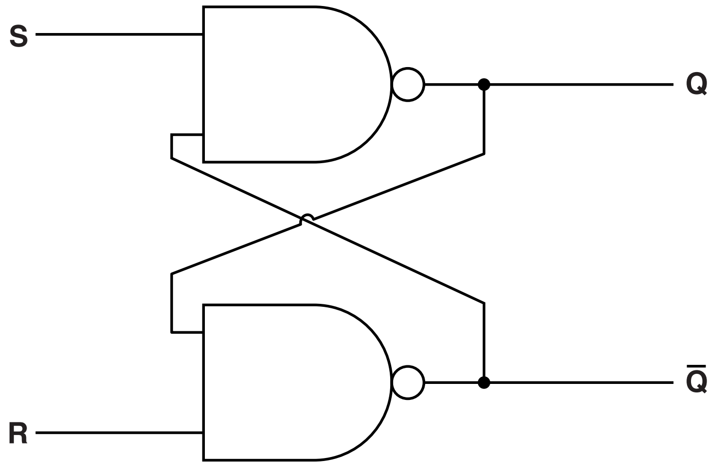
5 **(a)** Complete the truth table for this NAND gate:



A	B	X
0	0	
0	1	
1	0	
1	1	

[1]

A SR flip-flop is constructed using two NAND gates.



(b) **(i)** Complete the truth table for the SR flip-flop.

	S	R	Q	$\bar{Q}$
Initially	1	0	0	1
R changed to 1	1	1		
S changed to 0	0	1		
S changed to 1	1	1		
S and R changed to 0	0	0		

[4]

(ii) One of the combinations in the truth table should not be allowed to occur.

State the values of S and R that should not be allowed. Justify your choice.

S = ... R = ...

....

....

....

....[3]

Another type of flip-flop is the JK flip-flop.

(c) **(i)** Give one extra input present in the JK flip-flop.

....

....[1]

(ii) Give **one** advantage of the JK flip-flop.

....

....[1]

(d) Describe the role of flip-flops in a computer.

....

....

....

....[2]

6 An intruder detection system for a large house has four sensors. An 8-bit memory location stores the output from each sensor in its own bit position.

The bit value for each sensor shows:

- 1 – the sensor has been triggered
- 0 – the sensor has not been triggered

The bit positions are used as follows:

Not used				Sensor 4	Sensor 3	Sensor 2	Sensor 1

The output from the intruder detection system is a loud alarm.

(a) **(i)** State the name of the type of system to which intruder detection systems belong.

....[1]

(ii) Justify your answer to **part (i)**.

....

....[1]

(b) Name **two** sensors that could be used in this intruder detection system. Give a reason for your choice.

Sensor 1 ....

Reason ....

....

Sensor 2 ....

Reason ....

....[4]

The intruder system is set up so that the alarm will only sound if two or more sensors have been triggered.
An assembly language program has been written to process the contents of the memory location.

The table shows part of the instruction set for the processor used.

Instruction		Explanation
Op code	Operand	
LDD	<address>	Direct addressing. Load the contents of the given address to ACC
STO	<address>	Store the contents of ACC at the given address
INC	<register>	Add 1 to the contents of the register (ACC or IX)
ADD	<address>	Add the contents of the given address to the contents of ACC
AND	<address>	Bitwise AND operation of the contents of ACC with the contents of <address>
CMP	#n	Compare the contents of ACC with the number n
JMP	<address>	Jump to the given address
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True
JGT	<address>	Following a compare instruction, jump to <address> if the content of ACC is greater than the number used in the compare instruction
END		End the program and return to the operating system

(c) Part of the assembly code is:

	Op code	Operand
SENSORS:		B00001010
COUNT:		0
VALUE:		1
LOOP:	LDD	SENSORS
	AND	VALUE
	CMP	#0
	JPE	ZERO
	LDD	COUNT
	INC	ACC
	STO	COUNT
ZERO:	LDD	VALUE
	CMP	#8
	JPE	EXIT
	ADD	VALUE
	STO	VALUE
	JMP	LOOP
EXIT:	LDD	COUNT
TEST:	CMP	…
	JGT	ALARM

(i) Dry run the assembly language code. Start at `LOOP` and finish when `EXIT` is reached.

BITREG	COUNT	VALUE	ACC
B00001010	0	1	

[4]

(ii) The operand for the instruction labelled `TEST` is missing.

State the missing operand.

....[1]

(iii) The intruder detection system is improved and now has eight sensors.

One instruction in the assembly language code will need to be amended.

Identify this instruction ....

Write the amended instruction ....[2]

BLANK PAGE

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

8889038264

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2020**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Blank pages are indicated.

DC (JP) 199349

1 Carlos is writing exception handling code for his program.

(a) State what is meant by an **exception**.

....

.... [1]

(b) Give **three** situations where an exception handling routine would be required.

1 ....

....

2 ....

....

3 ....

....
[3]

(c) Describe the benefits of using exception handling in a program.

....

....

....

.... [2]

2 **(a)** Programs can be written using recursion.

Tick (✓) one or more boxes to show the features that **must** be included in a valid recursive algorithm.

Feature	**Must be included**
Incrementation	
General case	
Base case	
Selection case	
It calls itself	

[2]

(b) The following recursive procedure outputs every even number from the positive parameter value down to and including 2.

The procedure checks if the integer parameter is an even or an odd number. If the number is odd, the procedure converts it to an even number by subtracting 1 from it.

The function `MOD(ThisNum : INTEGER, ThisDiv : INTEGER)` returns the remainder value when `ThisNum` is divided by `ThisDiv`.

Complete the **pseudocode** for the recursive procedure.

```
PROCEDURE Count(BYVALUE .................................. : INTEGER)

    IF ..................................(Number, 2) <> 0

        THEN

            Number ← Number - 1

    ENDIF

    OUTPUT ............................................................

    IF Number > 0

        THEN

            ..................................(.................................. - 1)

    ENDIF

ENDPROCEDURE
```

[5]

(c) A program allows guests to input a meal option at a wedding.

Guests can choose meal option 1 or meal option 2.

The program will keep count of the numbers of each meal option chosen.

The program ends when a value other than 1 or 2 is entered. It then outputs the count of each meal option.

```
PROCEDURE MealsCount(BYREF MealOption1 : INTEGER, MealOption2 : INTEGER)

    DECLARE MealOption : INTEGER

    DECLARE MoreMeals : BOOLEAN

    MoreMeals ← True

    WHILE MoreMeals = True

        INPUT MealOption

        IF MealOption = 1

            THEN

                MealOption1 ← MealOption1 + 1

            ELSE

                IF MealOption = 2

                    THEN

                        MealOption2 ← MealOption2 + 1

                    ELSE

                        OUTPUT MealOption1, " ", MealOption2

                        MoreMeals ← False

                ENDIF

        ENDIF

    ENDWHILE

ENDPROCEDURE
```

The program contains a conditional loop.

Use **pseudocode** to rewrite the conditional loop as a recursive algorithm.

```
PROCEDURE MealsCount(BYREF MealOption1 : INTEGER, MealOption2 : INTEGER)

    DECLARE MealOption : INTEGER

..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................
..........................................................................................

ENDPROCEDURE
```

[5]

3 A declarative programming language is used to represent the following knowledge base.

```
01 person(jessica).

02 person(pradeep).

03 person(steffi).

04 person(johann).

05 sport(football).

06 sport(hockey).

07 sport(cricket).

08 sport(volleyball).

09 plays(johann, football).

10 plays(steffi, cricket).

11 plays(jessica, football).

12 will_not_play(pradeep, cricket).
```

These clauses have the following meanings:

Clause	Meaning
01	Jessica is a person
05	Football is a sport
09	Johann plays football
12	Pradeep refuses to play cricket

(a) Elle is a person who plays rugby but refuses to play hockey.

Write additional clauses to represent this information.

13 ....

14 ....

15 ....

16 ....

[4]

(b) Write the result returned by the goal:

```
plays(X, football).
```

`X =` .... [1]

(c) Y might play X, if Y is a person, X is a sport and Y does not refuse to play X.

Write this as a rule.

```
mightplay(Y , X)
```

`IF` ....

....

.... [5]

4 Object-oriented programming has several features. These include inheritance, classes, methods and properties.

(a) Describe what is meant by **inheritance**.

....

....

....

.... [2]

(b) Identify **two other** features of object-oriented programming.

1 ....

2 ....

[2]

5 A tennis club is developing a program to store details of the lessons it offers. The programmer has designed the class `Lesson` for the details of the lessons.

The following class diagram shows the design for the `Lesson` class.

Lesson
`LessonType : STRING // initialised in constructor to the parameter` `                     // value passed to the constructor` `Instructor : STRING // initialised in constructor to the parameter` `                     // value passed to the constructor`
`Constructor()       // method used to create and initialise an` `                     // object` `GetLessonType()     // returns LessonType value` `GetInstructor()     // returns Instructor value` `GetFee()            // returns the cost of a lesson` `SetLessonType()     // sets the LessonType to the parameter value` `SetInstructor()     // sets the Instructor to the parameter value`

(a) Write **program code** for the `Constructor()` method.

Use the appropriate constructor method for your chosen programming language.

Programming language ....

Program code

....

....

....

....

....

....

....

.... [3]

(b) Write **program code** for the `GetLessonType()` method.

Programming language ....

Program code

....

....

....

....

....

.... [2]

(c) `Fee` is the cost that a customer will pay for a lesson.

The method `GetFee()` validates the parameter value. The method is sent a parameter value that represents the skill level of the customer: beginner, intermediate or advanced.

The parameter will be a character:

- `'B'` for beginner
- `'I'` for intermediate
- `'A'` for advanced.

The method must check the parameter value is a valid character (`'B'`, `'I'` or `'A'`) and return the correct fee. It must return `-1` if it is not a valid character.

The fees are:

- $45 for a beginner
- $50 for an intermediate
- $55 for an advanced.

Write **program code** for the `GetFee()` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(d) The tennis club only offers nine different types of lesson. The lesson objects are stored in a 1D array.

Write **pseudocode** to declare an array `LessonArray` to store the nine lesson objects.

....

.... [2]

(e) The tennis club has the lesson 'Improve Your Serve' that has David as the instructor.

Write **program code** to create the lesson 'Improve Your Serve' as an instance of the class `Lesson`. The object needs to be stored in the third element of the array `LessonArray`.

Programming language ....

Program code

....

....

.... [3]

6 A theatre company stores customer login details to allow customers to book tickets online.

A hash table stores login details for 2000 customers.

Each customer's details are stored in a record.

The declaration for `CustomerRecord` is:

```
TYPE CustomerRecord

      DECLARE UserID : STRING

      DECLARE PINNumber : INTEGER

ENDTYPE
```

A 1D array, `CustomerDetails`, is used to implement the hash table. `CustomerDetails` is a global array. The 1D array has 6000 elements.

(a) The procedure `InitialiseHashTable()` initialises the hash table. `UserID` is initialised as an empty string, and `PINNumber` initialised to `0` for all of the records.

Write **pseudocode** for the procedure `InitialiseHashTable()`.

....

....

....

....

....

....

....

.... [4]

(b) The function `InsertRecord()` is used to insert a new record into the hash table.

The function, `Hash()`:

- takes a `UserID` as a parameter
- performs the hashing algorithm
- returns the calculated index of the user ID within the hash table.

If the hash table is full, the function `InsertRecord()` returns `-1`. If there is space available in the hash table, the record is inserted, and it returns the position of this record in the array.

Complete the **pseudocode** for the function.

```
FUNCTION InsertRecord(NewRecord : CustomerRecord) RETURNS INTEGER

    DECLARE Count : INTEGER

    DECLARE Index : INTEGER

    Count ← 0

    Index ← Hash(.....................................................................................)

    WHILE (CustomerDetails[Index].UserID <> "") ........................ (Count <= 5999)

        Index ← Index + 1

        Count ← Count + 1

        IF Index > 5999

            THEN

                ....................................................................

        ENDIF

    ENDWHILE

    IF Count > 5999

        THEN

            ............................................................

        ELSE

            CustomerDetails[..........................................] ← .......................................

            ............................................................

    ENDIF

ENDFUNCTION
```

[7]

7 **(a)** A shirt design company has an order form to order shirts. Customers can order multiple shirts using the same form.

The customer details section has the data:

- name
- address
- telephone number.

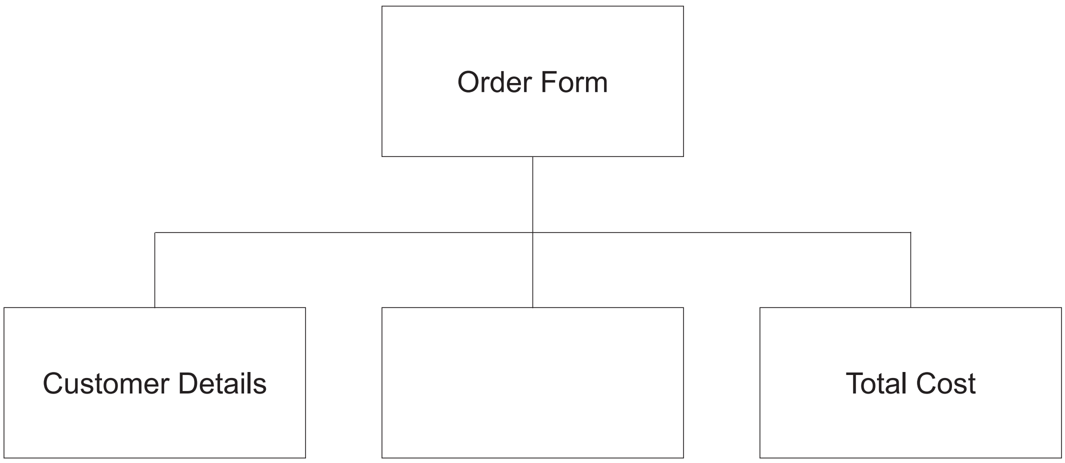
The order details section has the data:

- shirt ID
- colour
- cost.

A total cost for the order is also calculated.

The cost of each shirt is dependent on the size ordered. The sizes customers can order are small, medium and large.

Complete the following JSP data structure diagram for the order form.



[7]

(b) Each customer's order is stored as a record in a file. The customers' orders are stored in the order in which they arrive in the file and no key field is used.

(i) Identify this type of file structure.

.... [1]

(ii) Identify **two other** types of file structure.

1 ....

2 ....
[2]

(c) The procedure `UpdateTelephone()` allows the shirt company to update the record for a customer's details. The procedure will update the telephone number.

The program stores customer details as a custom data type, `Customer`.

The definition for this data type is:

```
TYPE Customer

    Name : STRING

    Address : STRING

    TelephoneNumber : STRING

ENDTYPE
```

The procedure `UpdateTelephone()` takes the customer record to be updated and the new telephone number as parameters. It then updates the telephone number in the record.

Complete the **pseudocode** for the procedure `UpdateTelephone()`.

```
PROCEDURE UpdateTelephone(.................... ThisCustomer : Customer,

          .................... NewTelephoneNumber : STRING)

    ..............................................................................

ENDPROCEDURE
```

[3]

(d) The shirt company is looking to implement a system to reward customers. The system includes:

- 10% discount on orders over $50
- free gift if order over $50 and if the order is placed on a Monday
- additional 5% discount if a customer has a loyalty card
- free delivery for a customer with a loyalty card and spends over $50.

Complete the following decision table for this system.

Conditions	**Order over $50**	**Y**	**Y**	**Y**	**Y**	**N**	**N**	**N**	**N**
	Monday	**Y**	**Y**	**N**	**N**	**Y**	**Y**	**N**	**N**
	Loyalty card	**Y**	**N**	**Y**	**N**	**Y**	**N**	**Y**	**N**
Actions	**Additional 5% discount**								
	10% discount								
	Free gift								
	Free delivery								

[4]

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

8118378603

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 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 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.
DO **NOT** WRITE IN ANY BARCODES.

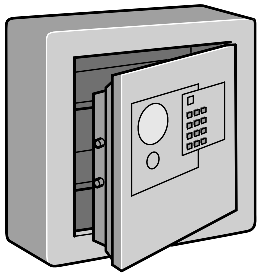
Answer **all** questions.
No marks will be awarded for using brand names of software packages or hardware.

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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

1 A user can lock a safety deposit box by inputting a 4-digit code. The user can unlock the box with the same 4-digit code.



There is a keypad on the door of the safety deposit box. The following diagram shows the keys on the keypad.

1	2	3
4	5	6
7	8	9
R	0	Enter

Initially, the safety deposit box door is open and the user has not set a code.

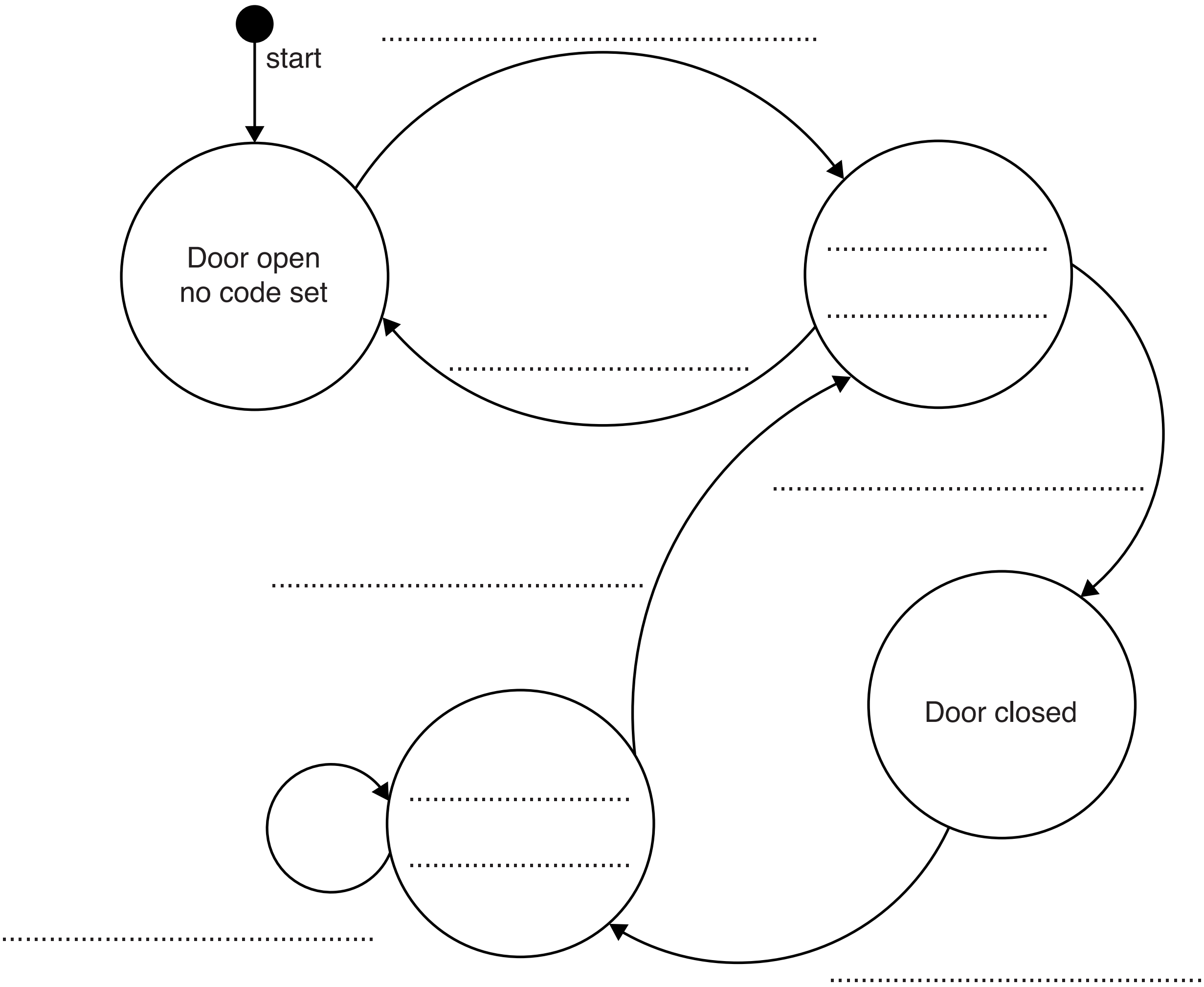
The operation of the safety deposit box is as follows:

A) To set a new code the door must be open. The user chooses a 4-digit code and sets it by pressing the numerical keys on the keypad, followed by the Enter key. Until the user clears this code, it remains the same. (See point E below)
B) The user can only close the door if the user has set a code.
C) To lock the door, the user closes the door, enters the set code and presses the Enter key.
D) To unlock the door, the user enters the set code. The door then opens automatically.
E) The user clears the code by opening the door and pressing the R key, followed by the Enter key. The user can then set a new code. (See point A above)

The following state transition table shows the transition from one state to another of the safety deposit box:

Current state	Event	Next state
Door open, no code set	4-digit code entered	Door open, code set
Door open, code set	R entered	Door open, no code set
Door open, code set	Close door	Door closed
Door closed	Set code entered	Door locked
Door locked	Set code entered	Door open, code set
Door locked	R entered	Door locked

(a) Complete the state-transition diagram.



[7]

(b) A company wants to simulate the use of a safety deposit box. It will do this with object-oriented programming (OOP).

The following diagram shows the design for the class `SafetyDepositBox`. This includes the properties and methods.

```
SafetyDepositBox
--------------------------------------------------------------------
Code     : STRING // 4 digits
State    : STRING // "Open-NoCode", "Open-CodeSet", "Closed"
                  // or "Locked"
--------------------------------------------------------------------
Create()          // method to create and initialise an object
                  // if using Python use __init__
Reset()           // clears Code
SetState()        // set state to parameter value
                  // and output new state
SetNewCode()      // sets Code to parameter value
                  // output message and new code
StateChange()     // reads keypad and takes appropriate action
```

Write **program code** for the following methods.

Programming language ....

(i) `Create()`

....

....

....

....

.... [3]

(ii) `Reset()`

....

....

.... [2]

(iii) `SetState()`

....

....

....

.... [2]

(iv) `SetNewCode()`

....

....

....

.... [2]

(v) The user must enter a 4-digit code.

Write **program code** for a function `Valid(s : STRING)` that returns:

- `TRUE` if the input string `s` consists of exactly 4 digits
- `FALSE` otherwise

Programming language ....

....

....

....

....

....

....

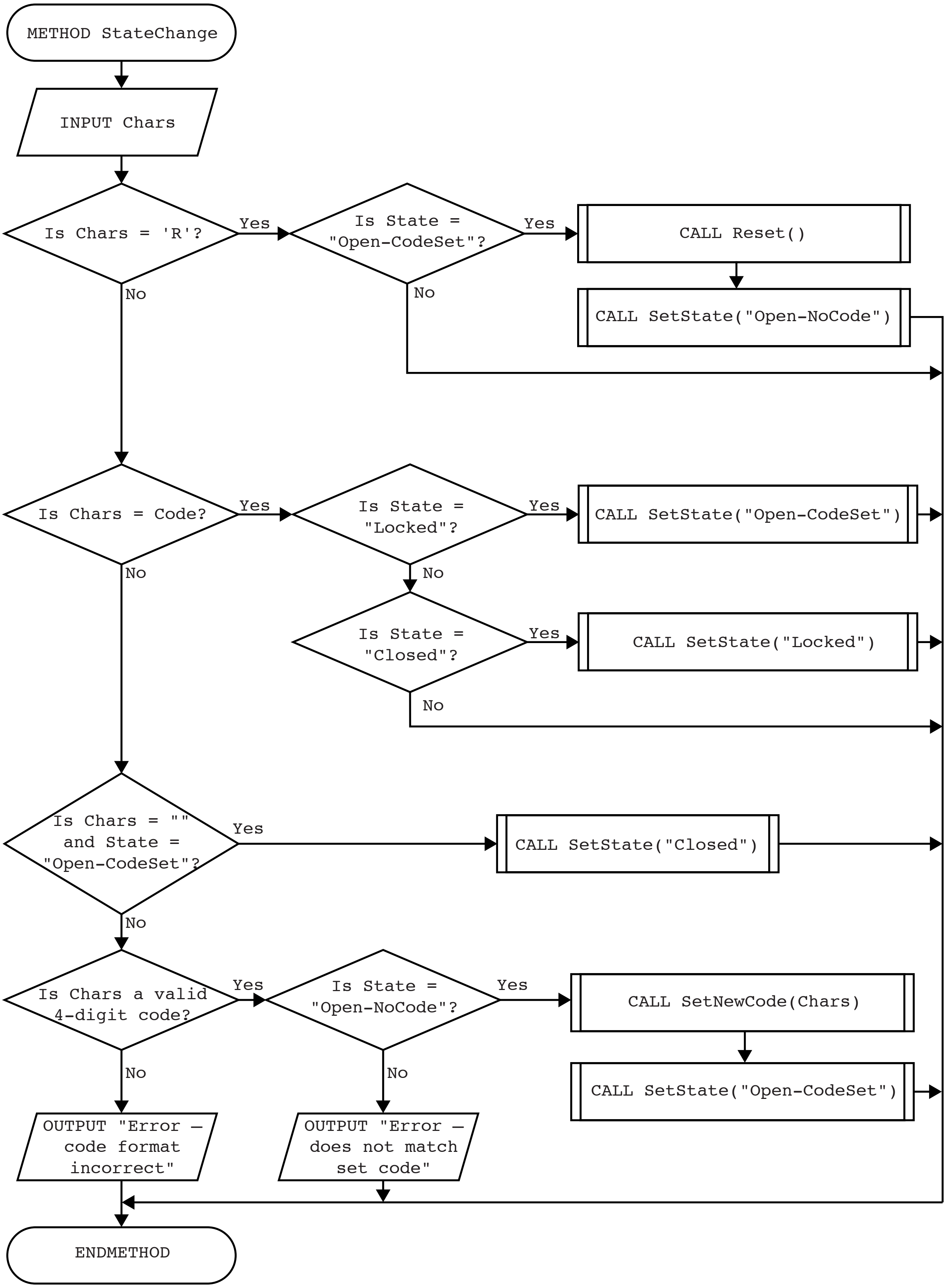
....

....

....

.... [4]

(vi) Convert the flowchart to **program code** for the method `StateChange()`. Use the properties and methods in the original class definition and the `Valid()` function from **part (v)**.



Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[12]

(vii) The company needs to write a program to simulate a safety deposit box. The program will create an object with identifier `ThisSafe`, which is an instance of the class `SafetyDepositBox`.

The main program design is:

```
instantiate ThisSafe (create and initialise ThisSafe)
loop forever (continually use ThisSafe)
   call StateChange() method
end loop
```

Write **program code** for the main program.

Programming language ....

....

....

....

....

....

....

....

....

....

....

.... [4]

(c) It is possible to declare properties and methods as either public or private.

The programmer has modified the class design for `SafetyDepositBox` as follows:

```
SafetyDepositBox
-----------------------
PRIVATE
   Code    : STRING
   State   : STRING
-----------------------
PUBLIC
   Create()
   StateChange()
PRIVATE
   Reset()
   SetState()
   SetNewCode()
```

(i) Describe the effects of declaring the `SafetyDepositBox` properties as private.

....

....

....

.... [2]

(ii) Describe the effects of declaring two methods of the class as public and the other three as private.

....

....

....

.... [2]

2 Circle the programming language that you have studied:

Visual Basic (console mode) Python Pascal Delphi (console mode)

(a) **(i)** Name the programming environment you have used when typing in program code.

....

....

List **three** features of the editor that helped you to write program code.

1 ....

....

2 ....

....

3 ....

.... [3]

(ii) Explain when and how your programming environment reports a syntax error.

When ....

....

....

How ....

....

.... [2]

Question 2 continues on page 12.

(iii) The table shows a module definition for `BinarySearch` in three programming languages.

Study **one** of the examples. Indicate your choice by circling A, B or C:

A **B** **C**

A) Python

```
def BinarySearch(List, Low, High, SearchItem):
   Index = -1
   while (Index == -1) AND (Low <= High):
      Middle = (High + Low) // 2
      if List[Middle] == SearchItem:
         Index = Middle
      elif List[Middle] < SearchItem:
         Low = Middle + 1
      else:
         High = Middle - 1
   return(Middle)
```

B) Pascal/Delphi

```
FUNCTION BinarySearch(VAR List : ARRAY OF INTEGER; Low, High,
                                    SearchItem : INTEGER) : INTEGER;
VAR Index, Middle : INTEGER;
BEGIN
   Index := -1;
   WHILE (Index = -1) & (Low <= High) DO
      BEGIN
         Middle := (High + Low) DIV 2;
         IF List[Middle] = SearchItem
            THEN Index := Middle
            ELSE IF List[Middle] < SearchItem
                    THEN Low := Middle + 1
                    ELSE High := Middle - 1;
      END;
   Result := Middle;
END;
```

C) Visual Basic

```
Function BinarySearch(ByRef List() As Integer, ByVal Low As Integer,
        ByVal High As Integer, ByVal SearchItem As Integer) As Integer
   Dim Index, Middle As Integer
   Index = -1
   Do While (Index = -1) & (Low <= High)
      Middle = (High + Low) \ 2
      If List(Middle) = SearchItem Then
         Index = Middle
      ElseIf List(Middle) < SearchItem Then
         Low = Middle + 1
      Else
         High = Middle - 1
      End If
   Loop
   BinarySearch = Middle
End Function
```


The programming environment reported a syntax error in the `BinarySearch` code.

State the line number: ....

Write the correct code for this line.

....[2]

(b) **(i)** State whether programs written in your programming language are compiled or interpreted.

....

.... [1]

(ii) A programmer corrects the syntax error and tests the function. It does not perform as expected when the search item is not in the list.

State the type of error: ....

Write down the line number where the error occurs.

....

Write the correct code for this line.

....[2]

(iii) State the programming environment you have used when debugging program code.

....

....

Name **two** debugging features and describe how they are used.

1 ....

....

....

....

2 ....

....

....

.... [4]

3 The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC), and an index register (IX).

Instruction		Explanation
Op code	**Operand**	
LDM	#n	Immediate addressing. Load the number n to ACC.
LDD	<address>	Direct addressing. Load the contents of the given address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
LDR	#n	Immediate addressing. Load the number n into IX.
STO	<address>	Store the contents of ACC at the given address.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
CMP	<address>	Compare the contents of ACC with the contents of <address>.
CMP	#n	Compare the contents of ACC with number n.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
OUT		Output to the screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

A programmer is writing a program that outputs a string, first in its original order and then in reverse order.

The program will use locations starting at address `NAME` to store the characters in the string. The location with address `MAX` stores the number of characters that make up the string.

The programmer has started to write the program in the table opposite. The Comment column contains descriptions for the missing program instructions.

Complete the program using op codes from the given instruction set.

Label	Op code	Operand	Comment
START:			// initialise index register to zero
			// initialise COUNT to zero
LOOP1:			// load character from indexed address NAME
			// output character to screen
			// increment index register
			// increment COUNT starts here
			// is COUNT = MAX ?
			// if FALSE, jump to LOOP1
REVERSE:			// decrement index register
			// set ACC to zero
			// store in COUNT
LOOP2:			// load character from indexed address NAME
			// output character to screen
			// decrement index register
			// increment COUNT starts here
			// is COUNT = MAX ?
			// if FALSE, jump to LOOP2
			// end of program
COUNT:			
MAX:	4		
NAME:	B01000110		// ASCII code in binary for 'F'
	B01010010		// ASCII code in binary for 'R'
	B01000101		// ASCII code in binary for 'E'
	B01000100		// ASCII code in binary for 'D'

[15]

4 Commercial software usually undergoes acceptance testing and integration testing.

Distinguish between the two types of testing by stating:

- who does the testing
- when the testing occurs
- the specific purpose of each type of testing

(i) Acceptance testing

Who ....

....

When ....

....

Purpose ....

....[3]

(ii) Integration testing

Who ....

....

When ....

....

Purpose ....

....[3]

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

8788782541

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2016**

1 hour 30 minutes

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 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.
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.

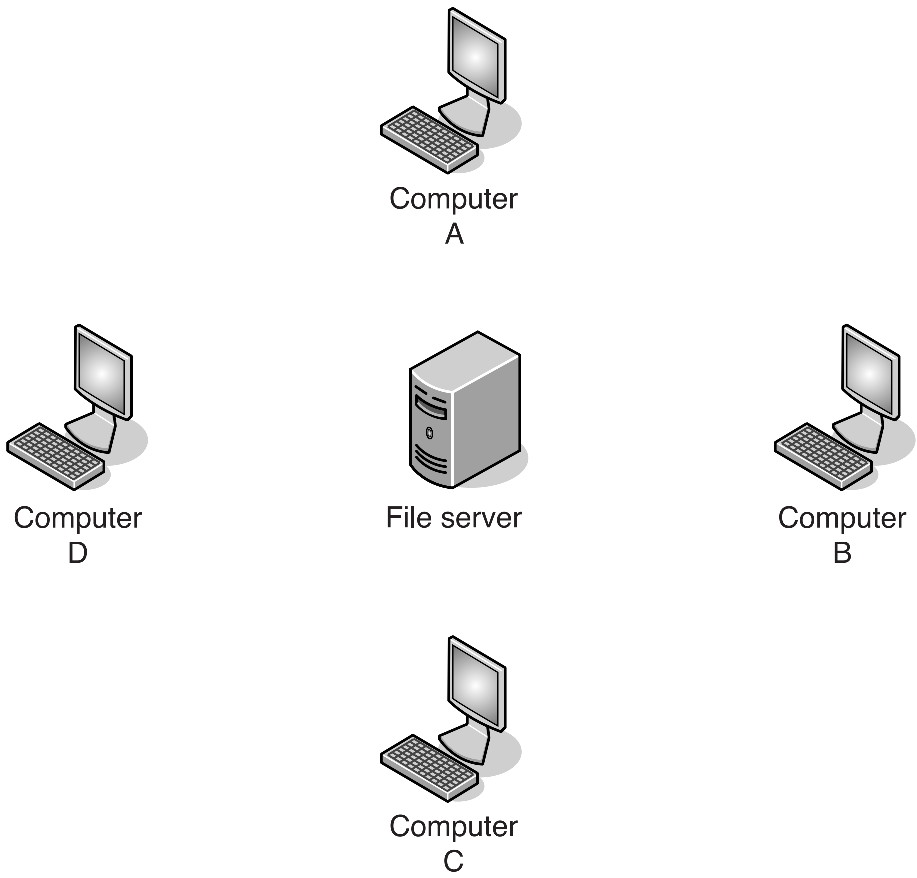
The maximum number of marks is 75.

This document consists of **15** printed pages and **1** blank page.

DC (NH/AR) 115894/3

1 A Local Area Network (LAN) consists of four computers and one server. The LAN uses a bus topology.

(a) Complete the diagram below to show how the computers and the File server could be connected.



[2]

(b) Computer C sends a data packet to Computer A.

Three statements are given below.

Tick (✓) to show whether each statement is true or false.

Statement	True	False
Computer C uses the IP address of Computer A to indicate that the packet is for Computer A.		
Computer B can read the packet sent from Computer C to Computer A.		
The File server routes the packet to Computer A.		

[3]

(c) Computer A starts transmitting a packet to Computer C. At exactly the same time, the File server starts transmitting a packet to Computer D. This causes a problem.

(i) State the name given to this problem.

....

....[1]

(ii) Give **three** steps taken by both Computer A and the File server to allow them to transmit their packets successfully.

Step 1 ....

....

Step 2 ....

....

Step 3 ....

....[3]

(d) Adding a switch to the LAN changes its topology. Explain how the use of a switch removes the problem identified in **part (c)(i)**.

....

....

....

....

....

....

....

....[4]

2 Digital certificates are used in Internet communications. A Certificate Authority (CA) is responsible for issuing digital certificates.

(a) Name **three** data items present in a digital certificate.

1 ....

2 ....

3 ....[3]

(b) The method of issuing a digital certificate is as follows:

1 A user starts an application for a digital certificate using their computer. On this computer a key pair is generated. This key pair consists of a public key and an associated private key.

2 The user submits the application to the CA. The generated .... **(i)** .... key and other application data are sent. The key and data are encrypted using the CA's .... **(ii)** .... key.

3 The CA creates a digital document containing all necessary data items and signs it using the CA's .... **(iii)** .... key.

4 The CA sends the digital certificate to the individual.

In the above method there are three missing words. Each missing word is either 'public' or 'private'.

State the correct word. Justify your choice.

(i) ....

Justification ....

....[2]

(ii) ....

Justification ....

....[2]

(iii) ....

Justification ....

....[2]

(c) Alexa sends an email to Beena.

Alexa's email program:

- produces a message digest (hash)
- uses Alexa's private key to encrypt the message digest
- adds the encrypted message digest to the plain text of her message
- encrypts the whole message with Beena's public key
- sends the encrypted message with a copy of Alexa's digital certificate

Beena's email program decrypts the encrypted message using her private key.

(i) State the name given to the encrypted message digest.

....[1]

(ii) Explain how Beena can be sure that she has received a message that is authentic (not corrupted or tampered with) and that it came from Alexa.

....

....

....

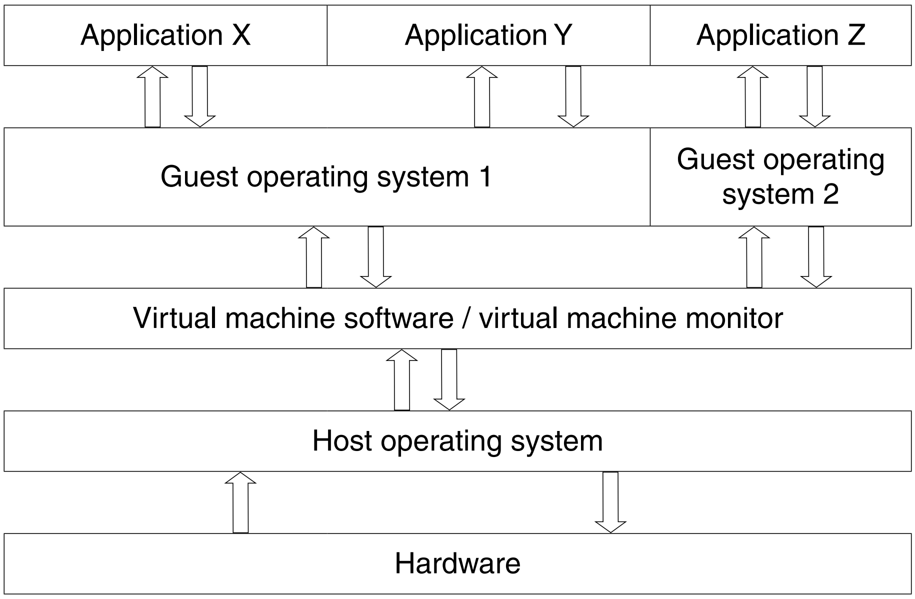
....[2]

(iii) Name **two** uses where encrypted message digests are advisable.

1 ....

2 ....[2]

3 **(a)** The following diagram shows how applications X, Y and Z can run on a virtual machine system.



(i) The virtual machine software undertakes many tasks.

Describe **two** of these tasks.

Task 1 ....

....

Task 2 ....

....[2]

(ii) Explain the difference between a **guest operating system** and a **host operating system**.

....

....

....

....[2]

(b) A company uses a computer as a web server. The manufacturer will no longer support the computer's operating system (OS) in six months' time. The company will then need to decide on a replacement OS.

The company is also considering changing the web server software when the OS is changed.

Whenever any changes are made, it is important that the web server service is not disrupted.

In developing these changes, the company could use virtual machines.

(i) Describe **two** possible uses of virtual machines by the company.

Use 1 ....

....

....

....

Use 2 ....

....

....

....[4]

The web server often has to handle many simultaneous requests.

(ii) The company uses a virtual machine to test possible solutions to the changes that they will need to make.

Explain **one** limitation of this approach.

....

....

....

....[2]

4 **(a)** Three file organisation methods and two file access methods are shown below.

Draw lines to link each file organisation method to its appropriate file access method or methods.

File organisation method	**File access method**
serial	direct
sequential	sequential
random	

[4]

(b) A bank has a very large number of customers. The bank stores data for each customer. This includes:

- unique customer number
- personal data (name, address, telephone number)
- transactions

The bank computer system makes use of three files:

- A – a file that stores customer personal data. This file is used at the end of each month for the production of the monthly statement.
- B – a file that stores encrypted personal identification numbers (PINs) for customer bank cards. This file is accessed when the customer attempts to withdraw cash at a cash machine (ATM).
- C – a file that stores all customer transaction records for the current month. Every time the customer makes a transaction, a new record is created.

For each of the files A, B and C, state an appropriate method of organisation. Justify your choice.

(i) File A organisation ....

Justification ....

....

....

....[3]

(ii) File B organisation ....

Justification ....

....

....

....[3]

(iii) File C organisation ....

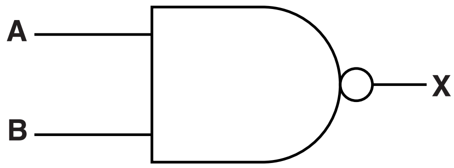
Justification ....

....

....

....[3]

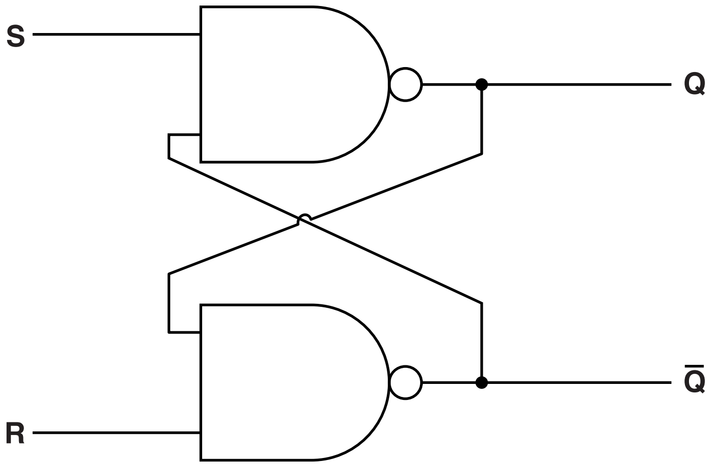
5 **(a)** Complete the truth table for this NAND gate:



A	B	X
0	0	
0	1	
1	0	
1	1	

[1]

A SR flip-flop is constructed using two NAND gates.



(b) **(i)** Complete the truth table for the SR flip-flop.

	S	R	Q	$\bar{Q}$
Initially	1	0	0	1
R changed to 1	1	1		
S changed to 0	0	1		
S changed to 1	1	1		
S and R changed to 0	0	0		

[4]

(ii) One of the combinations in the truth table should not be allowed to occur.

State the values of S and R that should not be allowed. Justify your choice.

S = ... R = ...

....

....

....

....[3]

Another type of flip-flop is the JK flip-flop.

(c) **(i)** Give one extra input present in the JK flip-flop.

....

....[1]

(ii) Give **one** advantage of the JK flip-flop.

....

....[1]

(d) Describe the role of flip-flops in a computer.

....

....

....

....[2]

6 An intruder detection system for a large house has four sensors. An 8-bit memory location stores the output from each sensor in its own bit position.

The bit value for each sensor shows:

- 1 – the sensor has been triggered
- 0 – the sensor has not been triggered

The bit positions are used as follows:

Not used				Sensor 4	Sensor 3	Sensor 2	Sensor 1

The output from the intruder detection system is a loud alarm.

(a) **(i)** State the name of the type of system to which intruder detection systems belong.

.... [1]

(ii) Justify your answer to **part (i)**.

....

.... [1]

(b) Name **two** sensors that could be used in this intruder detection system. Give a reason for your choice.

Sensor 1 ....

Reason ....

....

Sensor 2 ....

Reason ....

.... [4]

The intruder system is set up so that the alarm will only sound if two or more sensors have been triggered.
An assembly language program has been written to process the contents of the memory location.

The table shows part of the instruction set for the processor used.

Instruction		Explanation
Op code	Operand	
LDD	<address>	Direct addressing. Load the contents of the given address to ACC
STO	<address>	Store the contents of ACC at the given address
INC	<register>	Add 1 to the contents of the register (ACC or IX)
ADD	<address>	Add the contents of the given address to the contents of ACC
AND	<address>	Bitwise AND operation of the contents of ACC with the contents of <address>
CMP	#n	Compare the contents of ACC with the number n
JMP	<address>	Jump to the given address
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True
JGT	<address>	Following a compare instruction, jump to <address> if the content of ACC is greater than the number used in the compare instruction
END		End the program and return to the operating system

(c) Part of the assembly code is:

	Op code	Operand
SENSORS:		B00001010
COUNT:		0
VALUE:		1
LOOP:	LDD	SENSORS
	AND	VALUE
	CMP	#0
	JPE	ZERO
	LDD	COUNT
	INC	ACC
	STO	COUNT
ZERO:	LDD	VALUE
	CMP	#8
	JPE	EXIT
	ADD	VALUE
	STO	VALUE
	JMP	LOOP
EXIT:	LDD	COUNT
TEST:	CMP	…
	JGT	ALARM

(i) Dry run the assembly language code. Start at `LOOP` and finish when `EXIT` is reached.

BITREG	**COUNT**	**VALUE**	**ACC**
B00001010	0	1	

[4]

(ii) The operand for the instruction labelled `TEST` is missing.

State the missing operand.

....[1]

(iii) The intruder detection system is improved and now has eight sensors.

One instruction in the assembly language code will need to be amended.

Identify this instruction ....

Write the amended instruction ....[2]

BLANK PAGE

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

7906049179

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **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 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.
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.

The maximum number of marks is 75.

This document consists of **19** printed pages and **1** blank page.

CAMBRIDGE International Examinations

1 A linked list abstract data type (ADT) is to be used to store and organise surnames.

This will be implemented with a 1D array and a start pointer. Elements of the array consist of a user-defined type. The user-defined type consists of a data value and a link pointer.

Identifier	Data type	Description
`LinkedList`	`RECORD`	User-defined type
`Surname`	`STRING`	Surname string
`Ptr`	`INTEGER`	Link pointers for the linked list

(a) **(i)** Write **pseudocode** to declare the type `LinkedList`.

....

....

....

.... [3]

(ii) The 1D array is implemented with an array `SurnameList` of type `LinkedList`.

Write the **pseudocode** declaration statement for `SurnameList`. The lower and upper bounds of the array are 1 and 5000 respectively.

.... [2]

(b) The following surnames are organised as a linked list with a start pointer `StartPtr`.

`StartPtr: 3`

	1	**2**	**3**	**4**	**5**	**6**		**5000**
Surname	Liu	Yang	Chan	Wu	Zhao	Huang	...	
Ptr	4	5	6	2	0	1	...	

State the value of the following:

(i) `SurnameList[4].Surname` .... [1]

(ii) `SurnameList[StartPtr].Ptr` .... [1]

(c) Pseudocode is to be written to search the linked list for a surname input by the user.

Identifier	Data type	Description
`ThisSurname`	`STRING`	The surname to search for
`Current`	`INTEGER`	Index to array `SurnameList`
`StartPtr`	`INTEGER`	Index to array `SurnameList`. Points to the element at the start of the linked list

(i) Study the pseudocode in **part (c)(ii)**.

Complete the table above by adding the missing identifier details. [2]

(ii) Complete the pseudocode.

```
Current ← .................................................................
IF Current = 0
   THEN
      OUTPUT ..............................................................
   ELSE
      IsFound ← ...........................................................
      INPUT ThisSurname
      REPEAT
         IF ................................................ = ThisSurname
            THEN
               IsFound ← TRUE
               OUTPUT "Surname found at position ", Current
            ELSE
               // move to the next list item
               ............................................................
         ENDIF
      UNTIL IsFound = TRUE OR .............................................
      IF IsFound = FALSE
         THEN
            OUTPUT "Not Found"
      ENDIF
ENDIF
```

[6]

2 (a) (i) State what is meant by a recursively defined procedure.

....

....[1]

(ii) Write the line number from the pseudocode shown in **part (b)** that shows the procedure `X` is recursive. .... [1]

(b) The recursive procedure `X` is defined as follows:

```
01 PROCEDURE X(Index, Item)
02    IF MyList[Index] > 0
03       THEN
04          IF MyList(Index) >= Item
05             THEN
06                MyList[Index] ← MyList[Index + 1]
07          ENDIF
08          CALL X(Index + 1, Item)
09    ENDIF
10 ENDPROCEDURE
```

An array `MyList` is used to store a sorted data set of non-zero integers. Unused cells contain zero.

	1	2	3	4	5	6	7	8	9	10
MyList	3	5	8	9	13	16	27	0	0	0

(i) Complete the trace table for the dry-run of the pseudocode for the procedure `CALL X(1, 9)`.

		MyList									
Index	**Item**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	9	3	5	8	9	13	16	27	0	0	0

[4]

(ii) State the purpose of procedure `X` when used with the array `MyList`.

....

....[1]

3 A car hire company hires cars to customers. Each time a car is hired, this is treated as a transaction.

For each transaction, the following data are stored.

For the customer:

- customer name
- ID number

For the hire:

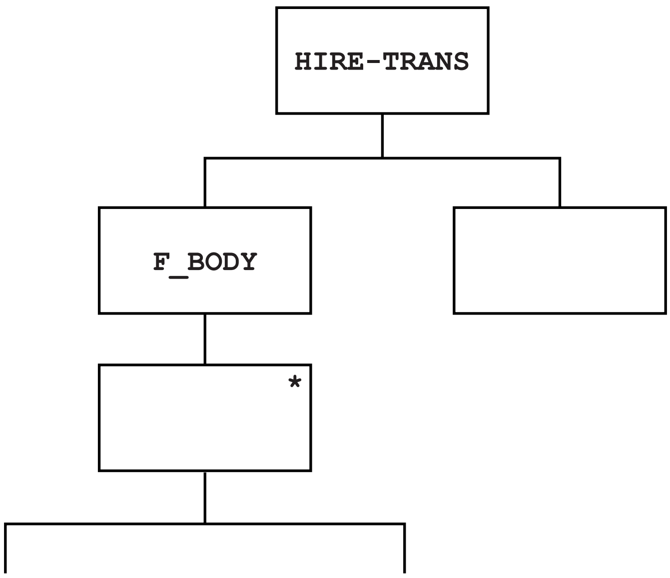
- car registration
- hire start date
- number of days hired

The transaction data are stored in a text file `HIRE-TRANS`. The file is made up of a file body, `F_BODY`, and a file trailer, `F_TRAILER`.

`F_BODY` has one transaction, `TRANS`, on each line.

(a) The first step in Jackson Structured Programming (JSP) design is to produce a JSP data structure diagram.

Complete the following JSP data structure diagram.



[7]

(b) The computer system will produce many printed reports.

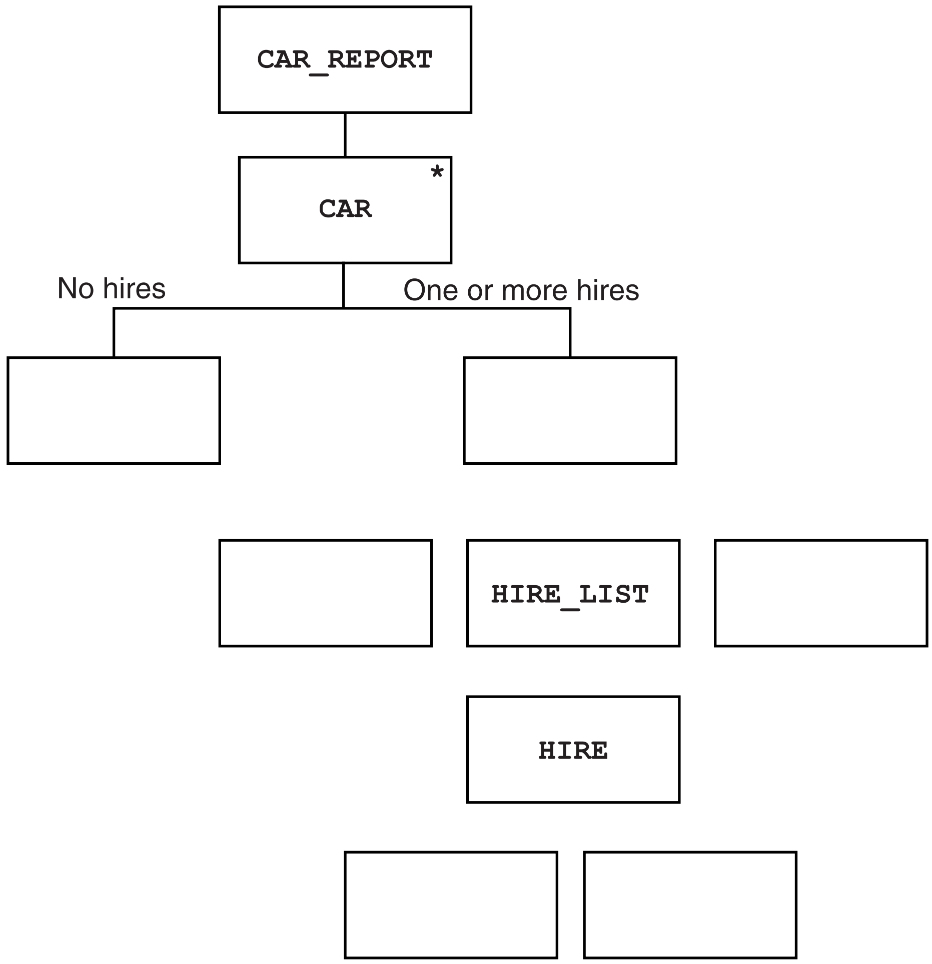
One report is `CAR_REPORT`. This displays all hire data for all cars.

For each car, the following data are displayed:

- the car data
- a list of all the hires
- the total number of hires

A car with zero hires is not included on the report.

Complete the following `CAR_REPORT` JSP data structure diagram.



[5]

4 When a car reaches a certain age, a safety assessment has to be carried out. A car's brakes and tyres must be tested. The tyre test result and the brakes test result for each car are recorded. If the car passes the assessment, a safety certificate is issued.

Cars have a unique three-character registration.

The following knowledge base is used:

```
01 car(a05).
02 car(h04).
03 car(a03).
04 car(h07).
05 car(a23).
06 car(p05).
07 car(b04).
08 carRegYear(a05, 2015).
09 carRegYear(h04, 2013).
10 carRegYear(a03, 2008).
11 carRegYear(h07, 2011).
12 carRegYear(a23, 2008).
13 carRegYear(p05, 2014).
14 carRegYear(b04, 2014).
15 testBrakes(h07, pass).
16 testTyres(h07, fail).
17 testBrakes(a03, fail).
18 testTyres(a03, fail).
19 testBrakes(a23, pass).
20 testTyres(a23, pass).
21 carAssessmentDue if carRegYear(Car, RegYear)
                                       and RegYear <= DeadlineYear.
22 issueCertificate(Car) if testTyres(Car, Result) and
                          testBrakes(Car, Result) and Result = pass.
```

(a) (i) `DeadlineYear` is assigned value `2011`.

Identify the car registrations for cars which are due to be tested.

....[1]

(ii) State how clause `22` determines whether or not a safety certificate will be issued.

....

....[1]

(b) If a car fails one of the two tests, a retest is allowed.

Write a new rule for this.

`retestAllowed(`............................`) if` ....

....

....[3]

(c) Logic programming uses a data structure called a list.

A new fact is added to the knowledge base.

`23 carList = [a03,p05,b04,h04,h07,a23].`

The following notation and operators are to be used with a list:

> `[X|Y]` denotes a list with:
>
> - `X` the first list element
> - `Y` the list consisting of the remaining list elements
>
> `[]` denotes an empty list

(i) The list `[a07,p03]` is denoted by `[A|B]`

State the value of `A` and `B`.

`A` = ...

`B` = ... [2]

(ii) The lists `[c03,d02,n05|C]` and `[c03,d02,n05,p05,m04]` are identical.

State the value of `C`.

`C` = ... [1]

(iii) The list `[a06,a02]` is denoted by `[D,E|F]`

State the value of `F`.

`F` = ... [1]

(d) The predicate `conCatCompare` is defined as a rule and returns `TRUE` or `FALSE` as follows:

`conCatCompare(X, Y, Z)` Concatenates the lists `X` and `Y` and compares the new list with list `Z`. If equal, the clause evaluates to `TRUE`, otherwise `FALSE`.

Consider the clause:

```
conCatCompare(X, Y, [a7,b6,c4])
```

If:

- the clause evaluates to `TRUE`
- and `Y` represents the list `[a7, b6, c4]`

State the value of `X`.

`X` = ...[1]

5 (a) A program calculates the exam grade awarded from a mark input by the user. The code is written as a function `CalculateGrade`.

The function:

- has a single parameter `Mark` of `INTEGER` data type
- returns the grade awarded `Grade` of `STRING` data type

The logic for calculating the grade is as follows:

Mark	Grade
Under 40	FAIL
40 and over and under 55	PASS
55 and over and under 70	MERIT
70 and over	DISTINCTION

The programmer designs the following table for test data:

Mark	Description	Expected result (Grade)
	Normal	
	Abnormal	
	Extreme/Boundary	

(i) Complete the table above. [3]

(ii) State why this table design is suitable for black box testing.

....

....[1]

(b) When designing and writing program code, explain what is meant by:

- an exception
- exception handling

....

....

....

....

....

....[3]

(c) A program is to be written to read a list of exam marks from an existing text file into a 1D array.

Each line of the file stores the mark for one student.

State **three** exceptions that a programmer should anticipate for this program.

1 ....

....

2 ....

....

3 ....

....[3]

(d) The following pseudocode is to read two numbers:

```
01 DECLARE Num1   : INTEGER
02 DECLARE Num2   : INTEGER
03 DECLARE Answer : INTEGER
04 TRY
05    OUTPUT "First number..."
06    INPUT Num1
07    OUTPUT "Second number..."
08    INPUT Num2
09    Answer ← Num1 / (Num2 – 6)
10    OUTPUT Answer
11 EXCEPT ThisException : EXCEPTION
12    OUTPUT ThisException.Message
13 FINALLY
14    // remainder of the program follows

…

29
30 ENDTRY
```

The programmer writes the corresponding program code.

A user inputs the number 53 followed by 6. The following output is produced:

```
First number...53
Second number...6
Arithmetic operation resulted in an overflow
```

(i) State the pseudocode line number which causes the exception to be raised.

.... [1]

(ii) Explain the purpose of the pseudocode on lines 11 and 12.

....

....

....

....

....

....[3]

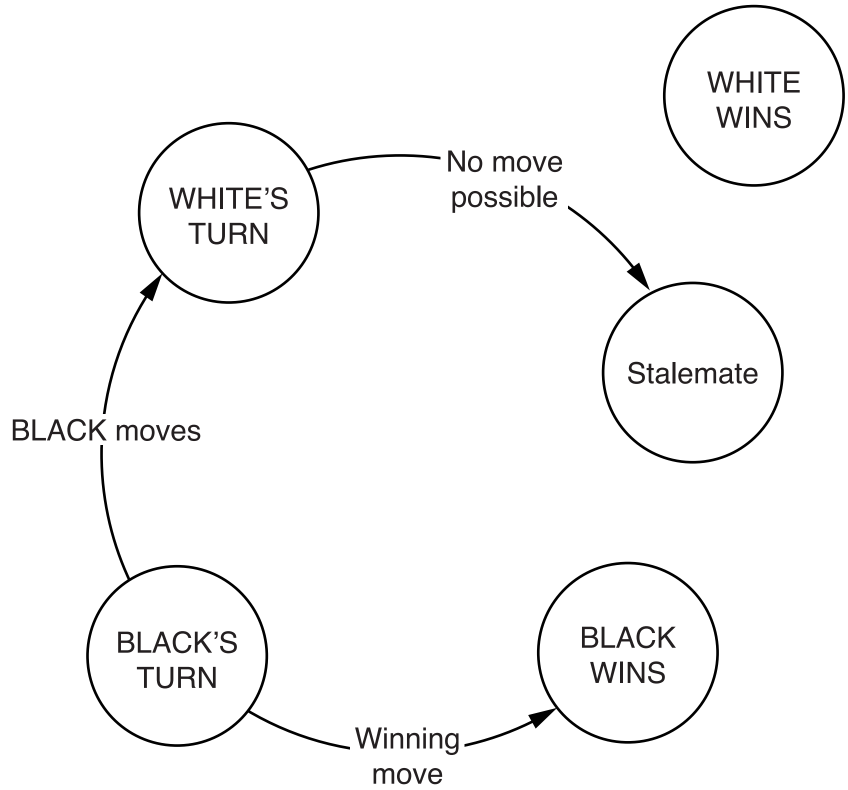
6 In a board game, one player has white pieces and the other player has black pieces. Players take alternate turns to move one of their pieces. White always makes the first move.

The game ends if:

- a player is unable to make a move when it is their turn. In this case, there is no winner. This is called 'stalemate'.
- a player wins the game as a result of their last move and is called a 'winner'.

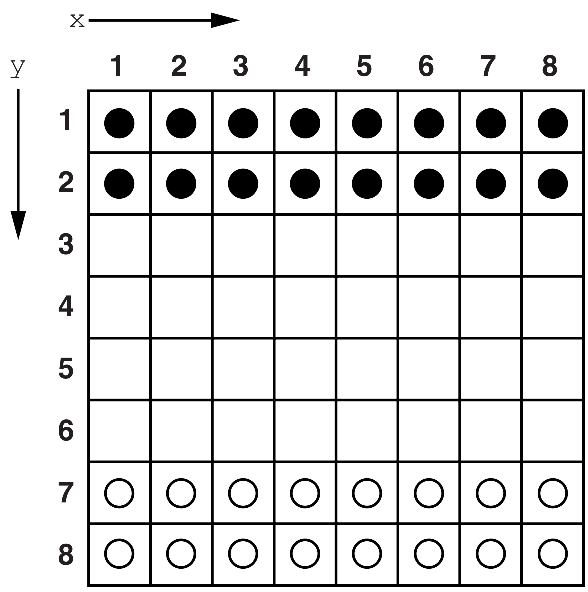
(a) A state-transition diagram is drawn to clarify how the game is played.

Complete the following state-transition diagram.



[4]

(b) The layout of the board at the start of the game is shown below:



The programmer decides to use a 2D array to represent the board. The index numbering to be used is as shown.

Each square on the board is either occupied by one piece only, or is empty.

The data stored in the array indicate whether or not that square is occupied, and if so, with a black piece or a white piece.

(i) Write **program code** to initialise the contents of the array to represent the board at the start of the game. Use characters as follows for each square:

- `'B'` represents a black piece ●
- `'W'` represents a white piece ○
- `'E'` represents an empty square

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

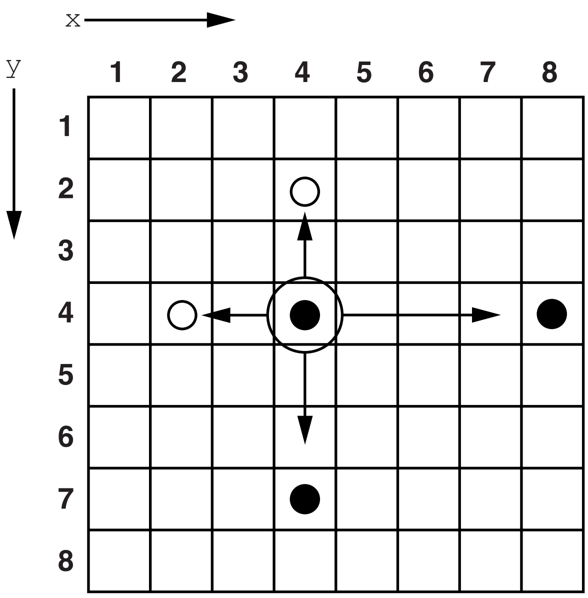
(ii) When a piece is to be moved, a procedure will calculate and output the possible destination squares for the moving piece.

A piece can move one or more squares, in the `x` or `y` direction, from its current position.

This will be a move:

- either to an empty square, with no occupied squares on the way
- or to a square containing a piece belonging to another player, with no occupied squares on the way. The other player's piece is then removed.

For example, for the circled black piece there are nine possible destination squares. Each of the two destination squares contains a white piece which would be removed.



The program requires a procedure `ValidMoves`.

It needs three parameters:

- `PieceColour` – colour of the moving piece
- `xCurrent` – current `x` position
- `yCurrent` – current `y` position

The procedure will calculate all possible destination squares **in the x direction only**.

Example output for the circled black piece is:

```
Possible moves are:
Moving LEFT
3 4
2 4 REMOVE piece
Moving RIGHT
5 4
6 4
7 4
```

Write **program code** for procedure `ValidMoves` with the following procedure header:

```
PROCEDURE ValidMoves(PieceColour : CHAR, xCurrent : INTEGER,
                                         yCurrent : INTEGER).
```

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

[5]

(c) The problem is well suited to an object-oriented design followed by object-oriented programming.

(i) Describe how classes and objects could be used in this problem.

....

....

....

....

....[2]

(ii) For a class you identified in **part(c)(i)**, state **two** properties and **two** methods.

Class ....

Properties

1 ....

2 ....

Methods

1 ....

2 ....

[2]

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

9910955723

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 2015**

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 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.
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.

The maximum number of marks is 75.

This document consists of **15** printed pages and **1** blank page.

DC (AC) 95228/3

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

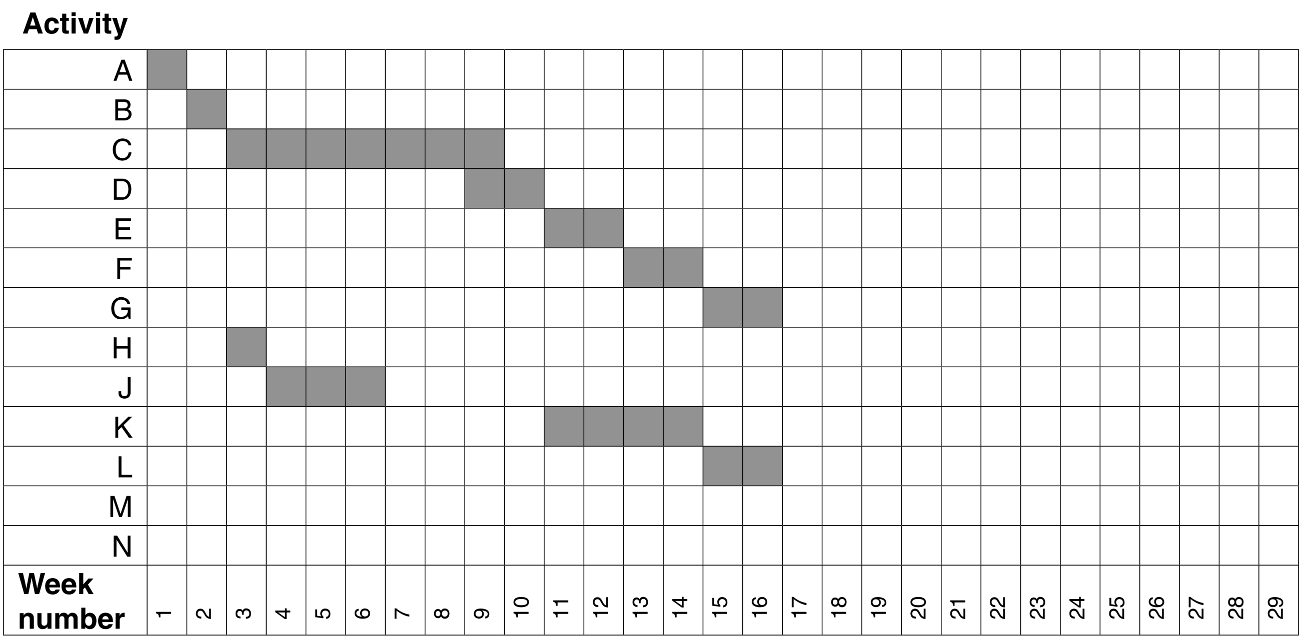
Complete the statement to indicate which high-level programming language you will use.

Programming language ………………………………………………………………………………………

1 A large software house has been asked to supply a computerised solution for a business. The project manager has drawn up a list of activities and their likely duration.

Activity	**Description**	**Weeks to complete**
A	Write requirement specification	1
B	Produce program design	1
C	Write module code	7
D	Module testing	2
E	Integration testing	2
F	Alpha testing	2
G	Install software and carry out acceptance testing	2
H	Research and order hardware	1
J	Install delivered hardware	3
K	Write technical documentation	4
L	Write user training guide	2
M	Train users on installed hardware and software	1
N	Sign off final system	1

(a) From this data a GANTT chart is constructed.



(i) Complete the GANTT chart by adding activities M and N. [2]

(ii) State the earliest completion date.

Week number .... [1]

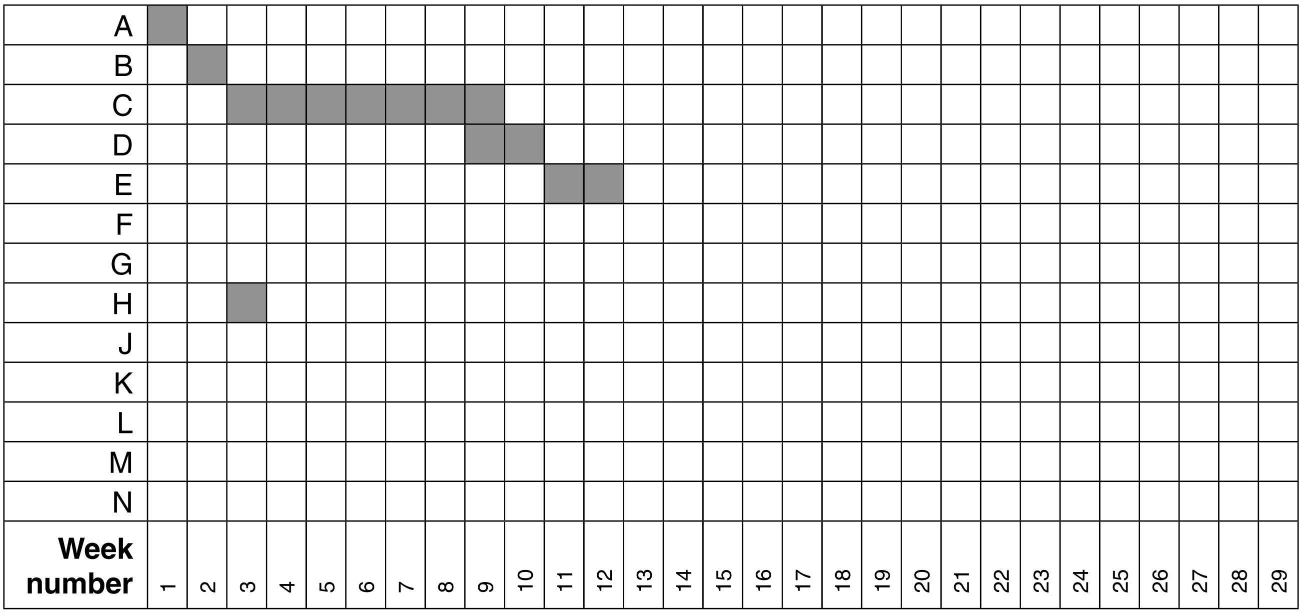
(b) There are problems with the progress of the project:

- Activity E showed that the code contained major errors. The senior programmer now estimates that:
  - further module coding will require another 2 weeks
  - further module testing will require another 2 weeks
  - further integration testing will require another 2 weeks
- The hardware delivery is delayed by 16 weeks.

A revised GANTT chart is now required.

(i) Complete the chart in the grid below.

Activity



[9]

(ii) State the new estimated completion date.

Week number.. [1]

2 A declarative programming language is used to represent the following facts and rules:

```
01  male(ahmed).
02  male(raul).
03  male(ali).
04  male(philippe).
05  female(aisha).
06  female(gina).
07  female(meena).
08  parent(ahmed, raul).
09  parent(aisha, raul).
10  parent(ahmed, philippe).
11  parent(aisha, philippe).
12  parent(ahmed, gina).
13  parent(aisha, gina).
14  mother(A, B) IF female(A) AND parent(A, B).
```

These clauses have the following meaning:

Clause	Explanation
`01`	Ahmed is male
`05`	Aisha is female
`08`	Ahmed is a parent of Raul
`14`	`A` is the mother of `B` if `A` is female and `A` is a parent of `B`

(a) More facts are to be included.

Ali and Meena are the parents of Ahmed.

Write the additional clauses to record this.

`15` ....

`16` .... [2]

(b) Using the variable `C`, the goal

```
parent(ahmed, C)
```

returns

```
C =  raul, philippe, gina
```

Write the result returned by the goal

```
parent(P, gina)
```

`P` = ... [2]

(c) Use the variable `M` to write the goal to find the mother of Gina.

.... [1]

(d) Write the rule to show that `F` is the father of `C`.

```
father(F, C)
```

`IF` ....

.... [2]

(e) Write the rule to show that `X` is a brother of `Y`.

```
brother(X, Y)
```

`IF` ....

....

....

....

.... [4]

3 A college has two types of student: full-time and part-time.

All students have their name and date of birth recorded.

A full-time student has their address and telephone number recorded.

A part-time student attends one or more courses. A fee is charged for each course. The number of courses a part-time student attends is recorded, along with the total fee and whether or not the fee has been paid.

The college needs a program to process data about its students. The program will use an object-oriented programming language.

(a) Complete the class diagram showing the appropriate properties and methods.

`Student`
`StudentName: STRING`
....
....
....
`ShowStudentName()`
....
....
....

`FullTimeStudent`
`Address: STRING`
....
....
....
`Constructor()`
`ShowAddress()`
....
....

`PartTimeStudent`
....
....
....
....
....
....
....
....

[7]

(b) Write **program code**:

(i) for the class definition for the superclass `Student`.

Programming language ....

....

....

....

....

....

....

....

.... [2]

(ii) for the class definition for the subclass `FullTimeStudent`.

Programming language ....

....

....

....

....

....

....

....

.... [3]

(iii) to create a new instance of `FullTimeStudent` with:

- identifier: `NewStudent`
- name: A. Nyone
- date of birth: 12/11/1990
- telephone number: 099111

Programming language ....

....

....

....

....

....

.... [3]

4 A dictionary Abstract Data Type (ADT) has these associated operations:

- Create dictionary `(CreateDictionary)`
- Add key-value pair to dictionary `(Add)`
- Delete key-value pair from dictionary `(Delete)`
- Lookup value `(Lookup)`

The dictionary ADT is to be implemented as a two-dimensional array. This stores key-value pairs.

The pseudocode statement

```
DECLARE Dictionary : Array[1:2000, 1:2] OF STRING
```

reserves space for 2000 key-value pairs in array `Dictionary`.

The `CreateDictionary` operation initialises all elements of `Dictionary` to the empty string.

(a) The hashing function `Hash` is to extract the first letter of the key and return the position of this letter in the alphabet. For example `Hash("Action")` will return the integer value 1.
(Note: The ASCII code for the letter A is 65.)

Complete the pseudocode:

```
FUNCTION Hash (..............................................) RETURNS ............................................

   DECLARE Number : INTEGER

   Number ← ..........................................................................................................

   ...................................................................................................................

ENDFUNCTION
```

[5]

(b) The algorithm for adding a new key-value pair to the dictionary is written, using pseudocode, as a procedure.

```
PROCEDURE Add(NewKey : STRING, NewValue : STRING)
   Index ← Hash(NewKey)
   Dictionary[Index, 1] ← NewKey          // store the key
   Dictionary[Index, 2] ← NewValue        // store the value
ENDPROCEDURE
```

An English-German dictionary of Computing terms is to be set up.

(i) Dry-run the following procedure calls by writing the keys and values in the correct elements of `Dictionary`.

```
Add("File", "Datei")
Add("Disk", "Platte")
Add("Error", "Fehler")
Add("Computer", "Rechner")
```

Dictionary

Index	Key	Value
1		
2		
3		
4		
5		
6		
7		
8		
:		
:		
1999		
2000		

[2]

(ii) Another procedure call is made: `Add("Drive", "Laufwerk")`

Explain the problem that occurs when this key-value pair is saved.

....

....

....

.... [2]

(iii) Describe a method to handle the problem identified in **part (b)(ii)**.

....

....

....

....[2]

(iv) Write **pseudocode** to implement the method you described in **part (b) (iii)**. Choose line numbers to indicate where your pseudocode should be inserted in the given pseudocode.

```
10  PROCEDURE Add(NewKey : STRING, NewValue : STRING)
20     Index ← Hash(NewKey)
30     Dictionary[Index, 1] ← NewKey      // store the key
40     Dictionary[Index, 2] ← NewValue    // store the value
50  ENDPROCEDURE
```

....

....

....

....

....

....

....

....

....

.... [4]

Question 5 begins on page 12.

5 The table shows assembly language instructions for a processor which has one general purpose register – the Accumulator (ACC).

Instruction		**Explanation**
Op Code	**Operand**	
LDM	#n	Immediate addressing. Load the number n to ACC
LDD	<address>	Direct addressing. Load the contents of the given address to ACC
STO	<address>	Store the contents of ACC at the given address
ADD	<address>	Add the contents of the given address to the ACC
INC	<register>	Add 1 to the contents of the register
CMP	<address>	Compare the contents of ACC with the contents of <address>
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False
END		Return control to the operating system

(a) (i) Dry-run this assembly language program using the trace table.

500	LDD 512
501	ADD 509
502	STO 512
503	LDD 511
504	INC ACC
505	STO 511
506	CMP 510
507	JPN 500
508	END
509	7
510	3
511	0
512	0

Trace table

	Memory address			
Accumulator	**509**	**510**	**511**	**512**
	7	3	0	0

[5]

(ii) Explain the role address 511 has in this assembly language program.

....

....

....

.... [2]

(b) Using opcodes from the given table, write instructions to set the value at address 509 to 12.

....

....

....

.... [2]

6 A company keeps details of its stock items in a file of records, `StockFile`.

(a) The record fields are the `ProductCode`, the `Price` and the `NumberInStock`.

Write the **program code** to declare the record structure `StockItem`.

Programming language ....

....

....

....

....

....

....

.... [4]

(b) Before records can be read from file `StockFile`, the file needs to be opened.

(i) Complete the pseudocode.

```
01 TRY
02     OPENFILE ......................................................................
03 EXCEPT
04     ...............................................................................
05 ENDTRY
```

[2]

(ii) Explain the reason for including lines `01`, `03`, `04`, `05`.

....

....

....

.... [2]

(c) A stock report program uses a variable of type `StockItem` declared as follows:

```
DECLARE ThisStockItem : Stockitem
```

The program reads each record in the file `StockFile` in turn.

The program outputs the fields `ProductCode` and `NumberInStock` for each record.

Write **pseudocode** for this.

....

....

....

....

....

....

....

.... [4]

BLANK PAGE

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2015**

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (LEG/SW) 95224/2

CAMBRIDGE International Examinations

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

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

Programming language ....

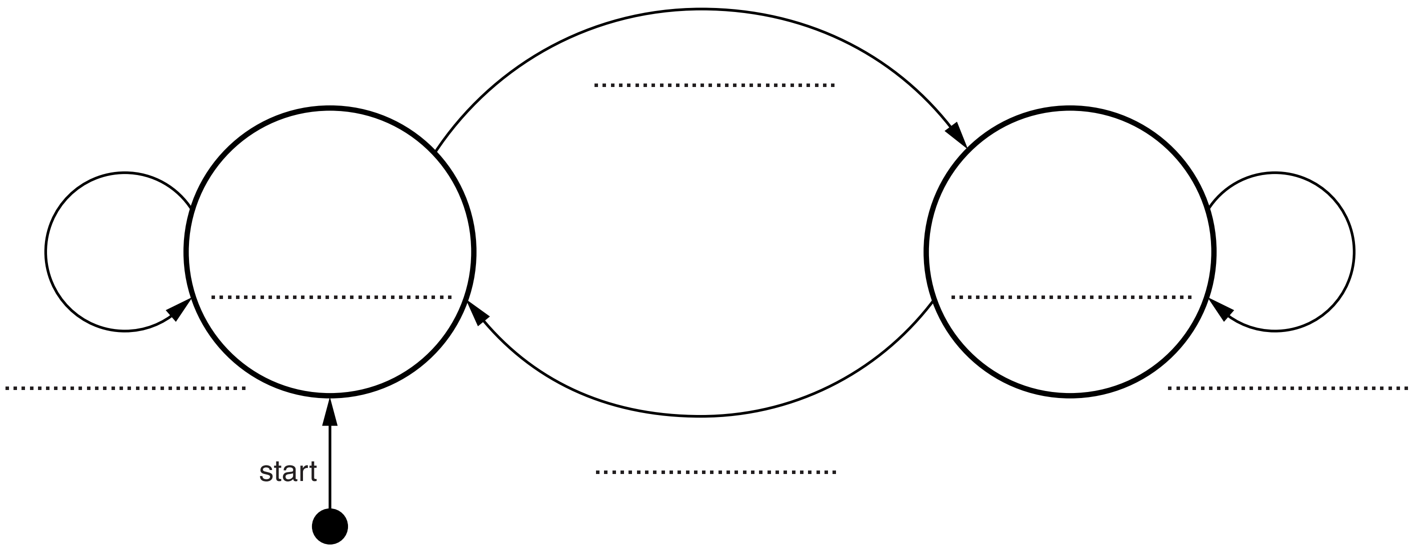
1 A turnstile is a gate which is in a locked state. To open it and pass through, a customer inserts a coin into a slot on the turnstile. The turnstile then unlocks and allows the customer to push the turnstile and pass through the gate.

After the customer has passed through, the turnstile locks again. If a customer pushes the turnstile while it is in the locked state, it will remain locked until another coin is inserted.

The turnstile has two possible states: **locked** and **unlocked**. The transition from one state to another is as shown in the table below.

Current state	Event	Next state
Locked	Insert coin	Unlocked
Locked	Push	Locked
Unlocked	Attempt to insert coin	Unlocked
Unlocked	Pass through	Locked

Complete the state transition diagram for the turnstile:



[5]

2 A declarative programming language is used to represent the knowledge base shown below:

```
01  capital_city(amman).
02  capital_city(beijing).
03  capital_city(brussels).
04  capital_city(cairo).
05  capital_city(london).
06  city_in_country(amman, jordan).
07  city_in_country(shanghai, china).
08  city_in_country(brussels, belgium).
09  city_in_country(london, uk).
10  city_in_country(manchester, uk).
11  country_in_continent(belgium, europe).
12  country_in_continent(china, asia).
13  country_in_continent(uk, europe).
14  city_visited(amman).
15  city_visited(beijing).
16  city_visited(cairo).
```

These clauses have the following meaning:

Clause	Explanation
01	Amman is a capital city
06	Amman is a city in the country of Jordan
11	Belgium is a country in the continent of Europe
14	The travel writer visited Amman

(a) More facts are to be included.

The travel writer visited the city of Santiago which is the capital city of Chile, in the continent of South America.

Write additional clauses to record this.

17 ....

....

18 ....

....

19 ....

....

20 ....

.... [4]

(b) Using the variable `ThisCountry`, the goal

```
country_in_continent(ThisCountry, europe)
```

returns

```
ThisCountry = belgium, uk
```

Write the result returned by the goal:

```
city_in_country(ThisCity, uk)
```

`ThisCity =` ....

.... [2]

(c) Complete the rule below to list the countries the travel writer has visited.

```
countries_visited(ThisCountry)
```

`IF` ....

....

....

....

.... [4]

3 A shop gives some customers a discount on goods totalling more than $20.
The discounts are:

- 5% for goods totalling more than $100
- 5% with a discount card
- 10% with a discount card and goods totalling more than $100

(a) Complete the decision table.

Conditions	goods totalling more than $20	Y	Y	Y	Y	N	N	N	N
	goods totalling more than $100	Y	Y	N	N	Y	Y	N	N
	have discount card	Y	N	Y	N	Y	N	Y	N
Actions	No discount								
	5% discount								
	10% discount								

[4]

(b) Simplify your solution by removing redundancies.

Conditions	goods totalling more than $20								
	goods totalling more than $100								
	have discount card								
Actions	No discount								
	5% discount								
	10% discount								

[5]

(c) The simplified table produced in **part (b)** is used as a design for program code.

The following identifier table shows the parameters to be passed to the function `Discount`. This function returns the discount amount as an integer.

Identifier	**Data type**
`GoodsTotal`	`INTEGER`
`HasDiscountCard`	`BOOLEAN`

Write **program code** for this function.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

4 A payroll program is to be written using an object-oriented programming language. An `Employee` class is designed. Two subclasses have been identified:

- `HourlyPaidEmployee` who is paid a monthly wage calculated from their hourly rate of pay and the number of hours worked during the month
- `SalariedEmployee` who is paid a monthly wage which is one 12th of their annual salary

(a) Draw an inheritance diagram for these classes.

[3]

(b) The design for the `Employee` class consists of:

- properties
  - `EmployeeName`
  - `EmployeeID`
  - `AmountPaidThisMonth`
- methods
  - `SetEmployeeName`
  - `SetEmployeeID`
  - `CalculatePay`

Write **program code** for the class definition of the superclass `Employee`.

Programming language ....

....

....

....

....

....

....

....

....

....

.... [5]

(c) (i) State the properties and/or methods required for the subclass `HourlyPaidEmployee`.

....

....

....

.... [4]

(ii) State the properties and/or methods required for the subclass `SalariedEmployee`.

....

....

....

.... [2]

(d) Name the feature of object-oriented program design that allows the method `CalculatePay` to be declared in the superclass `Employee`.

....

.... [1]

5 Data is stored in the array `NameList[1:10]`. This data is to be sorted.

(a) (i) Complete the pseudocode algorithm for an insertion sort.

```
FOR ThisPointer ← 2 TO ..............................................
   // use a temporary variable to store item which is to
   // be inserted into its correct location
   Temp ← NameList[ThisPointer]
   Pointer ← ThisPointer – 1

   WHILE (NameList[Pointer] > Temp) AND ..............................................
      // move list item to next location
      NameList[....................................] ← NameList[....................................]
      Pointer ← ..............................................
   ENDWHILE

   // insert value of Temp in correct location
   NameList[....................................] ← ..............................................
ENDFOR
```

[7]

(ii) A special case is when `NameList` is already in order. The algorithm in **part (a)(i)** is applied to this special case.

Explain how many iterations are carried out for each of the loops.

....

....

....

....

....

.... [3]

(b) An alternative sort algorithm is a bubble sort:

```
FOR ThisPointer ← 1 TO 9
   FOR Pointer ← 1 TO 9
       IF NameList[Pointer] > NameList[Pointer + 1]
          THEN
             Temp ← NameList[Pointer]
             NameList[Pointer] ← NameList[Pointer + 1]
             NameList[Pointer + 1] ← Temp
       ENDIF
    ENDFOR
ENDFOR
```

(i) As in **part (a)(ii)**, a special case is when `NameList` is already in order. The algorithm in **part (b)** is applied to this special case.

Explain how many iterations are carried out for each of the loops.

....

....

....

.... [2]

(ii) Rewrite the algorithm in **part (b)**, using **pseudocode**, to reduce the number of unnecessary comparisons. Use the same variable names where appropriate.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

6 A queue Abstract Data Type (ADT) has these associated operations:

- create queue
- add item to queue
- remove item from queue

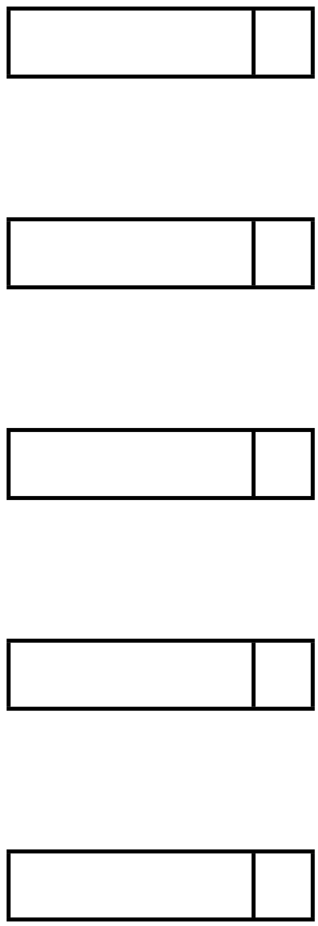
The queue ADT is to be implemented as a linked list of nodes.

Each node consists of data and a pointer to the next node.

(a) The following operations are carried out:

```
CreateQueue
AddName("Ali")
AddName("Jack")
AddName("Ben")
AddName("Ahmed")
RemoveName
AddName("Jatinder")
RemoveName
```

Add appropriate labels to the diagram to show the final state of the queue. Use the space on the left as a workspace. Show your final answer in the node shapes on the right:



[3]

(b) Using pseudocode, a record type, `Node`, is declared as follows:

```
TYPE Node
    DECLARE Name    : STRING
    DECLARE Pointer : INTEGER
ENDTYPE
```

The statement

```
DECLARE Queue : ARRAY[1:10] OF Node
```

reserves space for 10 nodes in array `Queue`.

(i) The `CreateQueue` operation links all nodes and initialises the three pointers that need to be used: `HeadPointer`, `TailPointer` and `FreePointer`.

Complete the diagram to show the value of all pointers after `CreateQueue` has been executed.

HeadPointer

TailPointer

FreePointer

Queue

	Name	Pointer
[1]		
[2]		
[3]		
[4]		
[5]		
[6]		
[7]		
[8]		
[9]		
[10]		

[4]

(ii) The algorithm for adding a name to the queue is written, using pseudocode, as a procedure with the header:

```
PROCEDURE AddName(NewName)
```

where `NewName` is the new name to be added to the queue.

The procedure uses the variables as shown in the identifier table.

Identifier	**Data type**	**Description**
Queue	Array[1:10] OF Node	Array to store node data
NewName	STRING	Name to be added
FreePointer	INTEGER	Pointer to next free node in array
HeadPointer	INTEGER	Pointer to first node in queue
TailPointer	INTEGER	Pointer to last node in queue
CurrentPointer	INTEGER	Pointer to current node

```
PROCEDURE AddName(BYVALUE NewName : STRING)
   // Report error if no free nodes remaining
   IF FreePointer = 0
      THEN
         Report Error
   ELSE
      // new name placed in node at head of free list
      CurrentPointer ← FreePointer
      Queue[CurrentPointer].Name ← NewName
      // adjust free pointer
      FreePointer ← Queue[CurrentPointer].Pointer
      // if first name in queue then adjust head pointer
      IF HeadPointer = 0
         THEN
            HeadPointer ← CurrentPointer
      ENDIF
      // current node is new end of queue
      Queue[CurrentPointer].Pointer ← 0
      TailPointer ← CurrentPointer
   ENDIF
ENDPROCEDURE
```

Complete the **pseudocode** for the procedure `RemoveName`. Use the variables listed in the identifier table.

```
PROCEDURE RemoveName()
   // Report error if Queue is empty
.......................................................................
.......................................................................
.......................................................................
.......................................................................
   OUTPUT Queue[...................................................].Name
   // current node is head of queue
.......................................................................
   // update head pointer
.......................................................................
   // if only one element in queue then update tail pointer
.......................................................................
.......................................................................
.......................................................................
.......................................................................
   // link released node to free list
.......................................................................
.......................................................................
.......................................................................
ENDPROCEDURE
```

[6]

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

0584967126

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2015**

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (SLM) 108502

CAMBRIDGE
International Examinations

[Turn over

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

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

Programming language ....

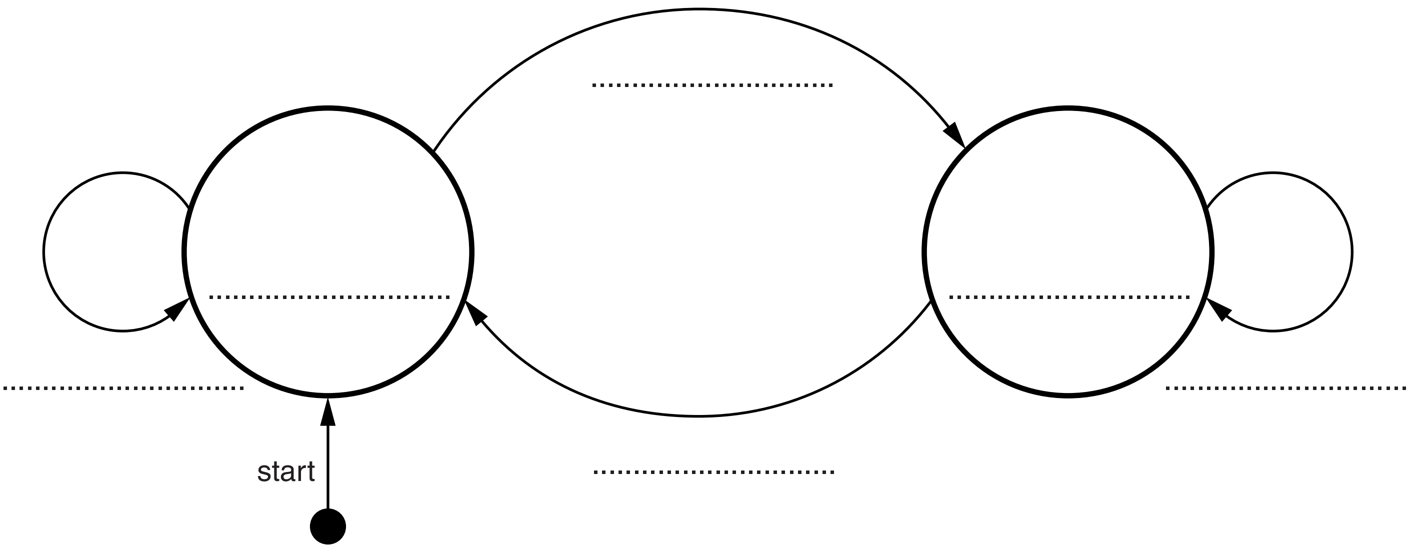
1 A turnstile is a gate which is in a locked state. To open it and pass through, a customer inserts a coin into a slot on the turnstile. The turnstile then unlocks and allows the customer to push the turnstile and pass through the gate.

After the customer has passed through, the turnstile locks again. If a customer pushes the turnstile while it is in the locked state, it will remain locked until another coin is inserted.

The turnstile has two possible states: **locked** and **unlocked**. The transition from one state to another is as shown in the table below.

Current state	Event	Next state
Locked	Insert coin	Unlocked
Locked	Push	Locked
Unlocked	Attempt to insert coin	Unlocked
Unlocked	Pass through	Locked

Complete the state transition diagram for the turnstile:



[5]

2 A declarative programming language is used to represent the knowledge base shown below:

```
01  capital_city(amman).
02  capital_city(beijing).
03  capital_city(brussels).
04  capital_city(cairo).
05  capital_city(london).
06  city_in_country(amman, jordan).
07  city_in_country(shanghai, china).
08  city_in_country(brussels, belgium).
09  city_in_country(london, uk).
10  city_in_country(manchester, uk).
11  country_in_continent(belgium, europe).
12  country_in_continent(china, asia).
13  country_in_continent(uk, europe).
14  city_visited(amman).
15  city_visited(beijing).
16  city_visited(cairo).
```

These clauses have the following meaning:

Clause	Explanation
01	Amman is a capital city
06	Amman is a city in the country of Jordan
11	Belgium is a country in the continent of Europe
14	The travel writer visited Amman

(a) More facts are to be included.

The travel writer visited the city of Santiago which is the capital city of Chile, in the continent of South America.

Write additional clauses to record this.

17 ....

....

18 ....

....

19 ....

....

20 ....

.... [4]

(b) Using the variable `ThisCountry`, the goal

```
country_in_continent(ThisCountry, europe)
```

returns

```
ThisCountry = belgium, uk
```

Write the result returned by the goal:

```
city_in_country(ThisCity, uk)
```

`ThisCity =` ....

.... [2]

(c) Complete the rule below to list the countries the travel writer has visited.

```
countries_visited(ThisCountry)
```

`IF` ....

....

....

....

.... [4]

3 A shop gives some customers a discount on goods totalling more than $20.
The discounts are:

- 5% for goods totalling more than $100
- 5% with a discount card
- 10% with a discount card and goods totalling more than $100

(a) Complete the decision table.

Conditions	goods totalling more than $20	Y	Y	Y	Y	N	N	N	N
	goods totalling more than $100	Y	Y	N	N	Y	Y	N	N
	have discount card	Y	N	Y	N	Y	N	Y	N
Actions	No discount								
	5% discount								
	10% discount								

[4]

(b) Simplify your solution by removing redundancies.

Conditions	goods totalling more than $20								
	goods totalling more than $100								
	have discount card								
Actions	No discount								
	5% discount								
	10% discount								

[5]

(c) The simplified table produced in **part (b)** is used as a design for program code.

The following identifier table shows the parameters to be passed to the function `Discount`. This function returns the discount amount as an integer.

Identifier	**Data type**
`GoodsTotal`	`INTEGER`
`HasDiscountCard`	`BOOLEAN`

Write **program code** for this function.

Programming language ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

4 A payroll program is to be written using an object-oriented programming language. An `Employee` class is designed. Two subclasses have been identified:

- `HourlyPaidEmployee` who is paid a monthly wage calculated from their hourly rate of pay and the number of hours worked during the month
- `SalariedEmployee` who is paid a monthly wage which is one 12th of their annual salary

(a) Draw an inheritance diagram for these classes.

[3]

(b) The design for the `Employee` class consists of:

- properties
  - `EmployeeName`
  - `EmployeeID`
  - `AmountPaidThisMonth`

- methods
  - `SetEmployeeName`
  - `SetEmployeeID`
  - `CalculatePay`

Write **program code** for the class definition of the superclass `Employee`.

Programming language ....

....

....

....

....

....

....

....

....

....

.... [5]

(c) (i) State the properties and/or methods required for the subclass `HourlyPaidEmployee`.

....

....

....

.... [4]

(ii) State the properties and/or methods required for the subclass `SalariedEmployee`.

....

....

....

.... [2]

(d) Name the feature of object-oriented program design that allows the method `CalculatePay` to be declared in the superclass `Employee`.

....

.... [1]

5 Data is stored in the array `NameList[1:10]`. This data is to be sorted.

(a) (i) Complete the pseudocode algorithm for an insertion sort.

```
FOR ThisPointer ← 2 TO .................................................
   // use a temporary variable to store item which is to
   // be inserted into its correct location
   Temp ← NameList[ThisPointer]
   Pointer ← ThisPointer – 1

   WHILE (NameList[Pointer] > Temp) AND .................................................
      // move list item to next location
      NameList[....................................] ← NameList[....................................]
      Pointer ← .................................................
   ENDWHILE

   // insert value of Temp in correct location
   NameList[....................................] ← .................................................
ENDFOR
```

[7]

(ii) A special case is when `NameList` is already in order. The algorithm in **part (a)(i)** is applied to this special case.

Explain how many iterations are carried out for each of the loops.

....

....

....

....

....

.... [3]

(b) An alternative sort algorithm is a bubble sort:

```
FOR ThisPointer ← 1 TO 9
   FOR Pointer ← 1 TO 9
       IF NameList[Pointer] > NameList[Pointer + 1]
          THEN
             Temp ← NameList[Pointer]
             NameList[Pointer] ← NameList[Pointer + 1]
             NameList[Pointer + 1] ← Temp
       ENDIF
    ENDFOR
ENDFOR
```

(i) As in **part (a)(ii)**, a special case is when `NameList` is already in order. The algorithm in **part (b)** is applied to this special case.

Explain how many iterations are carried out for each of the loops.

....

....

....

.... [2]

(ii) Rewrite the algorithm in **part (b)**, using **pseudocode**, to reduce the number of unnecessary comparisons. Use the same variable names where appropriate.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

6 A queue Abstract Data Type (ADT) has these associated operations:

- create queue
- add item to queue
- remove item from queue

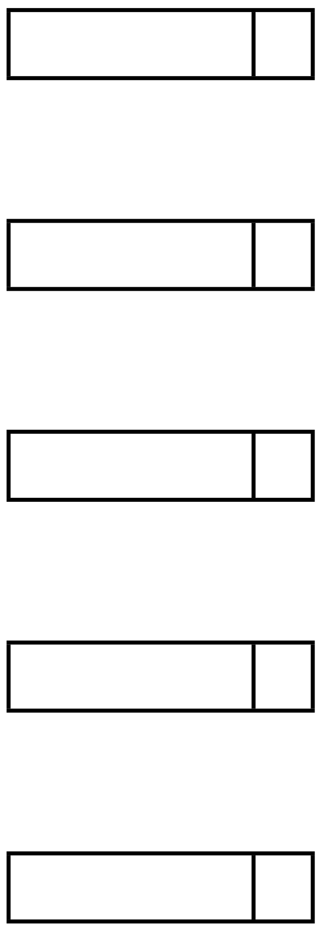
The queue ADT is to be implemented as a linked list of nodes.

Each node consists of data and a pointer to the next node.

(a) The following operations are carried out:

```
CreateQueue
AddName("Ali")
AddName("Jack")
AddName("Ben")
AddName("Ahmed")
RemoveName
AddName("Jatinder")
RemoveName
```

Add appropriate labels to the diagram to show the final state of the queue. Use the space on the left as a workspace. Show your final answer in the node shapes on the right:



[3]

(b) Using pseudocode, a record type, `Node`, is declared as follows:

```
TYPE Node
    DECLARE Name    : STRING
    DECLARE Pointer : INTEGER
ENDTYPE
```

The statement

```
DECLARE Queue : ARRAY[1:10] OF Node
```

reserves space for 10 nodes in array `Queue`.

(i) The `CreateQueue` operation links all nodes and initialises the three pointers that need to be used: `HeadPointer`, `TailPointer` and `FreePointer`.

Complete the diagram to show the value of all pointers after `CreateQueue` has been executed.

HeadPointer

TailPointer

FreePointer

Queue

	Name	Pointer
[1]		
[2]		
[3]		
[4]		
[5]		
[6]		
[7]		
[8]		
[9]		
[10]		

[4]

(ii) The algorithm for adding a name to the queue is written, using pseudocode, as a procedure with the header:

```
PROCEDURE AddName(NewName)
```

where `NewName` is the new name to be added to the queue.

The procedure uses the variables as shown in the identifier table.

Identifier	**Data type**	**Description**
`Queue`	`Array[1:10] OF Node`	Array to store node data
`NewName`	`STRING`	Name to be added
`FreePointer`	`INTEGER`	Pointer to next free node in array
`HeadPointer`	`INTEGER`	Pointer to first node in queue
`TailPointer`	`INTEGER`	Pointer to last node in queue
`CurrentPointer`	`INTEGER`	Pointer to current node

```
PROCEDURE AddName(BYVALUE NewName : STRING)
   // Report error if no free nodes remaining
   IF FreePointer = 0
      THEN
         Report Error
   ELSE
      // new name placed in node at head of free list
      CurrentPointer ← FreePointer
      Queue[CurrentPointer].Name ← NewName
      // adjust free pointer
      FreePointer ← Queue[CurrentPointer].Pointer
      // if first name in queue then adjust head pointer
      IF HeadPointer = 0
         THEN
            HeadPointer ← CurrentPointer
      ENDIF
      // current node is new end of queue
      Queue[CurrentPointer].Pointer ← 0
      TailPointer ← CurrentPointer
   ENDIF
ENDPROCEDURE
```

Complete the **pseudocode** for the procedure `RemoveName`. Use the variables listed in the identifier table.

```
PROCEDURE RemoveName()
   // Report error if Queue is empty
..............................................................................................
..............................................................................................
..............................................................................................
..............................................................................................
   OUTPUT Queue[..........................................................].Name
   // current node is head of queue
..............................................................................................
   // update head pointer
..............................................................................................
   // if only one element in queue then update tail pointer
..............................................................................................
..............................................................................................
..............................................................................................
..............................................................................................
   // link released node to free list
..............................................................................................
..............................................................................................
..............................................................................................
ENDPROCEDURE
```

[6]

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME		
CENTRE NUMBER		CANDIDATE NUMBER

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2020**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Blank pages are indicated.

DC (JP) 199348

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of +192.5 in this system. Show your working.

Mantissa **Exponent**

Working ....

....

....

....

.... [3]

(b) Calculate the normalised floating-point representation of –192.5 in this system. Show your working.

Mantissa **Exponent**

Working ....

....

....

....

.... [3]

(c) The floating-point representation has changed. There are now 12 bits for the mantissa and 4 bits for the exponent as shown.

Mantissa													**Exponent**			

Explain why +192.5 cannot be accurately represented in this format.

....

....

....

....

....

.... [3]

2 The diagram shows four files and three methods of file organisation.

Draw **one** line to match each file with its most appropriate method of file organisation.

File	**File organisation**
Text file	Sequential
File for recording the temperature every hour	Random
Master file for paying each employee every month	Serial
Customer user name and password file	

[4]

3 A mobile phone company uses circuit switching for voice calls and packet switching to send and receive other data.

(a) (i) Describe circuit switching.

....

....

....

....

....

.... [3]

(ii) Explain why the company uses circuit switching for voice calls.

....

....

....

.... [2]

(b) (i) Describe packet switching.

....

....

....

....

....

.... [3]

(ii) Explain why the company uses packet switching to send and receive other data.

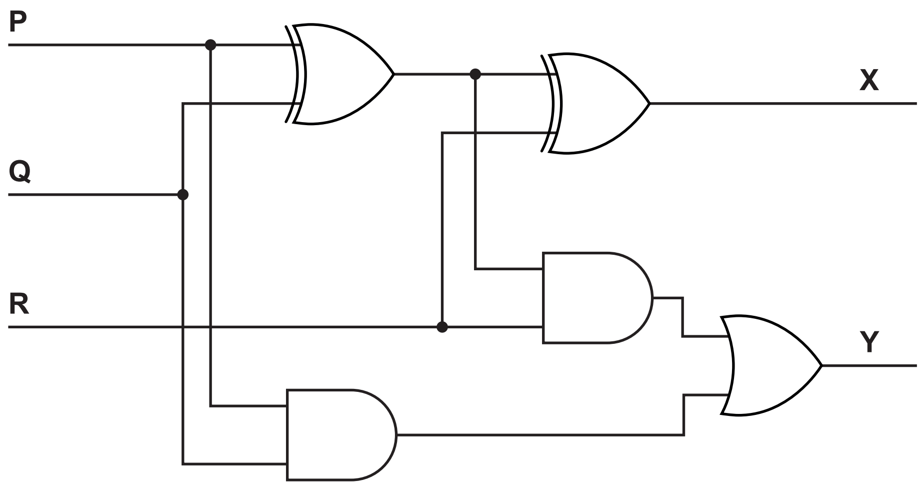
....

....

....

.... [2]

4 (a) Write the Boolean algebraic expressions for the following logic circuit.



X = ...

....

Y = ...

....
[5]

(b) The logic circuit given in **part (a)** is a full adder.

(i) Give the purpose of outputs **X** and **Y** in this circuit.

X ....

Y ....
[2]

(ii) Give the use of the input **R** in this circuit.

.... [1]

5 Complete these **three** statements about computer processors.

A processor with a few simple fixed-length instructions that have a small number of instruction formats is called a .... processor.

A processor with many complex variable-length instructions that has many instruction formats is called a .... processor.

Instruction-level parallelism, applied to the execution of instructions during the fetch-execute cycle, is called ....

[3]

6 Duraid writes a short program in a high-level programming language. An interpreter executes the program.

The following is part of Duraid's program.

```
DECLARE P, Q, R, X, Y : INTEGER
CONSTANT M = 10

P = 4
Q = 2
R = 1
X = (P + Q) * (P – Q)
Y = (M / Q) * (P + Q - R)
```

(a) Write the Reverse Polish Notation (RPN) for the following expression from Duraid's program.

`(P + Q) * (P – Q)`

.... [2]

(b) The interpreter is executing Duraid's program. The expressions are in infix form.

The interpreter converts the infix to RPN.

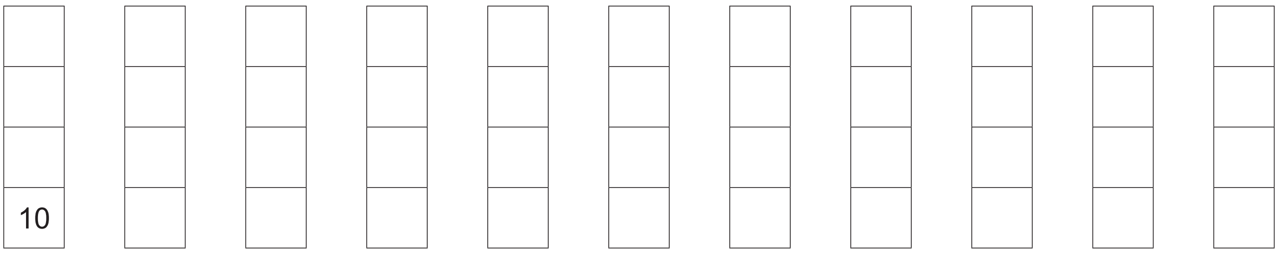
The RPN expression for `Y` is:

`M Q / P Q + R – *`

The interpreter evaluates this RPN expression using a stack.

(i) Show the changing contents of the stack, as the interpreter evaluates the expression for `Y`.

Use the values of the variables and constant given in the program. The first entry has been done for you.



[4]

(ii) Convert the following RPN expression back to its infix form.

```
P Q + M * R P - -
```

....

.... [2]

(c) Explain how RPN is used by an interpreter to evaluate expressions.

....

....

....

.... [2]

7 A computer at a remote weather station is performing three tasks:

- measuring and recording the temperature every 10 seconds
- measuring and recording the wind speed every 10 seconds
- sending the previous day's temperature and wind speed readings to a scientist at another location via the Internet.

The operating system is managing the multitasking of these tasks.

(a) At one point in time:

- the temperature measuring and recording task is idle
- the wind speed is being recorded
- the task to send the previous day's temperature and wind speed readings is waiting for an internet connection.

Identify the process state for each task. Give a reason why each task is in that process state.

Temperature measuring and recording process state ....

Reason ....

....

....

Wind speed measuring and recording process state ....

Reason ....

....

....

Sending process state ....

Reason ....

....

....

[6]

(b) The weather station computer uses an operating system.

Explain how this operating system uses interrupts to schedule the measuring and recording tasks.

....

....

....

....

....

....

....

.... [4]

8 Martha wants to send a private message to Joshua over the Internet.

(a) Martha and Joshua's computers have already exchanged digital certificates.

Identify **three** items that could be contained in a digital certificate.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) Joshua and Martha's digital certificates are used to ensure that Martha's message has not been altered during transmission.

Explain how asymmetric encryption uses the contents of the digital certificates to ensure that the message has not been altered during transmission.

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

9 A train cannot move if any of the eight automatic train doors are open. The train door monitoring system, set out below, checks that all the doors are closed before the train can move.

- If a monitoring system detects that a door is open, it sets a specific bit in address `500` to `1`.
- If the bit for door one is equal to `1`, the binary value for hexadecimal `FF` is sent to address `501`. The contents of address `501` are changed to make door 1's light flash when the door is open.
- If the bit for door two is equal to `1`, the binary value for hexadecimal `FF` is sent to address `502`. The contents of address `502` are changed to make door 2's light flash when the door is open.

  This is repeated for each door from 3 to 8.
- Each door sets its bit in address `500` to zero when the door closes, and the contents of the corresponding door address are set to zero.
- The train manager can identify which door is open from the flashing light.

The current contents of address `500` are:

	Door number							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Address `500`	1	0	0	1	0	0	1	0

(a) Complete the following table by writing the values stored in addresses `503` to `508`. Use the contents of address `500` shown above. Note that addresses `501` and `502` are complete.

`501`	1	1	1	1	1	1	1	1	Door 1
`502`	0	0	0	0	0	0	0	0	Door 2
`503`									Door 3
`504`									Door 4
`505`									Door 5
`506`									Door 6
`507`									Door 7
`508`									Door 8

[2]

(b) The following table shows assembly language instructions for the processor controlling the train door monitoring system that has one general purpose register, the Accumulator (ACC).

	Instruction		Explanation
Label	**Op code**	**Operand**	
	LDM	&n	Load the hexadecimal number n to ACC
	LDD	<address>	Load the contents of the location at the given address to ACC
	STO	<address>	Store the contents of ACC at the given address
	AND	&n	Bitwise AND the contents of ACC with the hexadecimal number n
	CMP	&n	Compare the contents of ACC with the hexadecimal number n
	JPE	<address>	Following a compare instruction, jump to <address> or <label> if the compare was True
<label>:	<op code>	<operand>	Labels an instruction
	WAIT		Macro to wait one second before the next instruction is executed

After rechecking the doors, address `500` now contains `10101010`.

(i) Complete the table by writing the values of the Accumulator (ACC) and the contents of address `501` as these instructions are executed **once** to check door **1**.

	Instruction		ACC	501
Label	**Op code**	**Operand**		
CHECK1:	LDD	500		
	AND	&80		
	CMP	&00		
	JPE	DOOR1		
	LDM	&FF		
DOOR1:	STO	501		
	WAIT			
	LDM	&00		
	STO	501		
	WAIT			
	JMP	CHECK1		

[4]

(ii) Write the assembly language instructions to check door **2**.

	Instruction	
Label	**Op code**	**Operand**

[4]

(c) Explain how the check door routines show a flashing light or no light.

....

....

....

.... [2]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

1747453880

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2018**

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 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.
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.

The maximum number of marks is 75.

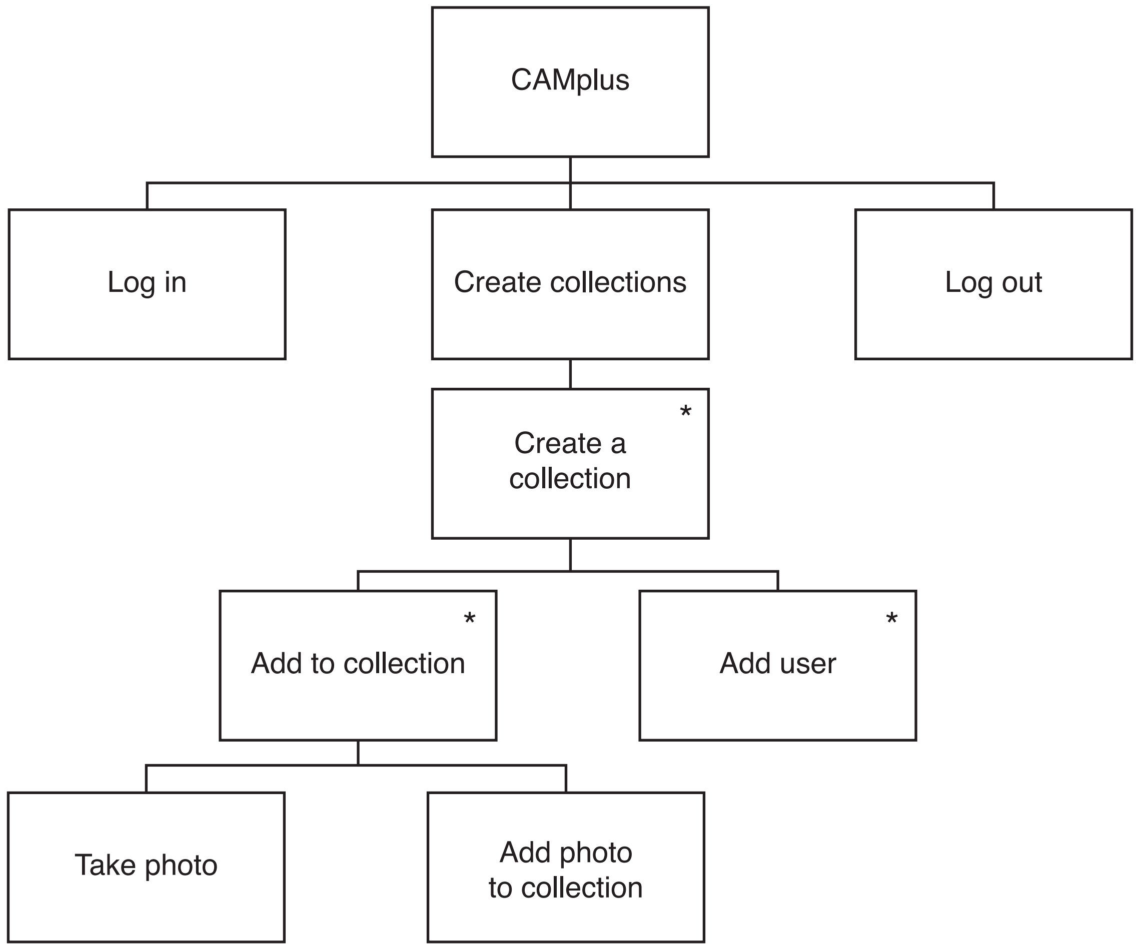
This document consists of **22** printed pages and **2** blank pages.

DC (NH/SW) 148546/2

1 Paul is using an application (app) called CAMplus. The app allows users to:

- log in
- create a new collection of photographs
- use the camera to take new photographs
- automatically add new photographs to the new collection
- share the new collection with other users
- start another collection or log out of the app.

The following JSP structure diagram represents the operation of CAMplus.



(a) An algorithm has been written in pseudocode to represent the **Create collections** operation from the JSP structure diagram. The algorithm is incomplete.

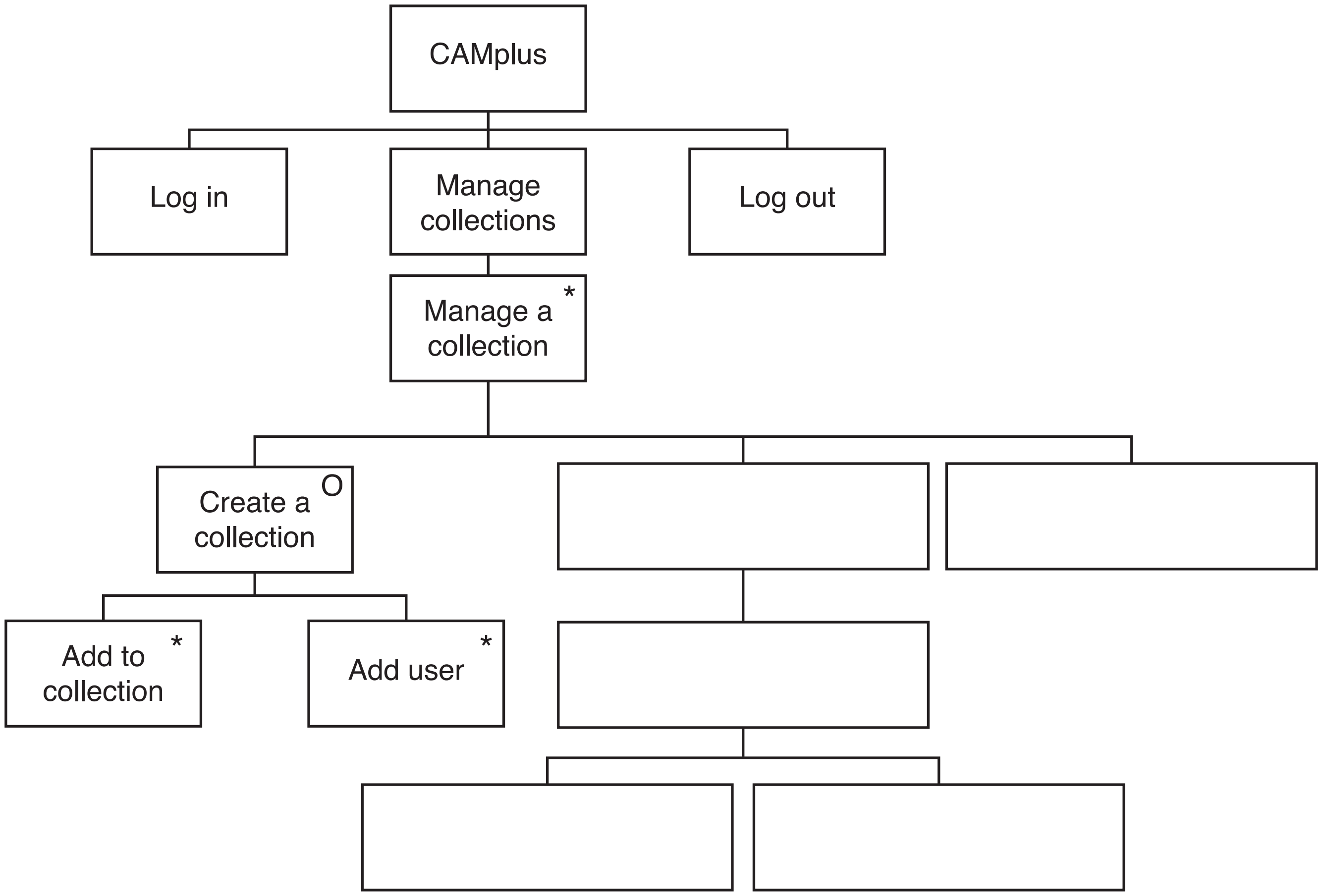
Write **pseudocode** to complete this algorithm.

```
REPEAT

   REPEAT

      CALL TakePhoto

      ..........................................................................

      OUTPUT "Do you want to take another photo?"

      INPUT AddPhoto

   UNTIL AddPhoto = "No"

   REPEAT

      ..........................................................................

      OUTPUT "Do you want to add another user?"

      INPUT NewUser

   UNTIL .................................... = "No"

   OUTPUT "Do you want to create another collection?"

   .............................................................................

UNTIL NewCollection = "No"
```

[4]

(b) The app is updated. Paul can now add and delete photos from chosen collections. Paul can also delete collections.

Complete the JSP structure diagram to show the changes.



[5]

Question 2 begins on the next page.

2 A declarative language is used to represent the following facts and rules about iguanas and lizards.

```
01   has(reptile, cold_blood).
02   has(reptile, air_breathing).
03   has(reptile, scales).
04
05   is_a(squamata, reptile).
06   is_a(iguana, squamata).
07   is_a(lizard, squamata).
08   is_a(green_iguana, iguana).
09   is_a(cayman, iguana).
10   is_a(smooth_iguana, iguana).
11
12   maxsize(green_iguana, 152).
13   maxsize(cayman, 90).
14   maxsize(smooth_iguana, 70).
```

These clauses have the following meaning:

Clause	**Explanation**
`01`	A reptile has cold blood.
`09`	A cayman is a type of iguana.
`12`	The maximum size of a green iguana is 152 cm.

(a) More facts are to be included.

A gecko is a type of lizard. It has a maximum size of 182 cm.

Write the additional clauses to record these facts.

`15` ....

`16` ....
[2]

(b) Using the variable `R`, the goal

```
is_a(R, squamata).
```

returns

```
R = iguana, lizard
```

Write the result returned by the goal

```
is_a(T, iguana).
```

`T =` ....[2]

(c) Write the goal, using the variable `X`, to find what a squamata is.

....[2]

(d) All iguanas and lizards are squamata. All squamata are reptiles.

Write a recursive rule to make all lizards and iguanas inherit the properties of reptiles.

```
has(X, Y)

IF
```

....

....[3]

(e) State what the following goal returns.

```
NOT(maxsize(cayman, 70)).
```

....[1]

3 The arrays PollData[1:10] and CardData[1:10] store data.

PollData	12	85	52	57	25	11	33	59	56	91
CardData	11	12	25	33	52	56	57	59	91	85

An **insertion sort** sorts these data.

(a) State why it will take less time to complete an insertion sort on CardData than on PollData.

....

....[1]

(b) The following pseudocode algorithm performs an insertion sort on the CardData array.

Complete the following **pseudocode** algorithm.

```
ArraySize ← 10
FOR Pointer ← 2 TO ..............................................................
  ValueToInsert ← CardData[Pointer]
  HolePosition ← ...................................................................
  WHILE (HolePosition > 1 AND (.......................... > ...................))
    CardData[HolePosition] ← CardData[.....................................]
    HolePosition ← ...............................................................
  ENDWHILE
  CardData[HolePosition] ← ....................................................
ENDFOR
```

[7]

(c) **(i)** A binary search algorithm is used to find a specific value in an array.

Explain why an array needs to be sorted before a binary search algorithm can be used.

....

....

....

....

....

....

....

....[2]

(ii) The current contents of `CardData` are shown.

11	12	25	33	52	56	57	59	85	91

Explain how a binary search will find the value `25` in `CardData`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[4]

(d) Complete this procedure to carry out a binary search on the array shown in **part (c)(ii)**.

```
PROCEDURE BinarySearch(CardData, SearchValue)

   DECLARE Midpoint : INTEGER

   First ← 1

   Last ← ARRAYLENGTH(.............................................)

   Found ← FALSE

   WHILE (First <= Last) AND NOT(Found)

      Midpoint ← ..................................................

      IF CardData[Midpoint] = SearchValue

         THEN

            Found ← TRUE

         ELSE

            IF SearchValue < CardData[Midpoint]

               THEN

                  Last ← ..........................................

               ELSE

                  First ← .........................................

            ENDIF

         ENDIF

   ENDWHILE

ENDPROCEDURE
```

[4]

Question 4 begins on the next page.

4 X-Games is an international extreme sports competition.

A program will store and process data about the teams in the competition.

- Each team is made up of members.
- Members can be added and removed from each team.
- Each member has a first name, last name, date of birth and gender.
- Each member can be an official or a competitor.
- Each official has a job title and may be first-aid trained.
- Each competitor takes part in one sport.

The program is written using object-oriented programming.

The program can output the full name and date of birth of any member. For example, “Nadia Abad 16/05/1995”

An introduction about a team member can be output using their name. For example, “Hello, I’m Nadia Abad”.

The program outputs a different version of the introduction for a competitor. This version includes the competitor’s sport. For example, “Hello, I’m Sally Jones and my sport is Skateboard Park.”

(a) Complete the following class diagram to show the attributes, methods and inheritance for the program.

You do not need to write the get and set methods.

Member
FirstName : STRING LastName : STRING DateOfBirth : DATE Gender : STRING
Constructor() Introduction() DisplayFullnameAndDateOfBirth()

Team
TeamName : STRING TeamList : ARRAY OF Member
Constructor() ....

Competitor
Sport : STRING
Constructor() Introduction()

Official
....
Constructor() DisplayJobTitle()

[3]

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

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

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

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(d) Omar Ellaboudy is an official at X-Games. He is first-aid trained and his job title is Judge. He is male and was born on 17/03/1993.

Write **program code** to create an instance of an object with the identifier `BMXJudge`. All attributes of the instance must be fully initialised.

Programming language ....

Program code

....

....

....

....

....

....[3]

Question 5 begins on the next page.

5 A company is developing an application program. The project manager has been asked to create a work breakdown schedule for the project as follows:

	Activity	Days to complete	Predecessor activity
A	Gather User Requirements	6	
B	Design work	4	A
C	Develop server code	4	B
D	Develop application code	5	B
E	User Interface Development	6	B
F	Test server code	2	C
G	Test application	2	D, E
H	Test application/server integration	6	F, G
I	Roll out mobile application	6	H

(a) A GANTT chart is created from the work breakdown schedule. Activities **A** and **B** have already been added to the chart.

Complete the GANTT chart.

Activity																														
A																														
B																														
C																														
D																														
E																														
F																														
G																														
H																														
I																														
Day number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**

[5]

(b) State which activities can run in parallel on the following days.

(i) Day 14

....[1]

(ii) Day 16

....[1]

(c) Explain how the project manager will use the GANTT chart to make sure the project is completed on time.

....

....

....

....[2]

6 An Abstract Data Type (ADT) is used to create an unordered binary tree. The binary tree is created as an array of nodes. Each node consists of a data value and two pointers.

A record type, `Node`, is declared using pseudocode.

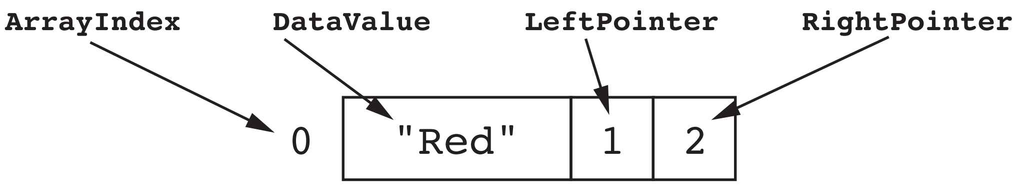
```
TYPE Node
   DECLARE DataValue : STRING
   DECLARE LeftPointer : INTEGER
   DECLARE RightPointer : INTEGER
ENDTYPE
```

The following statement declares an array `BinaryTree`.

```
   DECLARE BinaryTree : ARRAY[0:14] OF Node
```

A variable, `NextNode`, points to the next free node.

The following diagram shows a possible node.



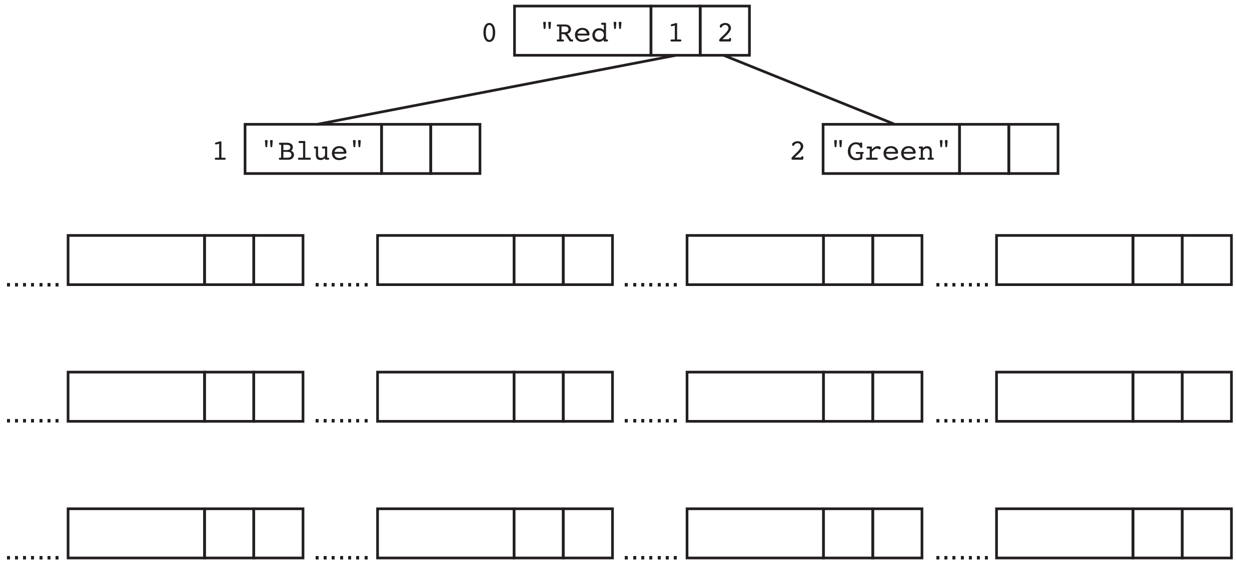
The commands in the following table create and add nodes to the binary tree.

Command	Comment
`CreateTree(NodeData)`	Sets `NextNode` to `0`. Writes `NodeData` into `DataValue` at the position `NextNode` Updates `NextNode` using `NextNode` = `NextNode + 1`
`AttachLeft(NodeData, ParentNode)`	Writes `NodeData` into `DataValue` of `NextNode` Sets the `LeftPointer` of node `ParentNode` to `NextNode` Updates `NextNode` using `NextNode` = `NextNode + 1`
`AttachRight(NodeData, ParentNode)`	Writes `NodeData` into `DataValue` of `NextNode` Sets the `RightPointer` of node `ParentNode` to `NextNode` Updates `NextNode` using `NextNode` = `NextNode + 1`

(a) The following commands are executed.

```
CreateTree("Red")
AttachLeft("Blue", 0)
AttachRight("Green", 0)
```

The following diagram shows the current state of the binary tree.



Write on the diagram to show the state of the binary tree after the following commands have been executed.

```
AttachRight("Black", 2)
AttachLeft("Brown", 2)
AttachLeft("Peach", 3)
AttachLeft("Yellow", 1)
AttachRight("Purple", 1)
AttachLeft("White", 6)
AttachLeft("Pink", 7)
AttachLeft("Grey", 9)
AttachRight("Orange", 9)
```

[5]

(b) A new command has been added to initialise the pointers of the binary tree to `–1` to indicate they are not in use.

A leaf is a node of the binary tree which has no children. In the case of this binary tree, a node with a `LeftPointer` of `–1` and a `RightPointer` of `–1` is a leaf.

Write a **recursive** function, in **program code**, to traverse the binary tree and output the value of `DataValue` for each leaf node.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[8]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

CANDIDATE NAME										
CENTRE NUMBER						CANDIDATE NUMBER				

1710814632

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2020**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages. Blank pages are indicated.

DC (LK) 188581/3

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) The following floating-point number stored is not normalised.

Calculate the denary value for the floating-point number. Show your working.

Mantissa

0	0	0	0	1	1	0	0	0	0	0	0

Exponent

0	1	0	1

Working ....

....

....

....

Denary value ....
[3]

(b) **(i)** Normalise the floating-point number given in **part (a)**.

Write your answer in the following boxes.

Mantissa

Exponent

[2]

(ii) Describe **one** problem that can occur when floating-point numbers are not normalised.

....

....

.... [2]

2 Data types can be classified as composite or non-composite.

A record is declared of type `box` using the following pseudocode.

```
TYPE size = (small, medium, large)

TYPE box

   DECLARE volume : size

   DECLARE price : REAL

   DECLARE colour : STRING

ENDTYPE

DECLARE myBox : ARRAY [1:6] OF box
```

(a) **(i)** Identify **one** composite and **three** non-composite data types used in the pseudocode.

Composite data type ....

Non-composite data type 1 ....

Non-composite data type 2 ....

Non-composite data type 3 ....
[4]

(ii) Identify the data type in the pseudocode that is enumerated.

.... [1]

(b) A box is red, with medium volume and a price of $10.99.

Write **pseudocode** to store the details of this box in the first element of the array.

....

....

....

....

.... [3]

3 The use of the TCP/IP protocol suite is essential for successful communication over the Internet.

(a) **(i)** Describe the TCP/IP protocol suite.

....

....

....

....

....

....

....

....

.... [5]

(ii) A group of over 100 students has produced a movie. The size of the movie file is very large.

The students would like to use peer-to-peer file sharing to share this file with friends and family.

Identify the **most appropriate** TCP/IP protocol for sharing this file over the Internet **and** describe the way this protocol works.

Protocol ....

Description ....

....

....

....

....

....

....

....

[5]

(b) **(i)** Files shared over the Internet are sent using packet switching or circuit switching methods.

Identify **and** describe the **most suitable** method for the large movie file from **part (a)(ii)**.

Method ....

Description ....

....

....

....

....

....

.... [4]

(ii) State **one** benefit and **one** drawback of the method you identified in **part (b)(i)**.

Benefit ....

....

Drawback ....

.... [2]

4 The following truth table represents a logic circuit with three inputs and two outputs.

INPUT			**OUTPUT**	
A	**B**	**C**	**X**	**Y**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(a) Write the Boolean expressions for the truth table as sum-of-products.

X = ...

....

Y = ...

.... [4]

(b) Complete the Karnaugh Maps (K-maps) for the truth table.

OUTPUT X
AB

		00	**01**	**11**	**10**
C	**0**				
	1				

OUTPUT Y
AB

		00	**01**	**11**	**10**
C	**0**				
	1				

[2]

(c) The K-maps can be used to simplify **one** of the expressions in **part (a)**.

(i) Draw loop(s) around appropriate group(s) of 1s to produce an optimal sum-of-products for the single output table that can be simplified in **part (b)**. [3]

(ii) Write the simplified sum-of-products expressions for this output from **part (c)(i)**.

.... [3]

(d) Identify the common logic circuit given by the truth table in **part (a)**. Give the use of each output.

Logic circuit ....

Use of **X** ....

Use of **Y** ....
[3]

5 Complete these statements about a virtual machine.

A virtual machine is .... that emulates a

.... computer system.

A virtual machine allows multiple .... operating systems to run

on one computer using a .... operating system.
[4]

6 Anita is studying computer science and she is confused about some of the computer security terminology as some of the words are similar.

Anita wants to know the similarities (features that are the same) and differences (features that are different) between some of the terms.

(a) Give the similarities **and** differences between a **public key** and a **private key**.

Similarities ....

....

....

....

....

Differences ....

....

....

....

....
[4]

(b) Give the similarities **and** differences between a **digital certificate** and a **digital signature**.

Similarities ....

....

....

....

....

Differences ....

....

....

....

....
[4]

(c) Give the similarities **and** differences between **phishing** and **pharming**.

Similarities ....

....

....

....

....

Differences ....

....

....

....

....
[4]

BLANK PAGE

7 A company has a number of lorries that deliver items around the country. The items in each lorry are its load. Each lorry has a monitoring system that provides information to the driver about the state of the load and other data from each trip.

- Data is stored in three memory locations with addresses `801` to `803`.
- Location `801` contains the distance travelled in kilometres for the current trip, stored as a binary integer.
- Location `802` contains the quantity of fuel used in litres for the current trip, stored as a fixed-point binary number with six places before the binary point and two places after the binary point.
- The four most significant bits of location `803` are flags used to identify problems with the load, for example it is too heavy. A flag is set to 1 if there is a problem, or 0 if not.
  The problems are:
    - Bit 7 – load too heavy
    - Bit 6 – load too high
    - Bit 5 – load unstable
    - Bit 4 – load not secured (risk of the load falling off)
    - Bits 0 to 3 are not used

(a) The current contents of addresses `801` to `803` are:

	Most significant bit							Least significant bit
801	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**0**
802	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**
803	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**

State the information that the current contents of addresses `801` to `803` will provide to the driver.

....

....

....

.... [3]

(b) A lorry has a load that is too heavy and is not secured. It has travelled 120 kilometres and used 35.25 litres of fuel.

Complete the contents of the addresses to record this information.

801								
802								
803								

[3]

(c) The following table shows the instructions for the lorry load monitoring system in assembly language. There is one general purpose register, the Accumulator (ACC).

Table 7.1

Instruction			**Explanation**
Label	**Op code**	**Operand**	
	`LDM`	`#n`	Load the number n to ACC
	`LDD`	`<address>`	Load the contents of the location at the given address to ACC
	`STO`	`<address>`	Store the contents of ACC at the given address
	`AND`	`#n`	Bitwise AND operation of the contents of ACC with the operand
	`CMP`	`#n`	Compare the contents of ACC with number n
	`JPE`	`<address>`	Following a compare instruction, jump to <address> or <label> if the compare was True
	`JMP`	`<address>`	Jump to the given address or label
`<label>:`	`<op code>`	`<operand>`	Labels an instruction

Note:

denotes immediate addressing
B denotes a binary number, for example `B01001010`
& denotes a hexadecimal number, for example `&4A`

(i) Write **assembly language** instructions to set the contents of addresses `801` and `802` to zero, and set all four most significant bits of the contents of address `803` to one.
Use the instruction set from **Table 7.1**.

....

....

....

....

....

.... [3]

(ii) A program written in assembly language, continuously checks the flags. If a flag is set, the program jumps to the error-handling routine at the specified label. For example, if the load is too heavy, the program jumps to the error-handling routine with the label `TOOHEAVY`. The error-handling routine instructions have not been provided.

A programmer has written most of the instructions for the program in the following table. There are four missing operands.

Complete the assembly language program by writing the **four** missing operands.

Label	Op code	Operand
CHECKLOAD:	LDD	803
	AND	&F0
	STO	TEMP
	AND	&80
	CMP	&80
	JPE	TOOHEAVY
	LDD	TEMP
	AND	&40
	CMP	
	JPE	TOOHIGH
	LDD	TEMP
	AND	
	CMP	&20
	JPE	UNSTABLE
	LDD	
	AND	&10
	CMP	&10
	JPE	NOTSECURED
	JMP	
TEMP:		

[4]

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

5416314248

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **May/June 2022**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

DC (CE/SG) 303765/2

1 Data types can be defined using pseudocode.

The data type, `BuildingRecord`, is defined in pseudocode as:

```
TYPE BuildingRecord
   DECLARE BuildingID : INTEGER
   DECLARE BuildingGroup : STRING
   DECLARE OwnerName : STRING
   DECLARE BuildingAddress : STRING
   DECLARE DateLastSold : DATE
   DECLARE PriceLastSold : REAL
ENDTYPE
```

A variable, `BuildingRegister`, is declared in pseudocode as:

```
   DECLARE BuildingRegister : BuildingRecord
```

(a) Write **pseudocode** statements to assign:

- `1067` to the `BuildingID` of `BuildingRegister`
- `house` to the `BuildingGroup` of `BuildingRegister`

....

....

....

.... [2]

(b) The type definition for `BuildingRecord` is changed. The data type for `BuildingGroup` is changed to an enumerated type, `BuildingType`, with values of house, bungalow, apartment and farm.

(i) Write the type declaration for `BuildingType` in **pseudocode**.

....

....

....

.... [2]

(ii) Write the new declaration for `BuildingGroup` in **pseudocode**.

....

.... [1]

(iii) Write the new **pseudocode** statement to assign `house` to `BuildingGroup` of `BuildingRegister`.

....

.... [1]

(c) The program is to be rewritten using Object-Oriented Programming (OOP). The data type `BuildingRecord` is to be changed to a class, `BuildingClass`.

The properties for `BuildingClass` are `BuildingID`, `BuildingGroup`, `OwnerName`, `BuildingAddress`, `DateLastSold` and `PriceLastSold`.

All the properties are set to `PRIVATE`, for example:

```
PRIVATE PriceLastSold : REAL
```

(i) Write the declaration in **pseudocode** for `OwnerName` as `PRIVATE`.

....

.... [1]

(ii) Explain why the properties have been set to `PRIVATE`.

....

....

....

.... [2]

2 A declarative language is used to represent the following facts about a school.

```
01 teaches(alan, mathematics).
02 teaches(ioana, geography).
03 teaches(nina, history).
04 teaches(alan, statistics).
05
06 studies(ahmed, history).
07 studies(freya, history).
08 studies(kim, history).
09 studies(freya, geography).
10 studies(hua, mathematics).
11 studies(hua, statistics).
12 studies(hua, geography).
13
14 tutors(alan, kim).
15 tutors(alan, hua).
16 tutors(alan, freya).
17 tutors(nina, ahmed).
```

These clauses have the following meaning:

Clause	Meaning
01	Alan teaches mathematics.
06	Ahmed studies history.
14	Alan is Kim's tutor.

(a) More facts are to be included. Sam studies history and Nina is his tutor.

Write the additional clauses to record these facts.

18 ....

19 ....
[2]

(b) Using the variable `Student`, the goal:

```
studies(Student, history)
```

returns

```
Student = freya, ahmed, kim
```

Write the result returned by the goal:

```
studies(Student, geography)
```

`Student =` .... [1]

(c) Write the goal, using the variable `X`, to find all the students who have a tutor that teaches them. For example, Hua has Alan for a tutor and is also taught mathematics by Alan.

....

....

....

.... [4]

3 The TCP/IP protocol suite has four layers. The application layer provides user services.

(a) Identify **two** protocols used by this layer. Describe the use of each protocol.

Protocol 1 ....

Description ....

....

....

Protocol 2 ....

Description ....

....

....
[4]

(b) Identify **two other** layers of the TCP/IP protocol suite. Describe the function of each layer.

Layer 1 ....

Description ....

....

....

Layer 2 ....

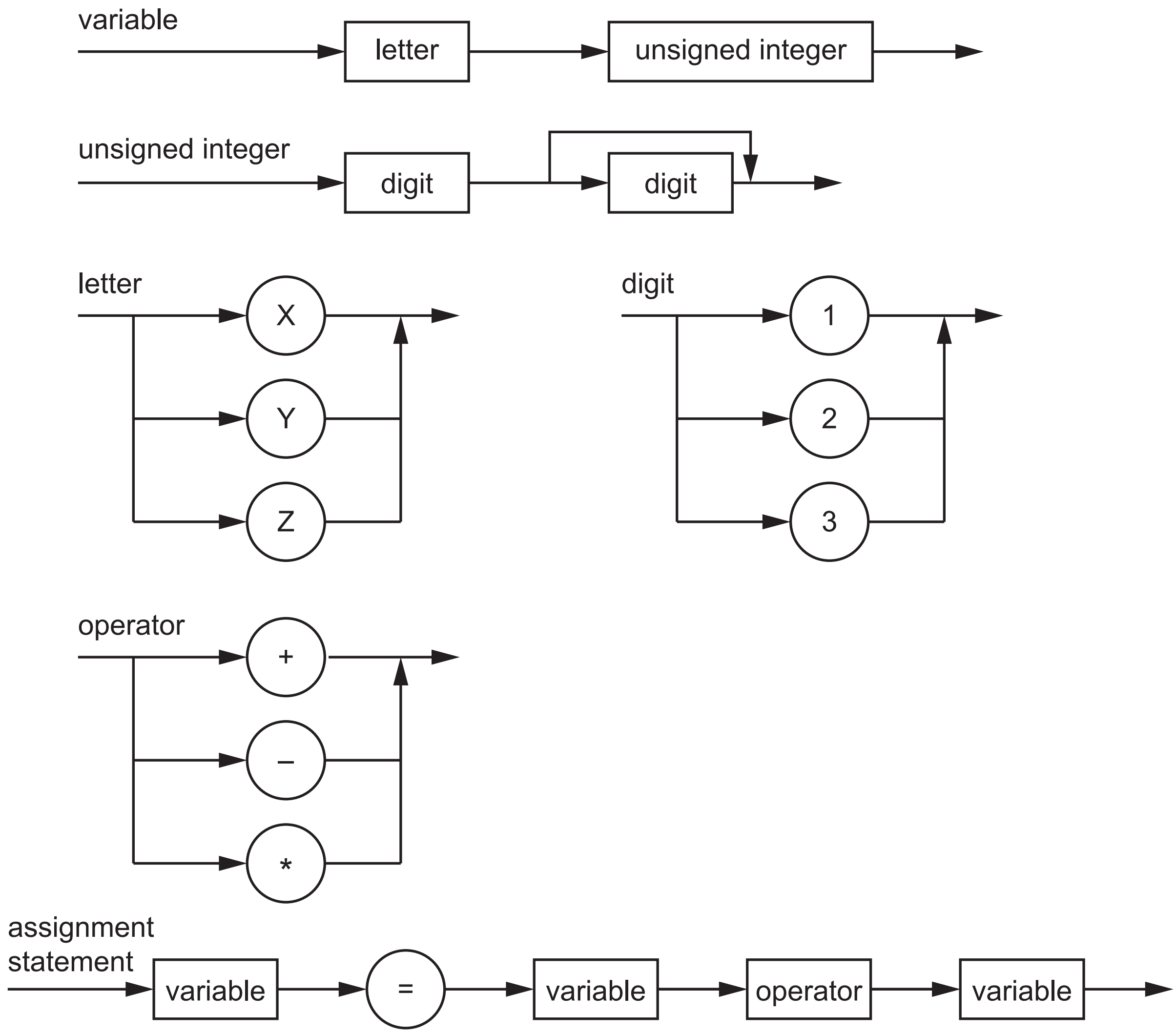
Description ....

....

....
[4]

4 The following syntax diagrams show the syntax of:

- a variable
- an unsigned integer
- a letter
- a digit
- an operator
- an assignment statement.



(a) The following assignment statements are invalid. State the reason in each case.

`X1 = Y2 - 12`

Reason ....

....

`Z = Y12 + Z1`

Reason ....

.... [2]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.

`<letter>` has been completed for you.

`<variable> ::=` ....

....

`<unsigned_integer> ::=` ....

....

`<letter> ::= X | Y | Z`

`<digit> ::=` ....

....

`<operator> ::=` ....

....

`<assignment_statement> ::=` ....

....
[5]

(c) The syntax of an assignment statement is changed to allow each of the variables on the right-hand side of the '=' symbol to be either a variable or an unsigned integer.

(i) Draw a syntax diagram for the new syntax of the **assignment statement**.

[3]

(ii) Write the Backus-Naur Form (BNF) for your syntax diagram.

....

....

....

.... [3]

5 There are four basic categories of computer architecture. Single Instruction Single Data (SISD) is one architecture.

Identify the **three other** categories of computer architecture.

Describe each category that you identify.

Architecture 1 ....

Description ....

....

....

....

Architecture 2 ....

Description ....

....

....

....

Architecture 3 ....

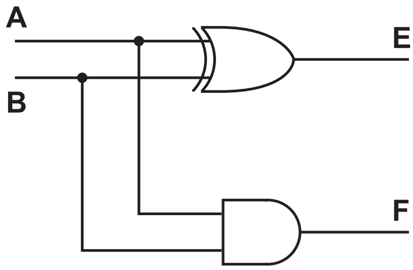
Description ....

....

....

....
[6]

6 A logic circuit has two inputs **A** and **B**, and two outputs **E** and **F**.



(a) Complete the truth table for this logic circuit.

INPUT		**OUTPUT**	
A	**B**	**E**	**F**
0	0		
0	1		
1	0		
1	1		

[2]

(b) **(i)** State the name of this logic circuit.

.... [1]

(ii) State the purpose of each output **E** and **F**.

Purpose of **E** ....

Purpose of **F** ....
[2]

7 A digital signature is used to validate the authenticity of an electronic message.

In order to produce a digital signature, a digital certificate is required.

(a) State how a digital certificate is obtained.

....

....

....

....

....

.... [3]

(b) (i) Explain how a digital signature is produced before the message is sent.

....

....

....

....

....

....

....

.... [3]

(ii) Explain how the digital signature can be checked on receipt to ensure that the message has not been altered during transmission.

....

....

....

....

....

....

....

.... [4]

8 A binary search or a linear search can be used to look for a specific value in an array.

(a) Complete this pseudocode algorithm for a linear search.

```
DECLARE MyList : ARRAY[0:9] OF INTEGER
DECLARE MaxIndex : INTEGER
DECLARE Index : INTEGER
DECLARE Found : BOOLEAN
DECLARE ValueToFind : ..........................................

INPUT ValueToFind
Found ← FALSE
Index ← 0
MaxIndex ← ..........................................

REPEAT
   IF MyList[Index] = ValueToFind THEN
      Found ← TRUE
   ENDIF
   Index ← ..........................................
UNTIL Found OR Index > MaxIndex

IF Found THEN
   OUTPUT "Value found at position ", Index
ELSE
   OUTPUT ..........................................
ENDIF
```

[4]

(b) **(i)** State the necessary condition for a binary search.

....

.... [1]

(ii) Describe how to perform a binary search.

....

....

....

....

....

....

....

.... [4]

(iii) Explain how the performance of a binary search varies according to the number of values in the array.

....

....

.... [1]

(c) Compare the performance of the algorithms for a binary search and a linear search using Big O notation for order of time complexity.

....

....

....

....

....

.... [3]

9 State the reasons for including exception handling routines when writing a program.

Include an example of an exception in your answer.

....

....

....

....

....

....

....

.... [4]

Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME	

CENTRE NUMBER		CANDIDATE NUMBER	

9961215436

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2019**

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 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.
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.

The maximum number of marks is 75.

This document consists of **19** printed pages and **1** blank page.

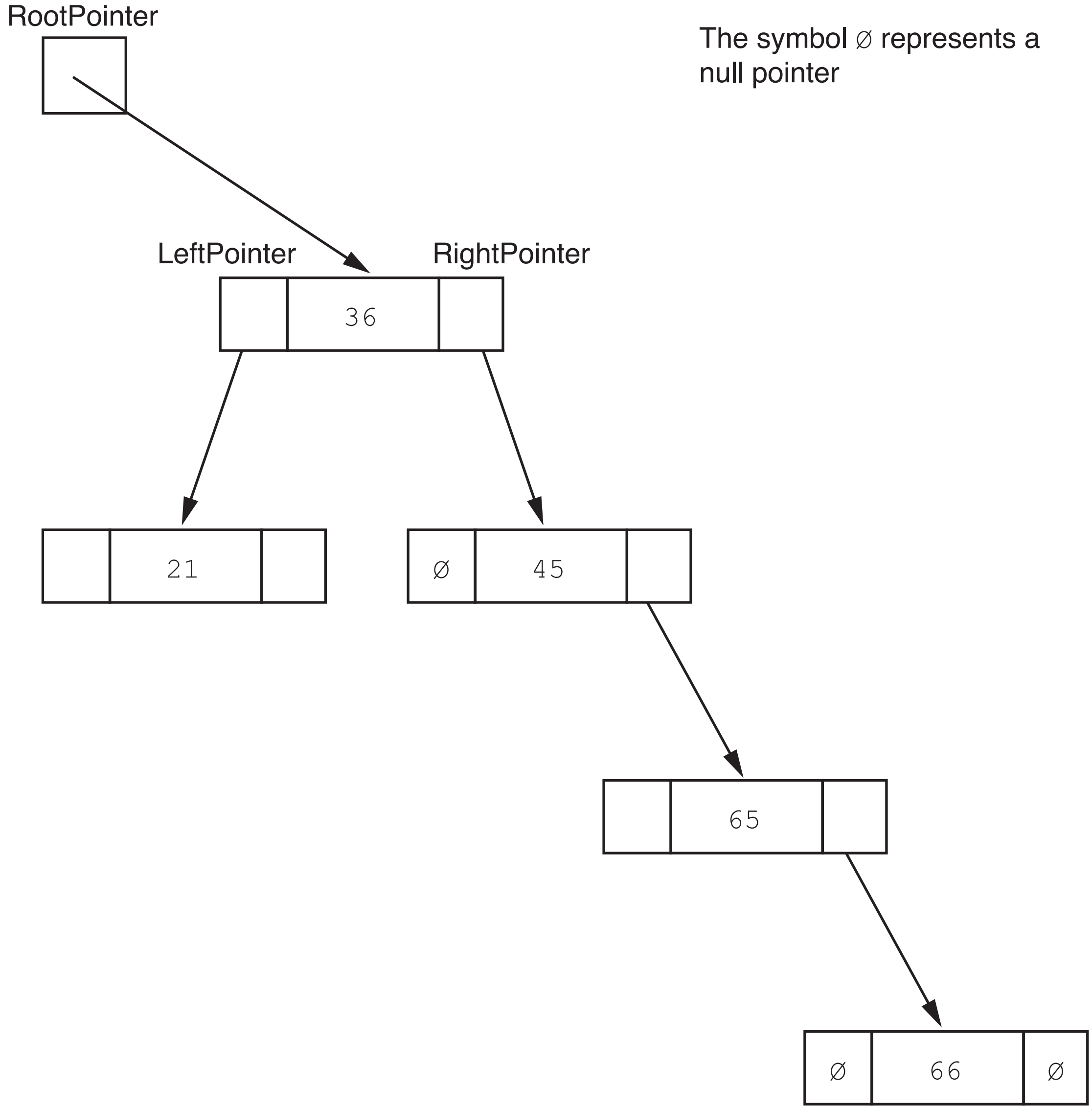
1 A company wants an online marking system for an examination.

(a) The following is a selection of data showing final marks.

`36, 45, 21, 65, 66, 13, 54, 53, 34`

A linked list of nodes will be used to store the data. Each node consists of the data, a left pointer and a right pointer. The linked list will be organised as a binary tree.

(i) Complete the binary tree to show how the data above will be organised.



[5]

(ii) The following diagram shows a 2D array that stores the nodes of the binary tree's linked list.

Add the correct pointer values to complete the diagram, using your answer from **part (a)(i)**.

RootPointer

0

FreePointer

Index	**LeftPointer**	**Data**	**RightPointer**
0		36	
1		45	
2		21	
3		65	
4		66	
5		13	
6		54	
7		53	
8		34	
9			

[6]

(b) The company wants to implement a program for the marking system. It will do this with object-oriented programming (OOP).

Many candidates take the examination. Each examination paper is given a `PaperID` that is made up of the centre (school) number followed by the candidate number.

Each examination paper is awarded a grade.

The following diagram shows the design for the `ExaminationPaper` class. This includes the attributes and methods.

ExaminationPaper
<pre>FinalMark : INTEGER // maximum 2 digits, initialised to 0 Grade   : STRING // "Pass", "Merit", "Distinction"   // or "Fail", initialised to "Fail" PaperID   : STRING // centre number followed by the   // candidate number, for example   // "ZZ00991001"</pre>
<pre>Create()   // creates and initialises a new instance   // of the ExaminationPaper class using   // language-appropriate constructor SetFinalMark()   // checks that the mark parameter has a   // valid value, if so, assigns it to   // FinalMark SetGrade()   // sets Grade based on FinalMark GetFinalMark()   // returns FinalMark GetGrade()   // returns Grade GetPaperID()   // returns PaperID</pre>

(i) The constructor receives the centre number and candidate number as parameter values to create `PaperID`. Other properties are initialised as instructed in the class diagram.

Write **program code** for the `Create()` constructor method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

.... [5]

(ii) Get and set methods are used to support the security and integrity of data in object-oriented programming.

Explain how get and set methods are used to support security and integrity.

....

....

....

....

....

.... [3]

(iii) Write **program code** for the following three get methods.

Programming language ....

`GetFinalMark()`

Program code

....

....

....

....

`GetGrade()`

Program code

....

....

....

....

`GetPaperID()`

Program code

....

....

....

....

[4]

(iv) The method `SetFinalMark()` checks that its `INTEGER` parameter `Mark` is valid. It is then set as the final mark if it is valid. A valid mark is greater than or equal to 0 and less than or equal to 90.

If the mark is valid, the method sets the final mark and returns `TRUE`.
If the mark is not valid, the method does not set the final mark and returns `FALSE`.

Write **program code** for `SetFinalMark(Mark : INTEGER)`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

.... [5]

(v) Write **program code** for the method:

`SetGrade(DistMark, MeritMark, PassMark : INTEGER)`

Use the properties in the original class definition.

Grades are awarded as follows:

Grade	Criteria
Distinction	`>= DistMark`
Merit	`>= MeritMark`
Pass	`>= PassMark`
Fail	`< PassMark`

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [4]

(vi) Emily is a candidate who has taken the examination paper. The grades are awarded as follows:

Grade	Criteria
Distinction	`>= 80`
Merit	`>= 70`
Pass	`>= 55`

The procedure `Main()` performs the following tasks.

- allows the centre number, candidate number and mark to be input, with suitable prompts
- assigns an instance of `ExaminationPaper` to the variable `ThisPaper`
- sets the mark for the object
- sets the grade for the object
- outputs the grade for the object

Write **program code** for the `Main()` procedure.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

(c) The examination paper will be taken by many candidates in centres around the world.

The program stores the objects of the `ExaminationPaper` class in a file. The company has decided to use a hash table, rather than a linked list to store the objects.

Explain why a hash table is more suitable than a linked list to store the objects.

....

....

....

....

....

....

....

.... [4]

Question 2 begins on the next page.

2 A stack is an Abstract Data Type (ADT).

(a) Tick (✓) **one** box to show the statement that describes a stack data structure.

Statement	**Tick (✓)**
Last in first out	
First in first out	
Last in last out	

[1]

(b) A stack contains the values `20`, `35`, `43`, `55`.

`55`	← Top of stack
`43`	
`35`	
`20`	

(i) Show the contents of the stack in **part (b)** after the following operations.

```
POP()

POP()

PUSH(10)
```


[1]

(ii) Show the contents of the stack from **part (b)(i)** after these further operations:

```
POP()

PUSH(50)

PUSH(55)

POP()

PUSH(65)
```


[1]

(iii) The stack is implemented as a 1D array, with eight elements, and given the identifier `ArrayStack`.

The global variable `Top` contains the index of the last element in the stack, or –1 if the stack is empty.

The function `Push()`:

- takes as a parameter an `INTEGER` value to place on the stack
- adds the value to the top of the stack and returns `TRUE` to show that the operation was successful
- returns `FALSE` if the stack is full.

Write an algorithm in **pseudocode** for the function `Push()`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [7]

3 **(a)** Identify **and** describe **two** features of an editor that can help a programmer to write program code.

Feature 1 ....

Description ....

....

....

Feature 2 ....

Description ....

....

....
[4]

(b) A programmer can use three types of test data when testing a program.

Identify the **three** different types of test data.

1 ....

2 ....

3 ....
[3]

4 (a) A program has sorted some data in the array, `List`, in ascending order.

The following binary search algorithm is used to search for a value in the array.

```
ValueFound ← FALSE
UpperBound ← LengthOfList - 1
LowerBound ← 0
NotInList ← FALSE

WHILE ValueFound = FALSE AND NotInList = FALSE
   MidPoint ← ROUND((LowerBound + UpperBound) / 2)

   IF List[LowerBound] = SearchValue
      THEN
         ValueFound ← TRUE
      ELSE
         IF List[MidPoint] < SearchValue
            THEN
               UpperBound ← MidPoint + 1
            ELSE
               UpperBound ← MidPoint – 1
         ENDIF
         IF LowerBound > MidPoint
            THEN
               NotInList ← TRUE
         ENDIF
   ENDIF
ENDWHILE

IF ValueFound = FALSE
   THEN
      OUTPUT "The value is in the list"
   ELSE
      OUTPUT "The value is not found in the list"
ENDIF
```

Note:

The pseudocode function

```
ROUND(Reall : REAL) RETURNS INTEGER
```

rounds a number to the nearest integer value.

For example: `ROUND(4.5)` returns `5` and `ROUND(4.4)` returns `4`

(i) There are four errors in the algorithm.

Write the line of code where an error is present **and** write the correction in **pseudocode**.

Error 1 ....

Correction ....

Error 2 ....

Correction ....

Error 3 ....

Correction ....

Error 4 ....

Correction ....
[4]

(ii) A binary search is one algorithm that can be used to search an array.

Identify another searching algorithm.

.... [1]

(b) The following is an example of a sorting algorithm. It sorts the data in the array `ArrayData`.

```
TempValue ← ""
REPEAT
   Sorted ← TRUE
   FOR Count ← 0 TO 4
      IF ArrayData[Count] > ArrayData[Count + 1]
         THEN
            TempValue ← ArrayData[Count + 1]
            ArrayData[Count + 1] ← ArrayData[Count]
            ArrayData[Count] ← TempValue
            Sorted ← FALSE
      ENDIF
   ENDFOR
UNTIL Sorted = TRUE
```

(i) Complete the trace table for the algorithm given in **part (b)**, for the `ArrayData` values given in the table.

Count	**TempValue**	**Sorted**	**ArrayData**					
			0	**1**	**2**	**3**	**4**	**5**
			5	20	12	25	32	29

[4]

(ii) Rewrite lines 4 to 12 of the algorithm in **part (b)** using a `WHILE` loop instead of a `FOR` loop.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [3]

(iii) Identify the algorithm shown in **part (b)**.

.... [1]

(iv) Identify another sorting algorithm.

.... [1]

BLANK PAGE

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2019**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **13** printed pages and **3** blank pages.

1 (a) The following incomplete table shows descriptions relating to the security of data transmission.

Complete the table with the appropriate terms.

	Description	Term
A	The original data to be transmitted as a message	....
B	An electronic document from a trusted authority that ensures authentication	....
C	An encryption method produced by a trusted authority that can be used by anyone	....

[3]

(b) (i) Explain the purpose of a digital signature.

....

....

....

.... [2]

(ii) Describe how a digital signature is produced for transmission with the message.

....

....

....

....

....

....

.... [3]

2 **(a)** A Boolean expression produces the following truth table.

INPUT			**OUTPUT**
A	**B**	**C**	**X**
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

(i) Write the Boolean expression for the truth table by applying the sum-of-products.

X = ...

.... [3]

(ii) Complete the Karnaugh Map (K-map) for the truth table in **part (a)**.

AB

C		**00**	**01**	**11**	**10**
	0				
	1				

[1]

The K-map can be used to simplify the function in **part (a)(i)**.

(iii) Draw loop(s) around appropriate groups in the table in **part (a)(ii)**, to produce an optimal sum-of-products. [2]

(iv) Write, using your answer to **part (a)(iii)**, a simplified Boolean expression for your Karnaugh map.

X = ... [2]

(b) Simplify the following expression using De Morgan's laws. Show your working.

$$\overline{(\overline{W} + X) \bullet (Y + \overline{Z})}$$

....

....

....

....

....

.... [3]

3 A computing department in a school has a Local Area Network (LAN) with a bus topology.

(a) A description of sending a message on a bus network is given.

Complete the following description by inserting an appropriate term in each space.

Computer 1 and Computer 2 are on the same bus network. Computer 1 sends a message to Computer 2. Before the message is sent, it is split into ....

Computer 1 needs to check that the .... is free, before sending the message, otherwise a .... will occur that will be managed by the .... protocol.

[4]

(b) The computing department's LAN needs to connect to the Internet.

Explain how each device is used in the operation of the bus network.

Router ....

....

....

....

Network Interface Card (NIC) ....

....

....

....

[4]

(c) The computing department's network is being adapted to allow students to connect wireless devices.

(i) Identify **two** types of hardware components the computing department will need to allow wireless connection.

1 ....

2 ....
[2]

(ii) Describe how the wireless connection sends and receives data.

....

....

....

....

....

....

....

.... [4]

4 Physical memory is managed using virtual memory and paging.

(a) Describe what is meant by **virtual memory**.

....

....

....

.... [2]

(b) (i) Explain how paging is used to manage virtual memory.

....

....

....

....

....

....

....

.... [4]

(ii) Give a suitable page replacement algorithm for this process.

.... [1]

(iii) One drawback of using virtual memory is disk thrashing.

Describe what is meant by the term **disk thrashing**.

....

....

....

.... [2]

5 A weather station uses monitoring and control systems.

(a) Describe the difference between a monitoring system and a control system.

....

....

....

.... [2]

(b) (i) The weather station records how the outside temperature changes over a period of time. The system will read the temperature once every hour, over a period of 100 days.

The temperature readings are automatically stored in a file. No other data are stored.

Explain why the weather station has decided to use serial organisation for the file.

....

....

....

.... [2]

(ii) Serial files can be accessed using sequential access.

Explain how sequential access could be used for the temperature readings file.

....

....

....

.... [2]

(iii) Name **and** describe a method of file organisation other than serial or sequential.

Method ....

Description ....

....

....

....

....

....

[4]

6 **(a)** State what is meant by a **user-defined data type**.

....

.... [2]

(b) A pseudocode declaration for a user-defined data type for the months of the year is as follows:

```
TYPE
  DECLARE Months: (January, February, March, April, May, June, July,
                   August, September, October, November, December)
ENDTYPE
```

(i) Identify this type of user-defined data type.

....

.... [1]

(ii) Write a **pseudocode** statement to declare a variable `CurrentMonth` of data type `Months`.

....

.... [1]

(iii) Write a **pseudocode** statement to assign the value `August` to the variable `CurrentMonth`.

....

.... [1]

7 The following are the first few lines of a source code program written in a high-level language. The source code program is to be translated by the language compiler.

```
// program written on 15 June 2019

DECLARE IsFound : Boolean;
DECLARE NoOfChildren : Integer;
DECLARE Count : Integer;
Constant TaxRate = 15;

// start of main program
For Count = 1 to 50
…
…
…
```

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

(i) Identify **two** types of data in the keyword table.

Type 1 ....

Type 2 ....
[2]

(ii) Identify **two** types of data in the symbol table.

Type 1 ....

Type 2 ....
[2]

(iii) Explain how the contents of the keyword and symbol tables are used to translate the source code program.

....

....

....

.... [2]

(iv) State **one** additional task completed at the lexical analysis stage that does not involve the use of a keyword or a symbol table.

....

.... [1]

(b) The final stage of compilation can be code optimisation.

Explain why code is optimised.

....

....

....

.... [2]

8 **(a)** The following 16-bit binary pattern represents a floating-point number stored in two's complement form. The twelve most significant bits are used for the mantissa and the four least significant bits are used for the exponent.

Most significant bit ↓ ... **Least significant bit** ↓

0	1	1	1	0	0	0	0	0	0	0	0	1	1	0	1

(i) Identify the binary value of the exponent.

.... [1]

(ii) Identify the binary value of the mantissa.

.... [1]

(iii) State whether the number stored is positive or negative. Justify your choice.

Positive or negative ....

Justification ....

....

.... [2]

(iv) Convert the binary floating-point number in **part (a)** into denary. Show your working.

Working ....

....

....

....

....

Denary value .... [3]

(b) The number of bits used for the exponent is increased to eight, and the number of bits used for the mantissa is decreased to eight.

State the effects of this change.

....

....

....

.... [2]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

CANDIDATE NAME	
CENTRE NUMBER	
CANDIDATE NUMBER	

7920751865

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **October/November 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (RW/SW) 206388/2

1 **(a)** Numbers are stored in a computer using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and exponent.

(i) Write the normalised floating-point representation of the following unsigned binary number using this system.

1011100.011001

Working ....

....

....

....

....

Mantissa **Exponent**

[2]

(ii) State the consequence of storing the binary number in **part (a)(i)** as a floating-point number in this system. Justify your answer.

Consequence ....

....

Justification ....

....

....

[2]

(b) Explain the reason why binary numbers are stored in normalised form.

....

....

....

....

....

....

.... [3]

2 Draw **one** line from each programming paradigm to its **most appropriate** description.

Programming paradigm	Description
Declarative	Programs using the instruction set of a processor
Imperative	Programs based on events such as user actions or sensor outputs
Low-level	Programs using the concepts of class, inheritance, encapsulation and polymorphism
Object-oriented	Programs with an explicit sequence of commands that update the program state, with or without procedure calls
	Programs that specify the desired result rather than how to get to it

[4]

3 Enumerated and pointer are two non-composite data types.

(a) Write **pseudocode** to create an enumerated type called `Parts` to include these parts sold in a computer shop:

`Monitor, CPU, SSD, HDD, LaserPrinter, Keyboard, Mouse`

....

....

.... [2]

(b) Write **pseudocode** to create a pointer type called `SelectParts` that will reference the memory location in which the current part name is stored.

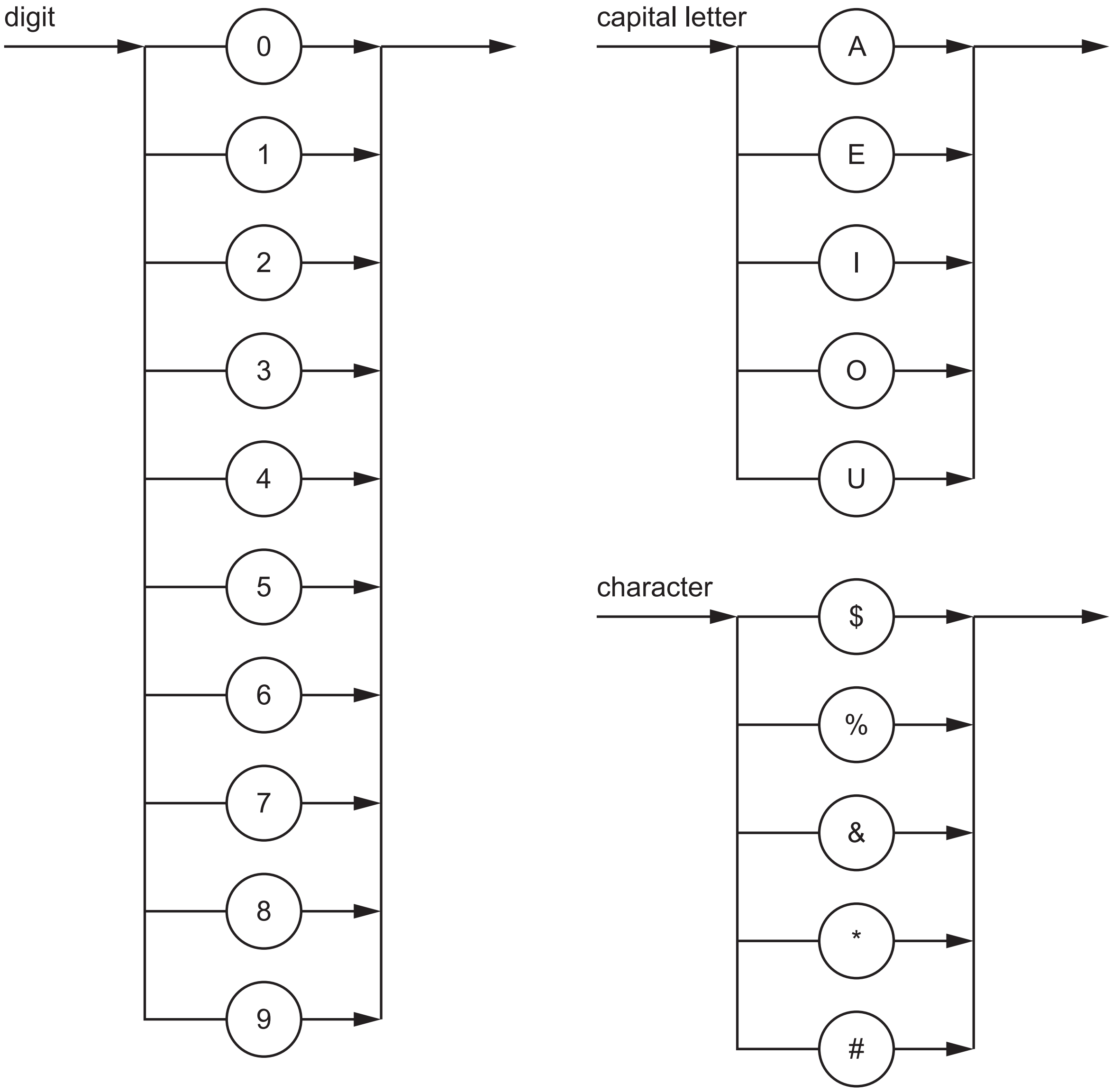
....

....

.... [2]

4 The following syntax diagrams for a particular programming language show the syntax of:

- a digit
- a capital letter
- a character.



(a) Write the Backus-Naur Form (BNF) notation of the syntax diagram for character.

....

....

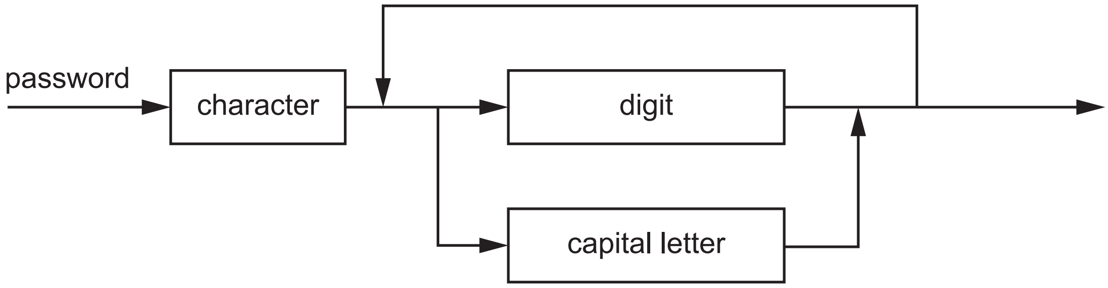
.... [2]

(b) A password must begin with a character and be followed by one or more digits or capital letters.

(i) State an example of a valid password.

.... [1]

(ii) A valid password is represented by the syntax diagram:



Write the BNF notation of the syntax diagram for password.

.... [4]

5 **(a)** Compare sequential and serial methods of file organisation.

....

....

....

....

....

....

....

....

.... [4]

(b) State the most suitable method of file access when a record is referenced by a unique address on a disk-type storage medium.

.... [1]

(c) State the most suitable method of file access when a bank stores its data records in ascending order of account number.

.... [1]

6 **(a)** Explain how packet switching is used to transfer messages across the internet.

....

....

....

....

....

....

....

....

....

.... [5]

(b) Outline the function of a router in packet switching.

....

....

....

....

....

.... [3]

7 **(a)** Write the Boolean expression that corresponds to the given truth table as a sum-of-products.

INPUT				**OUTPUT**
A	**B**	**C**	**D**	**Z**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Z = ...

.... [3]

(b) **(i)** Complete the Karnaugh map (K-map) for the given truth table.

CD \ AB	00	01	11	10
00				
01				
11				
10				

[2]

(ii) Draw loop(s) around appropriate group(s) of 1s in the K-map to produce an optimal sum-of-products. [2]

(iii) Write the Boolean expression from your answer to **part b(ii)** as a simplified sum-of-products.

Z = ...

.... [2]

(iv) Write the simplified Boolean expression for your answer to **part b(iii)**.

Z = ...

.... [1]

8 **(a)** Describe the purpose of the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.

....

....

....

....

.... [2]

(b) Explain how SSL/TLS protocols are used when a client-server communication is initiated.

....

....

....

....

....

....

....

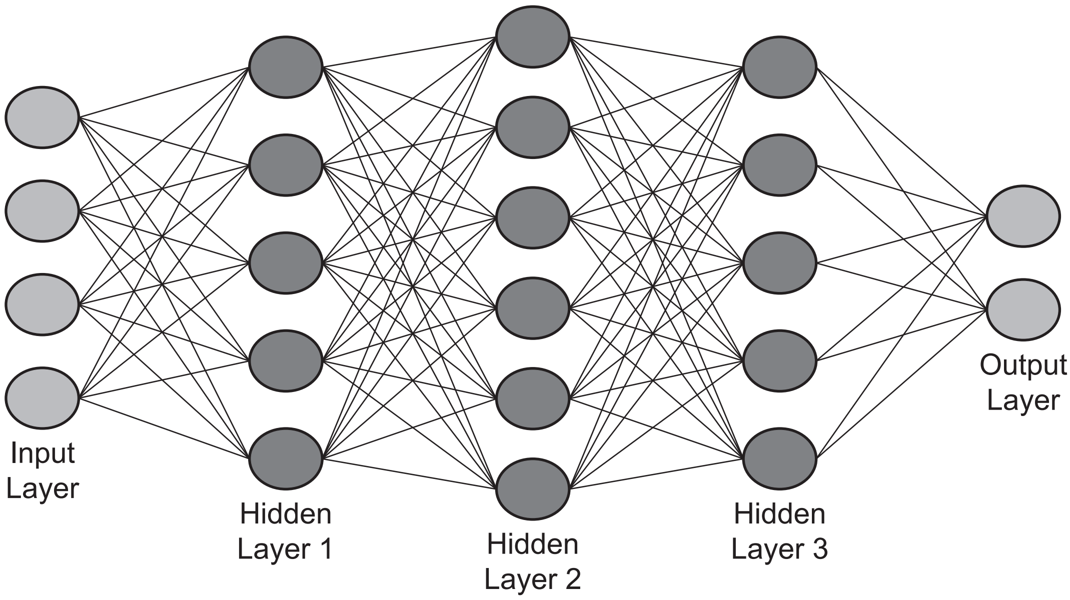
....

....

....

.... [4]

9 **(a)** The diagram shown represents an artificial neural network.



(i) State the reason for having multiple hidden layers in an artificial neural network.

....

.... [1]

(ii) Explain how artificial neural networks enable machine learning.

....

....

....

....

....

....

....

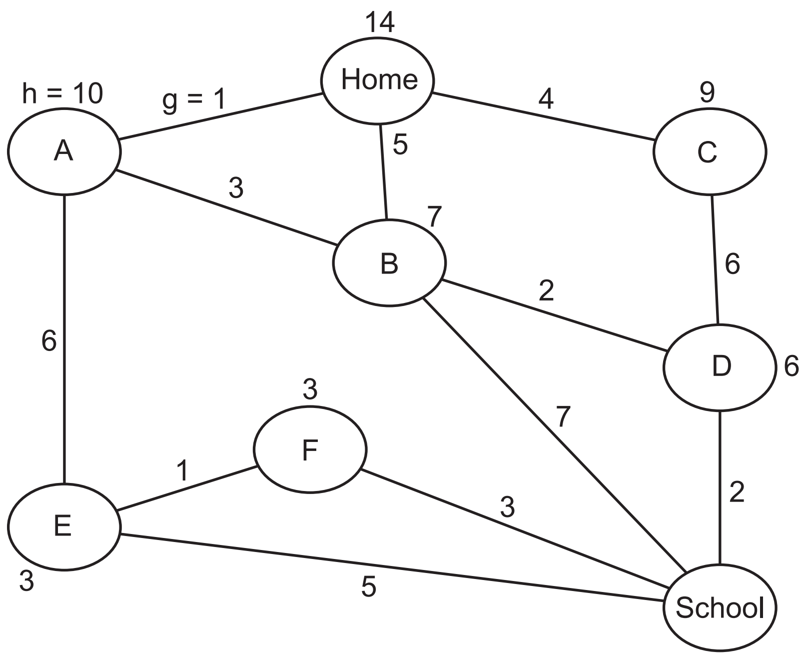
....

....

.... [4]

(b) Find the shortest path between the Home and School nodes using the A* algorithm. Show your working in the table provided.

The first two rows in the table have been completed.



Node	Cost from Home node (g)	Heuristic (h)	Total (f = g + h)
Home	0	14	14
A	1	10	11

Final path	

[5]

10 **(a)** State **three** essential features of **recursion**.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) Explain the reasons why a stack is a suitable Abstract Data Type (ADT) to implement recursion.

....

....

....

....

....

.... [3]

(c) Identify **two** ADTs other than a stack.

1 ....

2 ....
[2]

(d) The function `StackFull()` checks whether a stack is full.

The function uses the variable `TopOfStack` to represent the pointer to the most recent position used on the stack, and the variable `Max` to represent the maximum size of the stack. Assume `TopOfStack` and `Max` are global variables.

```
FUNCTION StackFull() RETURNS BOOLEAN
    IF TopOfStack = Max THEN
       RETURN TRUE
    ELSE
       RETURN FALSE
    ENDIF
ENDFUNCTION
```

An algorithm `AddInteger` is required to add a new integer data element to a stack.

The stack is implemented as an array `ArrayStack`.

The function `AddInteger()` calls `StackFull()` and returns an appropriate message.

Complete the pseudocode for the function `AddInteger()`.

```
FUNCTION AddInteger(NewInteger : INTEGER) RETURNS STRING
```

....

....

....

....

....

....

....

....

....

....

....

....

```
ENDFUNCTION
```

[5]

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

8982694890

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **May/June 2023**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (RW/FC) 313065/3

1 Numbers are stored in a computer using floating point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and exponent.

(a) Write the normalised floating-point representation of the following binary number using this system:

0101010.111

Show your working.

Working ....

....

....

....

....

Mantissa

Exponent

[2]

(b) Describe the reason why the normalised form of the following binary number cannot be represented accurately using this system.

0101011.111001

....

....

....

....

....

.... [3]

2 **(a)** Describe how records are organised and accessed in a sequential file.

....

....

....

....

....

.... [3]

(b) A hashing algorithm is used to calculate storage locations for records in a random access file. The algorithm calculates hash values using the function modulus 5.

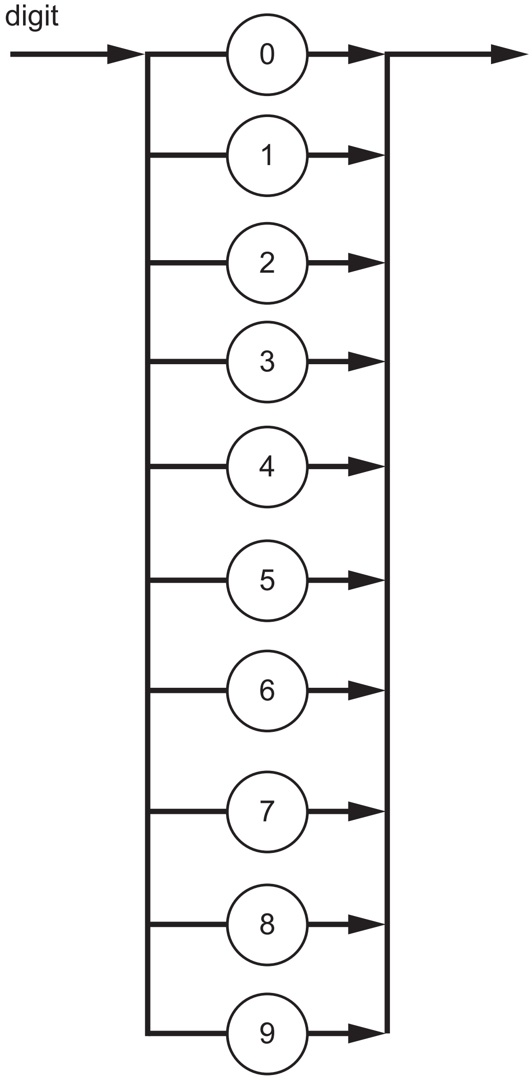
The function modulus gives the remainder after integer division.
For example, 3003 modulus 5 = 3, so the record key 3003 gives a hash value of 3.

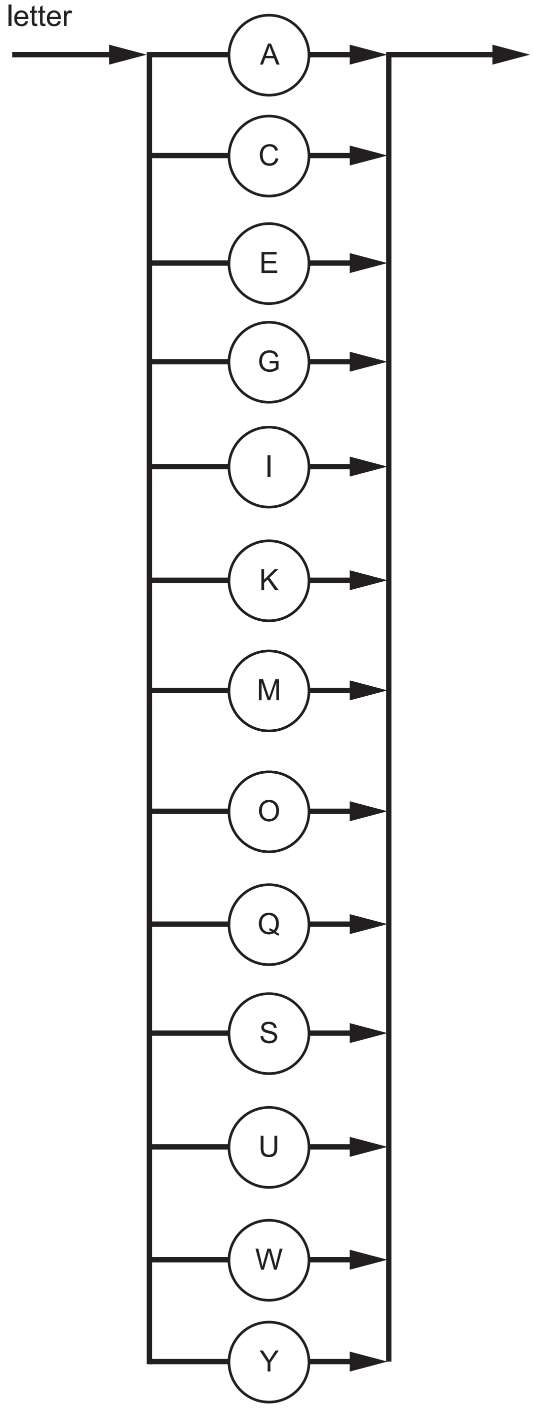
Complete the table to show the remaining hash values.

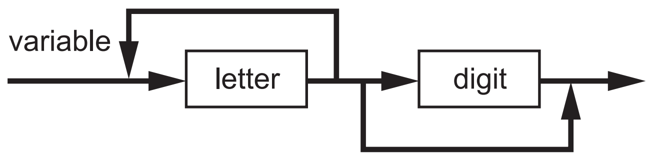
Record key	Hash value
3003	3
1029	
7630	

[2]

3 Several syntax diagrams are shown.







(a) State whether each variable is valid or invalid **and** give a reason for your choice in each case.

`9SW` ....

Reason ....

....

`UWY` ....

Reason ....

.... [2]

(b) `<word>` contains one or more letters.

Complete the Backus-Naur Form (BNF) for `<word>` **and** use this to complete the BNF for `<variable>`.

`<word> ::=` ....

....

`<variable> ::=` ....

.... [3]

(c) Vehicle registrations must begin with two letters and be followed by one, two or three digits.

Valid letters and digits are shown in the syntax diagrams on page 4.

(i) State an example of a valid vehicle registration.

....

.... [1]

(ii) Draw a syntax diagram for a vehicle registration.

[3]

4 Draw **one** line from each Object-Oriented Programming (OOP) term to its **most appropriate** description.

OOP term	Description
Encapsulation	methods used to return the value of a property
Getters	the process of putting data and methods together as a single unit
Polymorphism	methods used to update the value of a property
Setters	allows methods to be redefined for derived classes
	enables the defining of a new class that inherits from a parent class

[4]

5 **(a)** Encryption is used to scramble data to make it meaningless if intercepted.

Describe the purpose of quantum cryptography.

....

....

....

.... [2]

(b) Explain the differences between symmetric and asymmetric cryptography when encrypting and decrypting data.

....

....

....

....

....

.... [3]

6 **(a)** Write **pseudocode** statements to declare the composite data type, `TAppointments`, to hold data about patients for a dental clinic. It will include for each patient:

- name (first name and last name)
- date of birth
- telephone number
- date of last appointment
- date of next appointment
- all treatments are complete (yes or no).

....

....

....

....

....

....

....

.... [4]

(b) This pseudocode algorithm reads dental records stored in a random file using the user-defined data type `TAppointments` and prints the contents of the file, one record at a time.

Complete this file handling pseudocode:

```
DECLARE DentalRecord : ARRAY[1:250] OF TAppointments
DECLARE DentalFile : STRING
DECLARE Count : INTEGER
DentalFile ← "DentalFile.dat"
OUTPUT "The file ", DentalFile, " contains these records:"

OPENFILE ..........................................................................................

.................................................................... ← 1

REPEAT
   SEEK DentalFile, Count

   ........................................................................................
   OUTPUT DentalRecord[Count]
   Count ← Count + 1

.....................................................(DentalFile)

.......................................................................
```

[5]

7 **(a)** State **two** examples of where it would be appropriate to use packet switching.

....

....

....

.... [2]

(b) Give **four** differences between circuit switching and packet switching.

1 ....

....

2 ....

....

3 ....

....

4 ....

....
[4]

8 (a) Describe the use of pipelining in Reduced Instruction Set Computers (RISC).

....

....

....

.... [2]

(b) The processing of instructions is divided into five stages:

- instruction fetch (IF)
- instruction decode (ID)
- operand fetch (OF)
- instruction execute (IE)
- write back result (WB)

Each stage is carried out using a different register when pipelining is used.

Complete the table to show how a program consisting of **six** instructions would be completed using pipelining.

Clock cycles

Processor stages	1	2	3	4	5	6	7	8	9	10	11	12
IF												
ID												
OF												
IE												
WB												

[4]

9 This truth table represents a logic circuit.

INPUT				**OUTPUT**
A	**B**	**C**	**D**	**Z**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

(a) Write the Boolean logic expression that corresponds to the given truth table as the sum-of-products.

Z = ...

.... [3]

(b) Complete the Karnaugh map (K-map) for the given truth table.

CD \ AB	00	01	11	10
00				
01				
11				
10				

[2]

(c) Draw loop(s) around appropriate group(s) in the K-map to produce an optimal sum-of-products. [2]

(d) Write the Boolean logic expression from your answer to **part (c)** as a simplified sum-of-products.

Z = ...

.... [2]

(e) Use Boolean algebra to give your answer to **part (d)** in its simplest form.

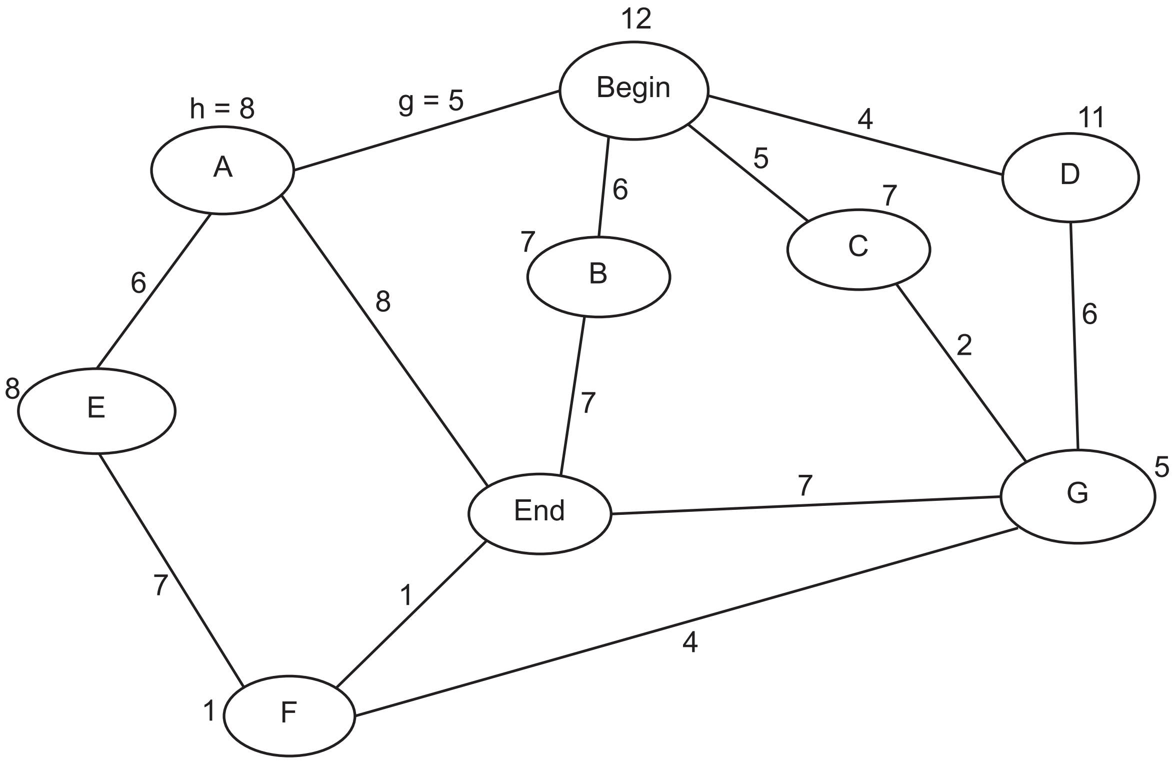
Z = ... [1]

10 (a) State **one** category of machine learning.

.... [1]

(b) Calculate the path that takes the shortest time to travel from the Begin node to the End node, using the A* algorithm.
Show your working in the table provided.

The first two rows have already been completed.



Start node	Destination node	Cost from start node (g)	Heuristic (h)	Total (f = g + h)
Begin	Begin	0	12	12
Begin	A	5	8	13

Final path	

[5]

11 (a) The pseudocode shown represents a queue Abstract Data Type (ADT) with procedures for initialisation and to add new items. It is incomplete.

```
CONSTANT MaxLength = 50
DECLARE FrontPointer : INTEGER
DECLARE RearPointer : INTEGER
DECLARE Length : INTEGER
DECLARE Queue : ARRAY[0 : MaxLength - 1] OF STRING

// initialisation of queue
PROCEDURE Initialise
   FrontPointer ← -1

   ........................................................................................................

   ............................................................ ← 0
ENDPROCEDURE

// adding a new item to the queue
PROCEDURE Enqueue(NewItem : STRING)

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

      RearPointer ← ..............................................................................
      IF RearPointer > MaxLength - 1 THEN
         RearPointer ← 0
      ENDIF

      ........................................................................................................
      Length ← Length + 1
   ENDIF
ENDPROCEDURE
```

(i) Study the pseudocode and insert the identifiers to complete this table.

Identifier	Data type	Description
	STRING	An array to store the contents of the queue.
	INTEGER	Points to the last item of the queue.
	INTEGER	Indicates the number of items in the queue.
	INTEGER	Points to the first item of the queue.

[2]

(ii) Complete the given pseudocode. [5]

(b) Explain the reasons why a queue ADT works better than a stack ADT in organising print jobs.

....

....

....

....

....

.... [3]

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME	

CENTRE NUMBER		CANDIDATE NUMBER	

1059195703

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (PQ/CGW) 205105/3

1 In a computer system, two real numbers, **A** and **B**, are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

Number A

	Mantissa												**Exponent**			
	0	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1

Number B

	Mantissa												**Exponent**			
	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0

(a) **(i)** Convert the binary values of the **mantissa** and the **exponent** for each number to their separate denary values.

A mantissa ....

....

A exponent ....

....

B mantissa ....

....

B exponent ....

.... [4]

(ii) Calculate the denary value of each floating-point number using your values from **part (a)(i)**.

Number A ....

....

....

Number B ....

....

.... [2]

(b) State which number, **A** or **B**, is stored in normalised floating-point form. Justify your answer.

Number ....

Justification ....

....

....

.... [3]

2 The TCP/IP protocol suite can be viewed as a stack with **four** layers.

(a) Write the correct descriptions for the **two** layers **and** the correct layers for the **two** descriptions given in the following table.

Layer	Description
Application	
Transport	
	Handles transmission of data
	Handles how data is physically sent

[4]

(b) Identify **and** state the purpose of **two** communication protocols other than TCP/IP.

Protocol 1 ....

Purpose ....

....

....

Protocol 2 ....

Purpose ....

....

.... [4]

3 **(a)** Describe, with the aid of a diagram, a bus topology network.

Description ....

....

....

....

....

.... [3]

(b) Describe the way in which a bus network uses Ethernet technology for communication.

....

....

....

....

....

....

....

.... [4]

BLANK PAGE

4 **(a)** The truth table for a logic circuit with four inputs is shown.

INPUT				OUTPUT
P	**Q**	**R**	**S**	**X**
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(i) Write the Boolean expression for the truth table as a sum-of-products.

X = ... [2]

(ii) Complete the Karnaugh Map (K-map) for the truth table.

		PQ			
		00	**01**	**11**	**10**
RS	**00**				
	01				
	11				
	10				

[2]

(iii) The K-map can be used to simplify the expression in **part (a)(i)**.

Draw loops around appropriate groups of 1s in the table in **part (a)(ii)** to produce an optimal sum-of-products. [2]

(iv) Write the simplified sum-of-products expression for your answer to **part (a)(iii)**.

X = ... [2]

(b) Simplify your expression for **X** in **part (a)(i)** using Boolean algebra. Show your working.

....

....

....

.... [2]

5 Flip-flops are used in computer construction.

(a) Describe the role of flip-flops in a computer.

....

....

....

....

....

.... [3]

(b) Describe the difference between an SR flip-flop and a JK flip-flop.

....

....

....

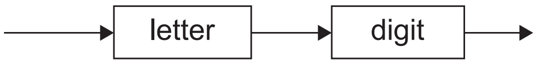
.... [2]

6 The syntax diagrams for a programming language show the syntax of:

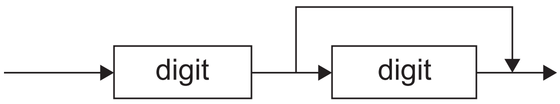
- an assignment statement
- a variable
- an unsigned integer
- a letter
- an operator
- a digit

assignment statement

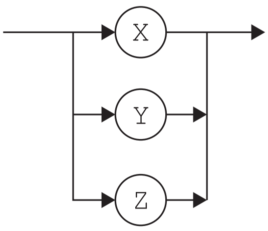
variable



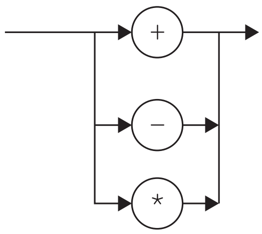
unsigned integer



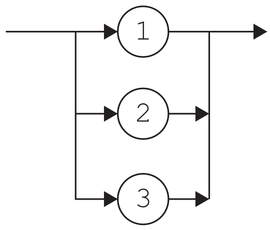
letter



operator



digit



(a) Give reasons why each of these statements is **invalid**.

```
X1 = Y - 21
```

....

```
Y3 := Y3 + 1
```

....

```
X1 = X2 * 7
```

....

[3]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.
`<letter>` has been completed for you.

`<letter> ::= X|Y|Z`

`<assignment_statement> ::=`

....

`<variable> ::=`

....

`<unsigned_integer> ::=`

....

`<operator> ::=`

....

`<digit> ::=`

....
[5]

(c) The syntax of an **assignment statement** is changed to allow for a variable or an unsigned integer before and after the operator.

(i) Draw an updated syntax diagram for the **assignment statement**.

[2]

(ii) Give the BNF for the revised **assignment statement** syntax.

....

....

....

....

.... [2]

7 **(a)** A digital certificate and a digital signature are used to ensure that a message is not changed during transmission.

Write an appropriate term in each space to complete the descriptions.

A digital certificate contains the .... key of the owner. A digital certificate is obtained from the ....

Before a private message is sent to the owner of the digital certificate, this key is used to .... the message.

A digital signature is also sent. The message is hashed to produce a .... , which is then encrypted with the sender's .... key to obtain the digital signature. [5]

(b) State **two** encryption protocols used in data transmission.

1 ....

2 ....
[2]

(c) Malware can harm computer systems.

Describe **two** methods that can be used to restrict the effect of malware.

....

....

....

....

....

....

....

.... [4]

8 Four greenhouses are used to grow tomatoes. The temperature inside each greenhouse should be kept between 10 and 20 degrees Celsius inclusive.
Each greenhouse has a temperature sensor.

A computer system is programmed to control each greenhouse's temperature by:

- turning on the heater and closing the ventilation when the temperature falls below 10 degrees
- turning off the heater and opening the ventilation when the temperature rises above 20 degrees.

(a) **(i)** State the name given to the type of system described.

.... [1]

(ii) Justify your answer to **part (i)**.

....

....

....

.... [2]

(b) The computer system stores the temperature readings for the four sensors in two's complement form and in four eight-bit memory locations with addresses `701` to `704`.

`701`	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**Greenhouse 1**
`702`	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**Greenhouse 2**
`703`	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**Greenhouse 3**
`704`	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**Greenhouse 4**

State the greenhouse number(s) where the temperature is out of range **and** give the value(s) of these temperature(s) in denary.

....

....

....

.... [2]

(c) The status of the heaters and the ventilation is shown at location `700`.

- A value of 1 means that the heater is on.
- A value of 0 (zero) means that the heater is off.
- A value of 1 means that the ventilation is open.
- A value of 0 (zero) means that the ventilation is closed.

The status of the heaters is shown in the most significant four bits; the status of the ventilation is shown in the least significant four bits.

The pattern of bits at location `700` shows that the heater for greenhouse 3 is on and the ventilation for greenhouse 1 is open.

Greenhouse number

	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
`700`	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**
	Heater				**Ventilation**			

Show the pattern of bits when the heater is on for greenhouses 1 and 2 only and no ventilation is open.

`700`								

[1]

(d) The table shows assembly language instructions for the greenhouse computer system that has one general purpose register, the accumulator (ACC).

Instruction			**Explanation**
Label	**Op code**	**Operand**	
	`LDM`	`&n`	Load the hexadecimal number n to ACC
	`LDD`	`<address>`	Load the contents of the location at the given address to ACC
	`STO`	`<address>`	Store the contents of ACC at the given address
	`AND`	`&n`	Bitwise AND operation of the contents of ACC with the hexadecimal number n
	`LSL`	`#n`	Bits in ACC are shifted denary number n places to the left. Zeros are introduced at the right hand end
	`CMP`	`&n`	Compare the contents of ACC with the hexadecimal number n
	`JPE`	`<address>`	Following a compare instruction, jump to <address> or <label> if the compare was True
`<label>:`	`<op code>`	`<operand>`	Labels an instruction

If the bit for a greenhouse's heater and the bit for the same greenhouse's ventilation are both set to 1, a routine at label `ERROR` is executed. This routine has not been provided.

(i) These assembly language instructions check for an error in the greenhouse 1 system.

```
LDD   700
AND   &88
CMP   &88
JPE   ERROR
```

Explain the purpose of each instruction.

....

....

....

....

....

....

....

.... [4]

(ii) Write the assembly language instructions to check for an error in the greenhouse 2 system.

....

....

....

....

....

.... [3]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

1261665674

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2019**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **13** printed pages and **3** blank pages.

DC (LK) 179424

Cambridge Assessment International Education

1 In a computer system, real numbers are stored using normalised floating-point representation with:

- twelve bits for the mantissa
- four bits for the exponent.

The mantissa and exponent are both in two's complement form.

(a) Calculate the denary value for the following binary floating-point number.

Show your working.

Mantissa

1	0	0	1	0	1	1	1	0	0	1	1

Exponent

0	1	1	1

Working ....

....

....

....

....

Answer ....
[3]

(b) Calculate the normalised floating-point representation of +1.5625 in this system.

Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(c) **(i)** Write the largest positive number that can be stored as a normalised floating-point number using this format.

Mantissa | | | | | | | | | | | | **Exponent** | | | |

[2]

(ii) Write the smallest non-zero positive number that can be stored as a normalised floating-point number using this format.

Mantissa | | | | | | | | | | | | **Exponent** | | | |

[2]

(d) The developer of a new programming language decides that all real numbers will now be stored using 20-bit normalised floating-point representation. She must decide how many bits to use for the mantissa and how many bits for the exponent.

Explain the trade-off between using either a large number of bits for the mantissa, or a large number of bits for the exponent.

....

....

....

....

....

.... [3]

2 Cables connect the computers in a university admissions department in a star topology. The server room contains the server and printer for the employees to use. The department has three employees. Each employee has a computer connected to the star network.

(a) **(i)** Draw a diagram to show this topology.

[3]

(ii) Explain the benefits to the admissions department of using a star topology.

....

....

....

....

....

....

....

....

....

.... [4]

(b) Each department of the university has its own network. All the department networks connect to the university's main Local Area Network (LAN). The LAN has a bus topology and uses the CSMA/CD protocol.

Describe the CSMA/CD protocol.

....

....

....

....

....

....

.... [3]

(c) Explain how the following devices are used to support the university LAN.

(i) Router ....

....

....

.... [2]

(ii) Network Interface Card (NIC) ....

....

....

.... [2]

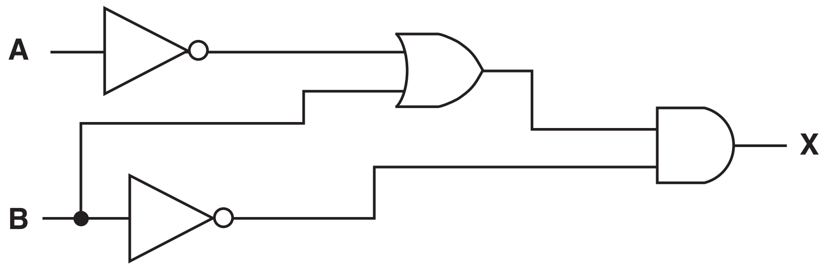
(iii) Wireless Access Point ....

....

....

.... [2]

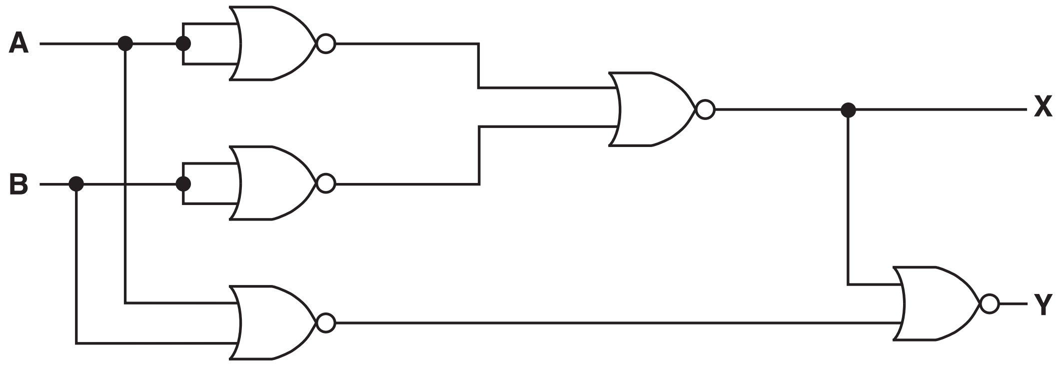
3 **(a)** The following logic circuit can be simplified to use only one gate.



Give the name of this single gate.

.... [1]

(b) **(i)** Complete the truth table for the logic circuit.



A	B	Working space	X	Y
0	0			
0	1			
1	0			
1	1			

[2]

(ii) Give the name of the logic circuit that has this truth table.

.... [1]

(iii) Give the uses for outputs **X** and **Y**.

X ....

Y ....

[2]

(c) Consider the following Boolean algebraic expression:

$$\overline{A} . \overline{B} . \overline{C} . \overline{D} + \overline{A} . \overline{B} . \overline{C} . D + \overline{A} . \overline{B} . C . D + \overline{A} . \overline{B} . C . \overline{D} + \overline{A} . B . \overline{C} . \overline{D}$$

Use Boolean algebra to simplify the expression. Show your working.

Working ....

....

....

....

....

....

....

....

....

Simplified expression .... [5]

4 A compiler uses a keyword table and a symbol table. Part of the keyword table is shown.

- Tokens for keywords are shown in hexadecimal.
- All of the keyword tokens are in the range 00 – 5F.

Keyword	Token
←	01
+	02
=	03
<>	04
IF	4A
THEN	4B
ENDIF	4C
ELSE	4D
REPEAT	4E
UNTIL	4F
TO	50
INPUT	51
OUTPUT	52
ENDFOR	53

Entries in the symbol table are allocated tokens. These values start from 60 (hexadecimal).

Study the following piece of pseudocode.

```
Counter ← 0
INPUT Password
REPEAT
   IF Password <> "Cambridge"
      THEN
         INPUT Password
   ENDIF
   Counter ← Counter + 1
UNTIL Password = "Cambridge"
OUTPUT Counter
```

(a) Complete the symbol table to show its contents after the lexical analysis stage.

Symbol	**Token**	
	Value	**Type**
`Counter`	`60`	Variable

[3]

(b) The output from the lexical analysis stage is stored in the following table. Each cell stores one byte of the output.

Complete the output from the lexical analysis using the keyword table **and** your answer to **part (a)**.

60	01																							

[2]

(c) The following table shows assembly language instructions for a processor which has one general purpose register, the Accumulator (ACC).

Instruction		Explanation
Op code	**Operand**	
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC
`ADD`	`<address>`	Add the contents of the given address to the ACC
`STO`	`<address>`	Store the contents of ACC at the given address

After the syntax analysis is completed successfully, the compiler generates object code.

The following lines of high level language code are compiled.

```
X = X + Y
Z = Z + X
```

The compilation produces the assembly language code as follows:

```
LDD 236
ADD 237
STO 236
LDD 238
ADD 236
STO 238
```

(i) The final stage in the compilation process that follows this code generation stage is code optimisation.

Rewrite the equivalent code after optimisation.

....

....

....

....

....

.... [3]

(ii) Explain why code optimisation is necessary.

....

....

....

.... [2]

5 **(a)** Wiktor is an employee of a travel agent. He uses asymmetric encryption to send confidential information to his manager.

Fill in the spaces with an appropriate term to complete the descriptions.

Asymmetric encryption uses different .... for encrypting and decrypting data. When Wiktor sends a message to his manager, the message is encrypted into .... using his manager's .... key. When the manager receives the message, it is decrypted using her .... key.

When the manager replies, the message is encrypted using Wiktor's .... key, and when Wiktor receives the message, it is decrypted into .... using his .... key. [5]

(b) When customers pay for their travel booking online, a secure connection is established using Secure Socket Layer (SSL).

Explain how the customer's browser and the server used to collect the payment will establish a secure connection.

....

....

....

....

....

....

....

....

....

....

....

.... [6]

(c) The manager is concerned about the threat of malware to the company computer systems.

Name **two** types of malware. State what the company should do to help prevent the effect of the malware.

The two methods of prevention must be different.

Malware type 1 ....

Prevention ....

....

Malware type 2 ....

Prevention ....

....

[4]

6 Monitoring and control systems have many different applications.

(a) Explain the importance of feedback in a control system.

....

....

....

....

....

.... [3]

(b) An indoor swimming pool is to be kept at a constant temperature of 28 degrees.

Describe the use of feedback in this control system.

....

....

....

....

....

....

....

.... [4]

(c) Give **one** example of a monitoring system. Explain why this is a monitoring system.

Monitoring system ....

....

Explanation ....

....

....

....

[3]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

7825200973

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2018**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **13** printed pages and **3** blank pages.

DC (RW/CT) 150986/3

1 **(a)** A computer system uses floating-point representation to store real numbers. The floating-point numbers have:

- 8 bits for the mantissa
- 8 bits for the exponent

The mantissa and exponent are both in two's complement form.

(i) Calculate the denary value of the following floating-point number. It is **not** in normalised form.

Mantissa								**Exponent**							
0	0	1	0	1	0	1	0	0	0	0	0	0	1	0	1

Show your working.

Working ....

....

....

....

....

Answer .... [3]

(ii) Convert the denary number +7.5 into a normalised floating-point number.

Show your working.

Mantissa								**Exponent**							

Working ....

....

....

....

.... [3]

(iii) Convert the denary number −7.5 into a normalised floating-point number.

Show your working.

Mantissa								**Exponent**							

Working ....

....

....

....

.... [3]

(b) A normalised floating-point number is shown.

Mantissa								**Exponent**							
0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1

(i) State the significance of this binary number.

....

....[1]

(ii) State what will happen if a positive number is added to this number.

....

....[1]

2 **(a)** A network can be set up using a star topology.

Give **three** features of a star topology.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) **(i)** Describe what is meant by **circuit switching**.

....

....

....

....[2]

(ii) The table shows statements that relate to circuit switching, packet switching or both.

Tick (✓) **one or more** boxes in each row to show whether the statement applies to circuit switching, packet switching or both.

Statements	**Circuit switching**	**Packet switching**
Shares bandwidth		
Data may arrive out of order		
Data can be corrupted		
Data are less likely to get lost		

[4]

3 **(a)** Consider the following Boolean expression.

$$A . \overline{B} . \overline{C} + A . B . \overline{C} + A . B . C$$

Use Boolean algebra to simplify the expression.

....

....

....

....

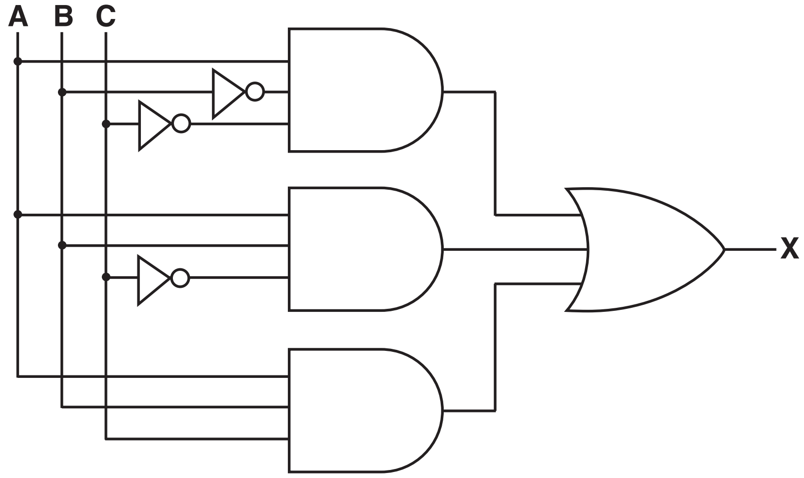
....

....

....

....[4]

(b) **(i)** Complete the truth table for the following logic circuit.



A	B	C	Working space	X
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

[2]

(ii) Complete the Karnaugh Map (K-map) for the truth table in **part (b)(i)**.

AB

C		**00**	**01**	**11**	**10**
	0				
	1				

[1]

(iii) Draw loops around appropriate groups of 1s in the table in **part (b)(ii)** to produce an optimal sum-of-products. [2]

(iv) Using your answer to **part (b)(iii)**, write a simplified sum-of-products Boolean expression.

X = ...[2]

(c) The truth table for a logic circuit with four inputs is shown.

INPUT				OUTPUT
A	**B**	**C**	**D**	**X**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(i) Complete the K-map for the truth table in **part (c)**.

AB

CD					

[4]

(ii) Draw loops around appropriate groups of 1s in the table in **part (c)(i)** to produce an optimal sum-of-products. [2]

(iii) Using your answer to **part (c)(ii)**, write a simplified sum-of-products Boolean expression.

X = ...[2]

4 A compiler uses a keyword table and a symbol table. Part of the keyword table is shown.

- Tokens for keywords are shown in hexadecimal.
- All of the keyword tokens are in the range `00 – 5F`.

Keyword	Token
←	`01`
+	`02`
=	`03`
`IF`	`4A`
`THEN`	`4B`
`ENDIF`	`4C`
`ELSE`	`4D`
`FOR`	`4E`
`STEP`	`4F`
`TO`	`50`
`INPUT`	`51`
`OUTPUT`	`52`
`ENDFOR`	`53`

Entries in the symbol table are allocated tokens. These values start from `60` (hexadecimal).

Study the following code.

```
INPUT Number1
INPUT Number2
INPUT Answer
IF Answer = Number1 + Number2
   THEN
      OUTPUT 10
   ELSE
      OUTPUT 0
ENDIF
```

(a) Complete the symbol table to show its contents after the lexical analysis stage.

Symbol	Token	
	Value	Type
`Number1`	`60`	Variable
`Number2`	`61`	Variable

[3]

(b) The output from the lexical analysis stage is stored in the following table. Each cell stores one byte of the output.

Complete the output from the lexical analysis. Use the keyword table and your answer to **part (a)**.

51	60																	

[2]

(c) A student uses the compiler to compile some different code.

After the syntax analysis is complete, the compiler generates object code.

The following line of code is compiled: `X ← A + B + C − D`

The compilation produces the following assembly language code.

```
LDD 236      // loads value A into accumulator
ADD 237      // adds value B to accumulator
ADD 238      // adds value C to accumulator
STO 540      // stores accumulator in temporary location
LDD 540      // loads value from temporary location into accumulator
SUB 239      // subtracts value D from accumulator
STO 235      // stores accumulator in X
```

(i) Identify the final stage in the compilation process that follows this code generation stage.

....[1]

(ii) Rewrite the equivalent code following the final stage.

....

....

....

....

....

....

....

....[3]

(iii) State **two** benefits of the process that is carried out in the final stage.

Benefit 1 ....

....

Benefit 2 ....

....

[2]

(d) An interpreter is executing a program. The program uses the variables `a`, `b`, `c` and `d`.

The program contains an expression that is written in infix form. The interpreter converts the infix expression to RPN.

The RPN expression is: `b a c + * d + 2 −`

The interpreter evaluates this RPN expression using a stack.

The current values are: `a = 1` `b = 2` `c = 2` `d = 3`

Show the changing contents of the stack as the interpreter evaluates the expression.

The first entry on the stack has been done for you.

2								

[4]

5 (a) Most desktop or laptop computers use CISC (Complex Instruction Set Computing) architecture. Most smartphones and tablets use RISC (Reduced Instruction Set Computing).

State **four** features that are different for the CISC and RISC architectures.

1 ....

....

2 ....

....

3 ....

....

4 ....

....

[4]

(b) In a RISC processor, four instructions (**A**, **B**, **C**, **D**) are processed using pipelining.

The following table shows five stages that take place when instructions are fetched and executed. In time interval **1**, instruction **A** has been fetched.

(i) In the table, write the instruction labels (**A**, **B**, **C**, **D**) in the correct time interval for each stage. Each operation only takes one time interval.

Stage	**Time interval**								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Fetch instruction	**A**								
Decode instruction									
Execute instruction									
Access operand in memory									
Write result to register									

[3]

(ii) When completed, the table in **part (b)(i)** shows how pipelining allows instructions to be carried out more rapidly. Each time interval represents one clock cycle.

Calculate how many clock cycles are saved by using pipelining in the example in **part (b)(i)**.

Show your working.

Working ....

....

....

Answer ....

[3]

[Turn over

(c) The table shows four statements about computer architecture.

Put a tick (✓) in each row to identify the computer architecture associated with each statement.

Statement	Architecture		
	SIMD	**MIMD**	**SISD**
Each processor executes a different instruction			
There is only one processor			
Each processor executes the same instruction input using data available in the dedicated memory			
Each processor typically has its own partition within a shared memory			

[4]

6 **(a)** The following table shows descriptions and terms relating to data transmission security.

Add appropriate descriptions and terms to complete the table.

	Description	**Term**
A	The result of encryption that is transmitted to the recipient.	....
B	The type of cryptography used where different keys are used; one for encryption and one for decryption.	....
C	....	**Digital certificate**
D	....	**Private key**

[4]

(b) The sequence of steps 1 to 7 describes what happens when setting up a secure connection using Secure Socket Layer (SSL).

Four statements are missing from the sequence.

A	If the browser trusts the certificate, it creates, encrypts and sends the server a symmetric session key using the server's public key.
B	Server sends the browser an acknowledgement, encrypted with the session key.
C	Server sends a copy of its SSL Certificate and its public key.
D	Server decrypts the symmetric session key using its private key.

Write **one** letter (**A** to **D**) in the appropriate space to complete the sequence.

1. Browser requests that the server identifies itself.
2. ……………
3. Browser checks the certificate against a list of trusted Certificate Authorities.
4. ……………
5. ……………
6. ……………
7. Server and browser now encrypt all transmitted data with the session key.

[3]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

1596084089

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2020**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Blank pages are indicated.

DC (PQ) 182206/3

1 Amar is alerted to a run-time error when he runs a program.

(a) A run-time error occurs when Amar attempts to open a file that does not exist.

State **three other** reasons why a run-time error may occur.

1 ....

....

2 ....

....

3 ....

....
[3]

(b) A program should be written with exception handling routines to manage run-time errors.

A program reads data from the text file `MyData.txt`. The program needs to report an exception if it attempts to open the file and the file does not exist.

Write **program code** to handle and report this exception.

Programming language ....

Program code

....

....

....

....

....

....

....

....

.... [2]

2 A business is developing a program that stores each customer's username and password in a hash table.

The hash table will be implemented as a 1D array, `CustomerLogIn`, of the custom data type `CustomerRecord`.

The declaration for `CustomerRecord` is:

```
TYPE CustomerRecord

      DECLARE Username : STRING

      DECLARE Password : STRING

ENDTYPE
```

The hash table will store a maximum of 3000 records. The key field will be `Username`.

(a) The program declares the hash table and initialises the username and password of all the records to an empty string.

Write **pseudocode** to declare **and** initialise the hash table.

....

....

....

....

....

....

....

.... [4]

(b) **(i)** A function, `SearchHashTable()` will search for a customer's record in the hash table.

The function `Hash()`:

- takes a `Username` as a parameter
- performs the hashing algorithm
- returns the calculated index of the username within the hash table.

The function `SearchHashTable()`:

- takes the username to search for as a parameter
- uses `Hash()` to calculate the first index of this username within the hash table
- returns either the index of the username if found, or –1 if not found.

Complete the **pseudocode** for the function `SearchHashTable()`.

```
FUNCTION SearchHashTable(BYVALUE SearchUser : STRING) RETURNS ..........................
   DECLARE Index : INTEGER
   DECLARE Count : INTEGER
   Index ← ............................................ (SearchUser)
   Count ← 0
   WHILE (CustomerLogIn[Index] ....................................... <> .........................................)
         AND(CustomerLogIn[Index].Username <> "")
         AND(Count < 2999)
      Index ← Index + 1
      Count ← Count + 1
      IF Index > ............................................
         THEN
            Index ← 0
      ENDIF
   ENDWHILE
      IF CustomerLogIn[Index].Username = ............................................
         THEN
            RETURN Index
         ELSE
            RETURN ............................................
      ENDIF
   ENDFUNCTION
```

[7]

(ii) Explain the purpose of the variable `Count` in the function `SearchHashTable()`.

....

....

....

.... [2]

3 Recursive algorithms can be used when creating programs.

(a) Describe what is meant by a **recursive algorithm**.

....

....

....

....

....

.... [3]

(b) A string that is a palindrome, reads the same forwards as it does backwards. For example, the name Anna is a palindrome.

The function `Substring(Variable, StartingCharacter, NumberOfCharacters)` returns one or more characters from a string. The first character is at position 0.

For example, the string `"Happy"` is stored in the variable `Word`.

- `Substring(Word, 1, 1)` would return the character `"a"`.
- `Substring(Word, 2, 3)` would return the characters `"ppy"`.

The function `Length()` returns the length of the string as an integer. For example, `Length(Word)` returns `5`.

The following is a recursive function to find out whether a string is a palindrome. The function returns `True` if the parameter is a palindrome, and returns `False` if it is not a palindrome.

Complete the **pseudocode** for the recursive algorithm to indicate whether a string is a palindrome.

```
FUNCTION IsPalindrome(CheckWord : STRING) RETURNS BOOLEAN
   IF ................................................................ <= 1
      THEN
         RETURN ........................................
   ENDIF
   IF Substring(CheckWord, 0, 1) <>
         Substring(CheckWord, ........................................ (CheckWord)-1, 1)
      THEN
         RETURN ........................................
      ELSE
         RETURN ........................................(Substring(CheckWord, 1,
                                                Length(CheckWord)-2))
   ENDIF
ENDFUNCTION
```

[5]

(c) The function `FindPower()` is a recursive function that calculates the result of a base number to the power of the exponent. For example, the result of $2^4 = 16$, as 2*2*2*2 = 16. In this example, 2 is the base number and 4 is the exponent.

The base number and the exponent are passed as parameters.

Write **pseudocode** for the recursive function `FindPower()`. Assume both the base and the exponent are positive integers.

....

....

....

....

....

....

....

....

....

....

....

.... [5]

4 **(a)** A tennis club has a booking form to book lessons with an instructor.

Club members can book up to five lessons using the booking form.

The customer details section has the data:

- name
- address
- telephone number.

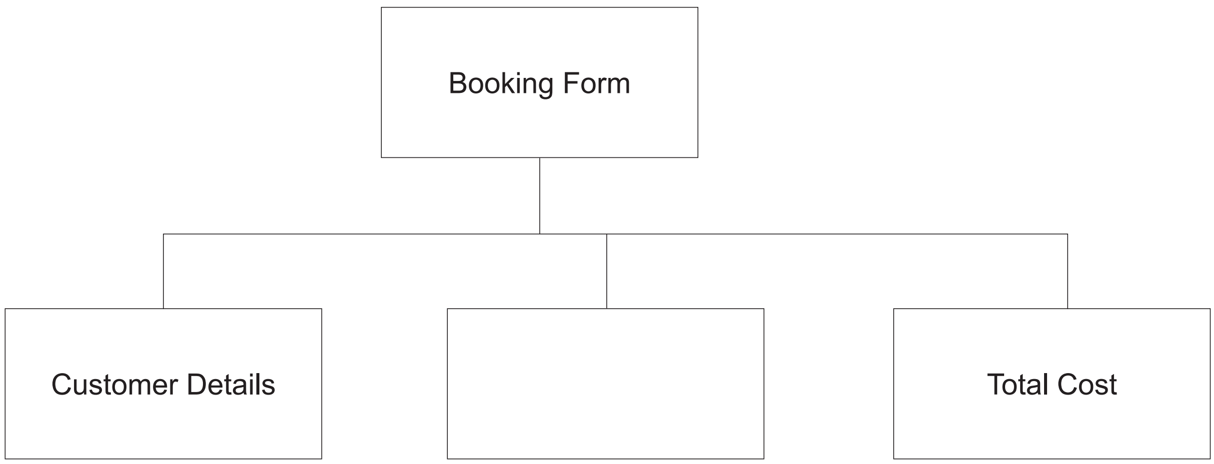
The lesson details section has the data:

- lesson type
- date and time
- lesson cost.

The cost of each lesson is dependent on the customer's type of membership. The membership can be bronze, silver or gold.

The total cost is also calculated.

Complete the following JSP data structure diagram for the booking form.



[7]

(b) State **two** programming constructs that are shown in a JSP data structure diagram.

1 ....

....

2 ....

.... [2]

5 A declarative programming language is used to represent the following knowledge base.

```
01 person(william).
02 person(deeraj).
03 person(ingrid).
04 person(meghan).
05 country(england).
06 country(spain).
07 country(bangladesh).
08 country(new_zealand).
09 country(malaysia).
10 country(mauritius).
11 visited(william, spain).
12 visited(ingrid, new_zealand).
13 visited(deeraj, spain).
14 visited(meghan, spain).
```

These clauses have the following meanings:

Clause	Meaning
02	Deeraj is a person
05	England is a country
11	William has visited Spain

(a) Gina is a person who has visited Cyprus.

Write additional clauses to represent this information.

15 ....

16 ....

17 .... [3]

(b) Write the result returned by the goal:

```
visited(X, spain).
```

X = ... [2]

(c) P might visit C, if P is a person, C is a country and P has not visited C.

Write this as a rule.

```
mightvisit(P , C)
```

IF ....

....

.... [4]

6 Object-oriented programming has several features. These include containment, classes, methods and properties.

(a) Describe what is meant by **containment**.

....

....

....

....

....

.... [3]

(b) Identify **two other** features of object-oriented programming.

1 ....

2 ....

[2]

7 A programmer is creating a computer game. The programmer has designed the class, `Character`, for the characters in the game.

The following class diagram shows the design for the `Character` class.

```
                              Character
-------------------------------------------------------------------------
Name : STRING      //  initialised in constructor to the parameter
                       value passed to the constructor
Skill : INTEGER    //  initialised in constructor to 0
Health : INTEGER   //  initialised in constructor to 50
Shield : INTEGER   //  initialised in constructor to a random value
                       between 1 and 25 (inclusive)
-------------------------------------------------------------------------
Constructor()      //  method used to create and initialise an object
GetName()          //  returns Name value
GetSkill()         //  returns Skill value
GetHealth()        //  returns Health value
GetShield()        //  returns Shield value
SetSkill()         //  increases Skill by the parameter value
SetHealth()        //  increases or decreases Health by the parameter
                       value
SetShield()        //  increases or decreases Shield value by the
                       parameter value
```

(a) Write **program code** for the `Constructor()` method. Use the appropriate constructor method for your chosen programming language.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(b) Write **program code** for the `GetSkill()` method.

Programming language ....

Program code

....

....

....

....

.... [2]

(c) The method `SetSkill()` validates the parameter value and updates the value of `Skill`.

The method is passed an `INTEGER` parameter that must be between 10 and 25 (inclusive). A value outside of this is not valid.

If the parameter value is valid, the method will increase `Skill` by the parameter value. The maximum value that `Skill` can be increased to is 200. For example:

- `Skill` currently stores 180
- it is passed a valid parameter value of 25
- `Skill` will now store 200.

The method must return:

- `−1` if the parameter value is not valid
- `1` if the value of `Skill` is updated **and** `Skill` is less than 200
- `0` if the value of `Skill` is `200`.

Write **program code** for the `SetSkill()` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

.... [6]

(d) There are five characters in the game. All the character objects are stored in a 1D array.

Write **pseudocode** to declare the array, `CharacterArray`, to store the five character objects.

....

.... [2]

(e) The game has the character with the name Victory.

Write **program code** to create the character Victory as an instance of the class `Character`. The object needs to be stored in the first element of the array `CharacterArray`.

Programming language ....

Program code

....

....

.... [3]

Question 8 begins on the next page.

8 Files can be structured in serial, sequential or random format.

Tick (✓) **one** box in each row to show whether the statement applies to **Serial**, **Sequential** or **Random** format.

Statement	Serial	Sequential	Random
Uses a hashing algorithm			
No key field is used when storing data, for example, it is stored in chronological order			
Collisions can occur			
Least efficient for a very large number of records			
Most efficient for a very large number of records			

[3]

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 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 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.
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.

The maximum number of marks is 75.

This document consists of **17** printed pages and **3** blank pages.

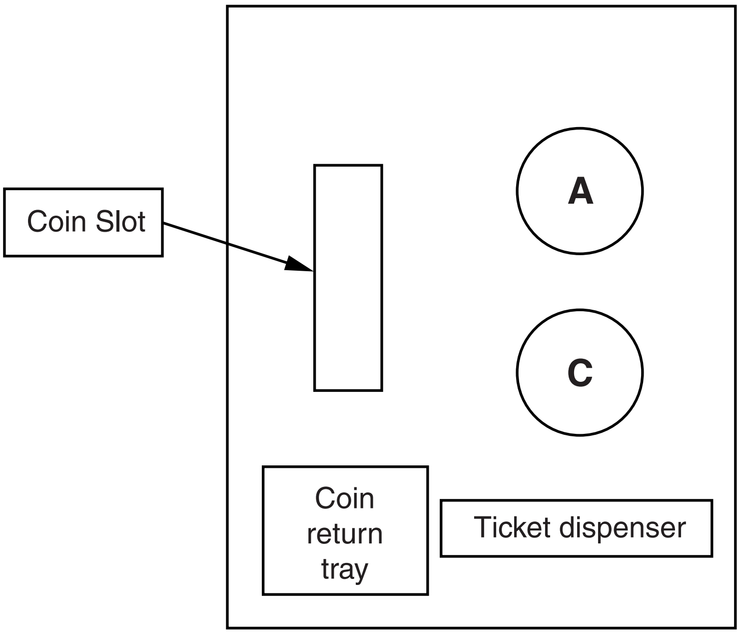
DC (LEG/SW) 115870/4

CAMBRIDGE International Examinations

1 The ticket machine in the following diagram accepts the following coins: 10, 20, 50 and 100 cents.

The ticket machine has:

- a slot to insert coins
- a tray to return coins
- a ticket dispenser
- two buttons:
  - button **A** (Accept)
  - button **C** (Cancel)



When the user has inserted as many coins as required, they press button **A** to print the ticket.

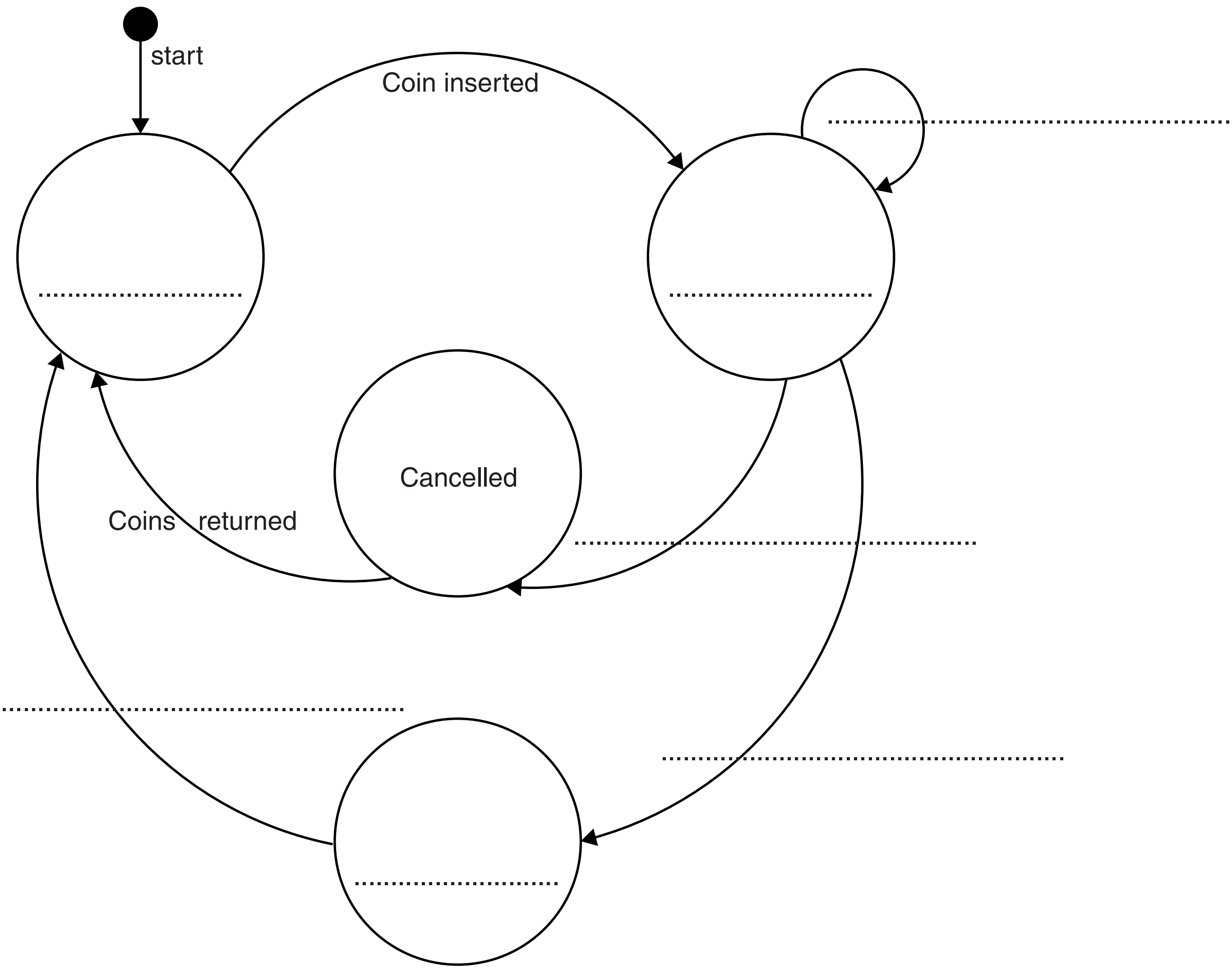
To cancel the transaction, the user can press button **C**. This makes the machine return the coins.

Invalid coins have no effect.

The following state transition table shows the transition from one state to another of the ticket machine:

Current state	Event	Next state
Idle	Coin inserted	Counting
Counting	Coin inserted	Counting
Counting	Button **C** pressed	Cancelled
Cancelled	Coins returned	Idle
Counting	Button **A** pressed	Accepted
Accepted	Ticket printed	Idle

(a) Complete the state-transition diagram.



[7]

(b) A company wants to simulate the use of a ticket machine. It will do this with object-oriented programming (OOP).

The following diagram shows the design for the class `TicketMachine`. This includes its attributes and methods.

```
TicketMachine
-------------------------------------------------------------------------
Amount  : INTEGER // total value of coins inserted in cents
State   : STRING  // "Idle", "Counting", "Cancelled"
                  // or "Accepted"
-------------------------------------------------------------------------
Create()          // method to create and initialise an object
                  // if using Python use __init__
SetState()        // set state to parameter value
                  // and output new state
StateChange()     // insert coin or press button,
                  // then take appropriate action
CoinInserted()    // parameter is a string
                  // change parameter to integer
                  // and add coin value to Amount
ReturnCoins()     // output Amount, then set Amount to zero
PrintTicket()     // print ticket, then set Amount to zero
```

Write **program code** for the following methods.

Programming language ....

(i) `Create()`

....

....

....

....

....[3]

(ii) `SetState()`

....

....

....

....[2]

(iii) `ReturnCoins()`

....

....

....

.... [2]

(iv) Each coin inserted must be one of the following: 10, 20, 50 or 100 cents.

Write **program code** for a function `ValidCoin(s : STRING)` that returns:

- `TRUE` if the input string is one of `"10", "20", "50"` or `"100"`
- `FALSE` otherwise

Programming language ....

....

....

....

....

....

....

....

....

....

....[3]

(v) Write **program code** for the method `CoinInserted()`

....

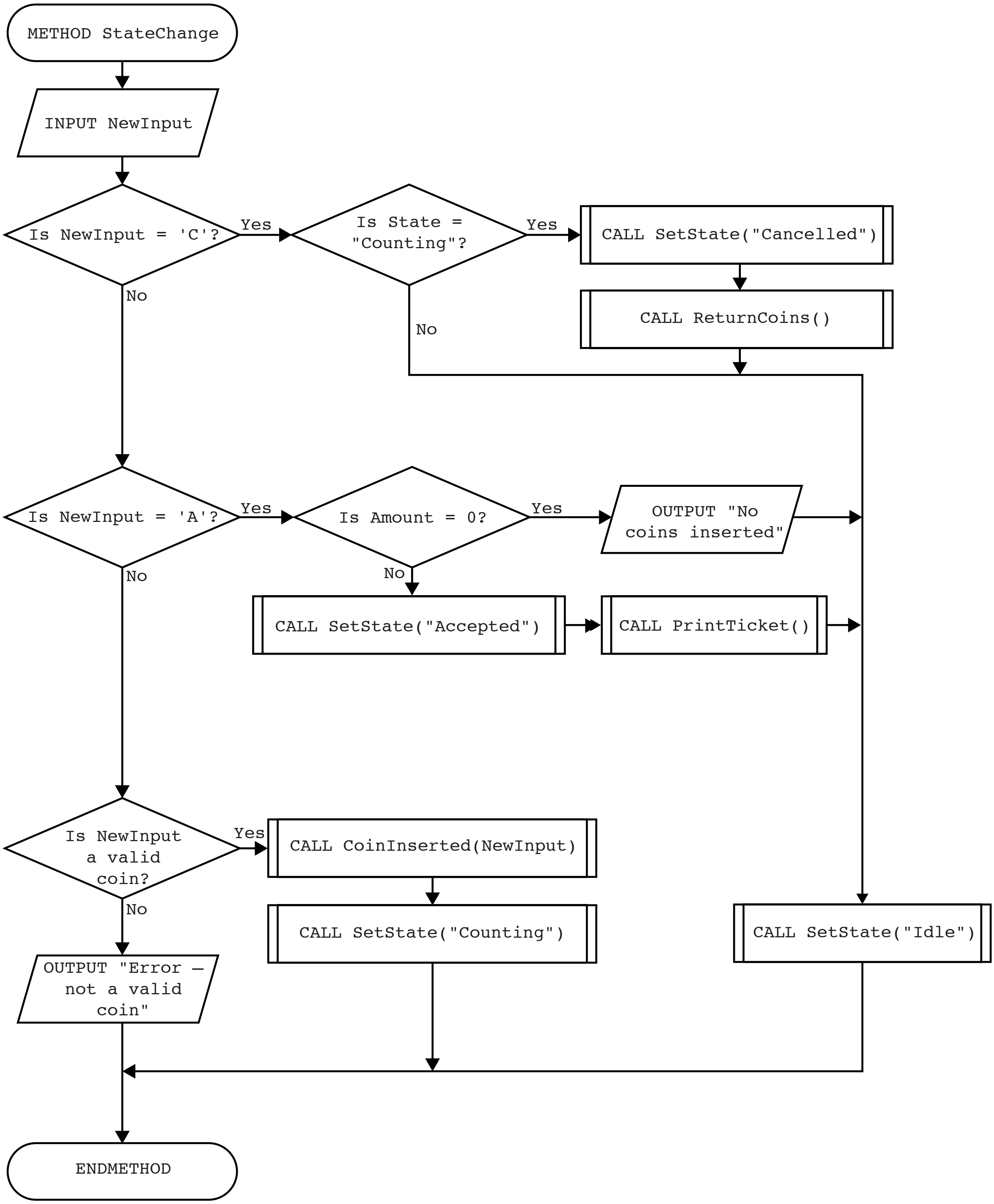
....

....

....

....[2]

(vi) Convert the flowchart to **program code** for the method `StateChange()`.
Use the attributes and methods in the original class definition and the `ValidCoin()` function from **part (iv)**.



Programming language ....

....[12]

(vii) The company needs to write a program to simulate a parking meter. The program will create an object with identifier `ParkingMeter`, which is an instance of the class `TicketMachine`.

The main program design is:

```
instantiate ParkingMeter (create and initialise ParkingMeter)
loop forever (continually use ParkingMeter)
   call StateChange() method
end loop
```

Write **program code** for the main program.

Programming language ....

....

....

....

....

....

....

....

....

....[4]

(c) It is possible to declare attributes and methods as either public or private.

A programmer has modified the class design for `TicketMachine` as follows.

```
TicketMachine
---------------------------
PRIVATE
   Amount : INTEGER
   State  : STRING
---------------------------
PUBLIC
   Create()
   StateChange()
PRIVATE
   SetState()
   CoinInserted()
   ReturnCoins()
   PrintTicket()
```

(i) Describe the effects of declaring the `TicketMachine` attributes as private.

....

....

....

....[2]

(ii) Describe the effects of declaring two methods of the class as public and the other four as private.

....

....

....

....[2]

2 Commercial software usually undergoes alpha testing and beta testing.

Distinguish between the two types of testing by stating:

- who does the testing
- when the testing occurs
- the specific purpose of each type of testing

(i) Alpha testing

Who ....

....

When ....

....

Purpose ....

....[3]

(ii) Beta testing

Who ....

....

When ....

....

Purpose ....

....[3]

3 **(a)** The numerical difference between the ASCII code of an upper case letter and the ASCII code of its lower case equivalent is 32 denary (32_{10}).

For example, `'F'` has ASCII code 70 and `'f'` has ASCII code 102.

	Bit number							
	7	6	5	4	3	2	1	0
ASCII code	ASCII code in binary							
70	0	1	0	0	0	1	1	0
102	0	1	1	0	0	1	1	0

The bit patterns differ only at bit number 5. This bit is 1 if the letter is lower case and 0 if the letter is upper case.

(i) A program needs a mask to ensure that a letter is in **upper case.**

Write the binary pattern of the mask in the space provided in the table below.

	Bit number							
	7	6	5	4	3	2	1	0
ASCII code	ASCII code in binary							
70	0	1	0	0	0	1	1	0
102	0	1	1	0	0	1	1	0
Mask								

Give the bit-wise operation that needs to be performed using the mask and the ASCII code.

....[2]

(ii) A program needs a mask to ensure that a letter is in **lower case**.

Write the binary pattern of the mask in the space provided in the table below.

	Bit number							
	7	6	5	4	3	2	1	0
ASCII code	ASCII code in binary							
70	0	1	0	0	0	1	1	0
102	0	1	1	0	0	1	1	0
Mask								

Give the bit-wise operation that needs to be performed using the mask and the ASCII code.

....[2]

The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC), and an index register (IX).

Instruction		Explanation
Op code	Operand	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the given address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`LDR`	`#n`	Immediate addressing. Load the number n into IX.
`STO`	`<address>`	Store the contents of ACC at the given address.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`CMP`	`#n`	Compare the contents of ACC with number n.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`AND`	`#n`	Bitwise AND operation of the contents of ACC with the operand.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`#n`	Bitwise XOR operation of the contents of ACC with the operand.
`XOR`	`<address>`	Bitwise XOR operation of the contents of ACC with the contents of <address>.
`OR`	`#n`	Bitwise OR operation of the contents of ACC with the operand.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

A programmer is writing a program that will output the first character of a string in upper case and the remaining characters of the string in lower case.

The program will use locations from address `WORD` onwards to store the characters in the string. The location with address `LENGTH` stores the number of characters that make up the string.

The programmer has started to write the program in the following table. The comment column contains descriptions for the missing program instructions.

(b) Complete the program using op codes from the given instruction set.

Label	Op code	Operand	Comment
START:			// initialise index register to zero
			// get first character of WORD
			// ensure it is in upper case using MASK1
			// output character to screen
			// increment index register
			// load 1 into ACC
			// store in COUNT
LOOP:			// load next character from indexed address WORD
			// make lower case using MASK2
			// output character to screen
			// increment COUNT starts here
			// is COUNT = LENGTH ?
			// if FALSE, jump to LOOP
			// end of program
COUNT:			
MASK1:			// bit pattern for upper case
MASK2:			// bit pattern for lower case
LENGTH:		4	
WORD:		B01100110	// ASCII code in binary for 'f'
		B01110010	// ASCII code in binary for 'r'
		B01000101	// ASCII code in binary for 'E'
		B01000100	// ASCII code in binary for 'D'

[12]

Question 4 begins on page 15.

4 Circle the programming language that you have studied:

Visual Basic (console mode) Python Pascal Delphi (console mode)

(a) **(i)** Name the programming environment you have used when typing in program code.

....

....

List **three** features of the editor that helped you to write program code.

1 ....

....

2 ....

....

3 ....

.... [3]

(ii) Explain when and how your programming environment reports a syntax error.

When ....

....

....

How ....

....

.... [2]

(iii) The table shows a module definition for `BubbleSort` in three programming languages.

Study **one** of the examples. Indicate your choice by circling A, B or C:

A **B** **C**

A) Python

```
def BubbleSort(SList, Max):
   NoMoreSwaps = False
   while NoMoreSwaps == False:
      NoMoreSwaps = True
      for i in (Max - 1):
         if SList[i] > SList[i + 1]:
            NoMoreSwaps = True
            Temp = SList[i]
            SList[i] = SList[i + 1]
            SList[i + 1] = Temp
```

B) Pascal/Delphi

```
PROCEDURE BubbleSort(VAR SList : ARRAY OF INTEGER; Max : INTEGER);
VAR NoMoreSwaps : BOOLEAN; i, Temp : INTEGER;
BEGIN
   REPEAT
      NoMoreSwaps := TRUE;
      FOR i := 1 TO (Max – 1)
         IF SList[i] > SList[i + 1]
            THEN
               BEGIN
                  NoMoreSwaps := TRUE;
                  Temp := SList[i];
                  SList[i] := SList[i + 1];
                  SList[i + 1] := Temp;
               END;
   UNTIL NoMoreSwaps;
END;
```

C) Visual Basic

```
Sub BubbleSort(ByRef SList() As Integer, ByVal Max As Integer)
   Dim NoMoreSwaps As Boolean, i, Temp As Integer
      Do
         NoMoreSwaps = True
         For i : 0 To (Max – 1)
            If SList(i) > SList(i + 1) Then
               NoMoreSwaps = True
               Temp = SList(i)
               SList(i) = SList(i + 1)
               SList(i + 1) = Temp
            End If
         Next
      Loop Until (NoMoreSwaps = True)
End Sub
```

The programming environment reported a syntax error in the `BubbleSort` code.

State the line number ....

Write the correct code for this line.

....[2]

(b) (i) State whether programs written in your programming language are compiled or interpreted.

....

....[1]

(ii) A programmer corrects the syntax error and tests the function. It does not perform as expected. The items are not fully in order.

State the type of error ....

Write the line number where the error occurs.

....

Write the correct code for this line.

....[2]

(iii) State the programming environment you have used when debugging program code.

....

Name **two** debugging features and describe how they are used.

1 ....

....

....

....

2 ....

....

....

....[4]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

6195603694

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

DC (LO) 214518

1 In a particular computer system, two real numbers, **A** and **B**, are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

Number A

Mantissa													Exponent			
1	1	0	0	0	0	0	0	0	0	0	0		0	0	1	0

Number B

Mantissa													Exponent			
0	1	1	1	0	0	0	0	0	0	0	0		1	1	1	1

(a) (i) Identify whether each number is positive or negative. Justify your answer.

Number A ....

....

Number B ....

....
[2]

(ii) Convert the binary values of the **mantissa** and the **exponent** for each number to their separate denary values.

A mantissa ....

....

A exponent ....

....

B mantissa ....

....

B exponent ....

....
[4]

(iii) Calculate the denary value of each floating-point number using your values from **part (a)(ii)**.

Number A ....

....

....

Number B ....

....

....
[2]

(b) State which number, **A** or **B**, is stored in normalised floating-point form. Justify your answer.

Number ....

Justification ....

....

....

....

....
[3]

2 The TCP/IP protocol suite can be viewed as a stack with **four** layers.

(a) Write the correct descriptions for the **two** layers **and** the correct layers for the **two** descriptions given in the following table.

Layer	**Description**
Application	
	Handles forwarding of packets
Internet / Network	
	Handles how data is physically sent

[4]

(b) **(i)** Explain why communication protocols are necessary.

....

....

....

.... [2]

(ii) Identify **and** describe **one other** communication protocol. State its purpose.

Protocol ....

Description ....

....

Purpose ....

....

[3]

3 Describe, with the aid of a diagram for each one, the bus and star network topologies.

Bus

Description ....

....

....

....

....

....

Star

Description ....

....

....

....

....

....

[6]

4 **(a)** The truth table for a logic circuit with four inputs is shown.

INPUT				**OUTPUT**
P	**Q**	**R**	**S**	**X**
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

(i) Write the Boolean expression for the truth table as a sum-of-products.

X = ... [2]

(ii) Complete the Karnaugh Map (K-map) for the truth table.

		PQ			
		00	**01**	**11**	**10**
RS	**00**				
	01				
	11				
	10				

[2]

(iii) The K-map can be used to simplify the expression in **part (a)(i)**.

Draw loops around appropriate groups of 1s in the table in **part (a)(ii)** to produce an optimal sum-of-products. [2]

(iv) Write the simplified sum-of-products expression for your answer to **part (a)(iii)**.

X = ... [2]

(b) Simplify your expression for **X** in **part (a)(i)** using Boolean algebra. Show your working.

....

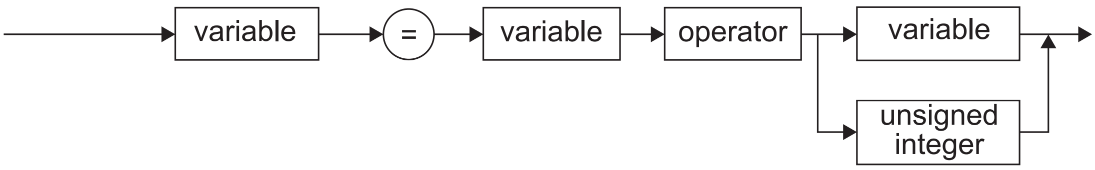
....

.... [2]

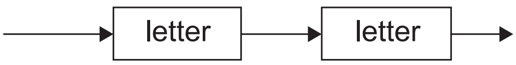
5 The following syntax diagrams for a programming language show the syntax of:

- an assignment statement
- a variable
- an unsigned integer
- a digit
- a letter
- an operator

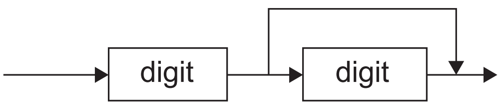
assignment statement



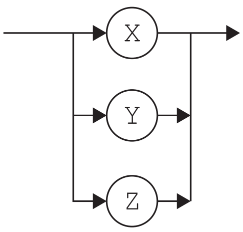
variable



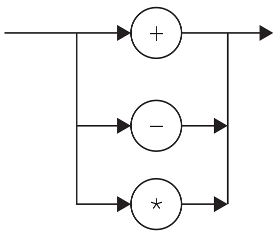
unsigned integer



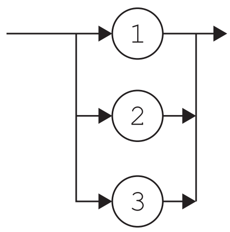
letter



operator



digit



(a) Give reasons why each of these statements is **invalid**.

`X = XY + 21`

....

`YZ := YZ * 3`

....

`XY = XY – 5`

....

[3]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.

`<letter>` has been completed for you.

`<letter> ::= X|Y|Z`

`<assignment_statement> ::=`

....

`<variable> ::=`

....

`<digit> ::=`

....

`<unsigned_integer> ::=`

....

`<operator> ::=`

....

[5]

(c) The syntax of a **variable** is changed to allow one or two letters followed by zero, one or two digits.

(i) Draw an updated syntax diagram for the **variable**.

[3]

(ii) Give the BNF for the revised **variable**.

....

....

....

....

.... [3]

6 Encryption is used to provide security when messages are transferred over a communication link.

(a) (i) Explain the way in which asymmetric key cryptography is used to encrypt a message being sent from one computer user to another over the Internet.

....

....

....

....

....

....

....

.... [4]

(ii) State **two** benefits of using asymmetric key cryptography.

1 ....

....

2 ....

....
[2]

(b) (i) Explain the way in which Transport Layer Security (TLS) provides communication security over a computer network.

....

....

....

....

....

....

....

.... [4]

(ii) State **two** situations where the use of TLS would be appropriate.

1 ....

....

2 ....

....
[2]

7 Four shipping containers are used to store goods on the dockside at a port. The temperature inside each container should be kept between 5 and 8 degrees Celsius inclusive.
Each container has a temperature sensor.

A computer system is programmed to control each container's temperature by:

- turning on the heater and turning off the air conditioning unit when the temperature falls below 5 degrees
- turning off the heater and turning on the air conditioning unit when the temperature rises above 8 degrees.

(a) **(i)** State the name given to the type of system described.

.... [1]

(ii) Justify your answer to **part (i)**.

....

....

....

.... [2]

(b) The computer system stores the temperature readings for the four sensors in two's complement form and in four eight-bit memory locations with addresses `301` to `304`.

`301`	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**Container 1**
`302`	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**Container 2**
`303`	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**Container 3**
`304`	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**Container 4**

State the container number(s) where the temperature is out of range **and** give the value(s) of these temperature(s) in denary.

....

....

....

.... [2]

(c) The status of the heaters and the air conditioning units is shown at location `300`.

A value of 1 means that the device is on and a value of 0 (zero) means that the device is off.

The status of the heaters is shown in the most significant four bits; the status of the air conditioning units is shown in the least significant four bits.

The pattern of bits at location `300` shows that the heater for container 4 is on and the air conditioning unit for container 1 is on.

Container number

	1	2	3	4	1	2	3	4
300	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**
	Heater				**Air conditioning**			

Show the pattern of bits when the heater is on for containers 1 and 2 and no air conditioning units are on.

300								

[1]

(d) The following table shows assembly language instructions for the container computer system that has one general purpose register, the Accumulator (ACC).

Instruction			**Explanation**
Label	**Op code**	**Operand**	
	LDM	&n	Load the hexadecimal number n to ACC
	LDD	<address>	Load the contents of the location at the given address to ACC
	STO	<address>	Store the contents of ACC at the given address
	AND	&n	Bitwise AND operation of the contents of ACC with the hexadecimal number n
	LSL	#n	Bits in ACC are shifted denary number n places to the left. Zeros are introduced at the right hand end
	CMP	&n	Compare the contents of ACC with the hexadecimal number n
	JPE	<address>	Following a compare instruction, jump to <address> or <label> if the compare was True
<label>:	<op code>	<operand>	Labels an instruction

If the bit for a container's heater and the bit for the same container's air conditioning unit are both set to 1, a routine at label ERROR is executed. This routine has not been provided.

(i) These assembly language instructions check for an error in the container 1 system.

```
LDD 300
AND &88
CMP &88
JPE ERROR
```

Explain the purpose of each instruction.

....

....

....

....

....

....

....

.... [4]

(ii) Write the assembly language instructions to check for an error in the container 4 system.

....

....

....

....

....

.... [3]

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

2155396653

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2016**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **11** printed pages and **1** blank page.

DC (NF/AR) 116626/4

CAMBRIDGE
International Examinations

[Turn over

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent

(a) Calculate the floating-point representation of +2.5 in this system. Show your working.

Mantissa **Exponent**

	•										

....

....

....

....

....

.... [3]

(b) Calculate the floating-point representation of −2.5 in this system. Show your working.

Mantissa **Exponent**

	•										

....

....

....

....

....

.... [3]

(c) Find the denary value for the following binary floating-point number. Show your working.

Mantissa													Exponent			
0	● 0	1	1	0	0	0	0	0	0	0	0		0	0	1	1

....

....

....

....

....

.... [3]

(d) **(i)** State whether the floating-point number given in **part (c)** is normalised or not normalised.

.... [1]

(ii) Justify your answer given in **part (d)(i)**.

....

.... [1]

(e) The system changes so that it now allocates 8 bits to both the mantissa and the exponent.

State **two** effects this has on the numbers that can be represented.

1 ....

....

2 ....

.... [2]

2 There are four stages in the compilation of a program written in a high-level language.

(a) Four statements and four compilation stages are shown below.

Draw a line to link each statement to the correct compilation stage.

Statement	**Compilation stage**
This stage removes any comments in the program source code.	Lexical analysis
This stage could be ignored.	Syntax analysis
This stage checks the grammar of the program source code.	Code generation
This stage produces a tokenised version of the program source code.	Optimisation

[4]

(b) Write the Reverse Polish Notation (RPN) for the following expressions.

(i) `(A + B) * (C – D)`

.... [2]

(ii) `–A / B * 4 / (C – D)`

.... [3]

(c) An interpreter is executing a program. The program uses the variables `w`, `x`, `y` and `z`.

The program contains an expression written in infix form. The interpreter converts the infix expression to RPN. The RPN expression is:

`x w z + y - *`

The interpreter evaluates this RPN expression using a stack.

The current values of the variables are:

`w = 1  x = 2  y = 3  z = 4`

(i) Show the changing contents of the stack as the interpreter evaluates the expression.

The first entry on the stack has been done for you.

2						

[4]

(ii) Convert back to its original infix form, the RPN expression:

`x w z + y - *`

....

.... [2]

(iii) Explain **one** advantage of using RPN for the evaluation of an expression.

....

....

....

.... [2]

3 A computer operating system (OS) uses paging for memory management.

In paging:

- main memory is divided into equal-size blocks, called page frames
- each process that is executed is divided into blocks of the same size, called pages
- each process has a page table that is used to manage the pages of this process

The following table is the incomplete page table for a process X.

Page	Presence flag	Page frame address	Additional data
1	1	132	
2	1	245	
3	1	232	
4	0	0	
5	1	542	
6	0	0	
135	0	0	

When a particular page of the process is currently in main memory, the Presence flag entry in the page table is set to 1.

If the page is not currently present in memory, the Presence flag is set to 0.

(a) The page frame address entry for Page 2 is 245.

State what the value 245 could represent.

.... [1]

(b) Process X executes until the next instruction is the first instruction in Page 4. Page 4 is not currently in main memory.

State a hardware device that could be storing this page.

.... [1]

(c) When an instruction to be accessed is not present in main memory, its page must be loaded into a page frame. If all page frames are currently in use, the contents of a page frame will be overwritten with this new page.

The page that is to be replaced is determined by a page replacement algorithm.

One possible algorithm is to replace the page that has been resident in main memory for the longest time.

(i) Give the additional data that would need to be stored in the page table.

....

.... [1]

(ii) Complete the table entries below to show what happens when Page 4 is swapped into main memory. Assume that Page 5 is the one to be replaced.

In the final column, give an example of the data you have identified in **part (c)(i)**.

Page	Presence flag	Page frame address	Additional data
4	....	....	....

[3]

An alternative algorithm is to replace the page that has been used least.

(iii) Give the different additional data that the page table would now need to store.

....

.... [1]

(iv) In the following table, complete the missing data to show what happens when Page 3 is swapped into main memory. Assume that Page 1 is the one to be replaced.

In the final column, give an example of the data you have identified in **part (c)(iii)**.

Page	Presence flag	Page frame address	Additional data
3	....	....	....

[3]

(d) Explain why the algorithms given in **part (c)** may not be the best choice for efficient memory management.

Longest resident ....

....

....

....

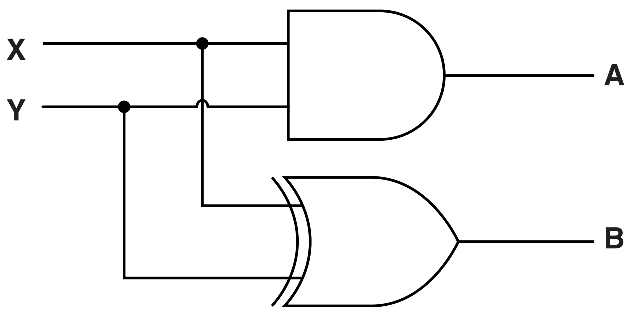
Least used ....

....

....

.... [4]

4 (a) (i) Complete the truth table for this logic circuit.



Input		**Output**	
X	**Y**	**A**	**B**
0	0		
0	1		
1	0		
1	1		

[2]

(ii) State the name given to this logic circuit.

.... [1]

(iii) Name the labels usually given to **A** and **B**.

Label **A** ....

Label **B** ....

Explain why your answers are more appropriate for the **A** and **B** labels.

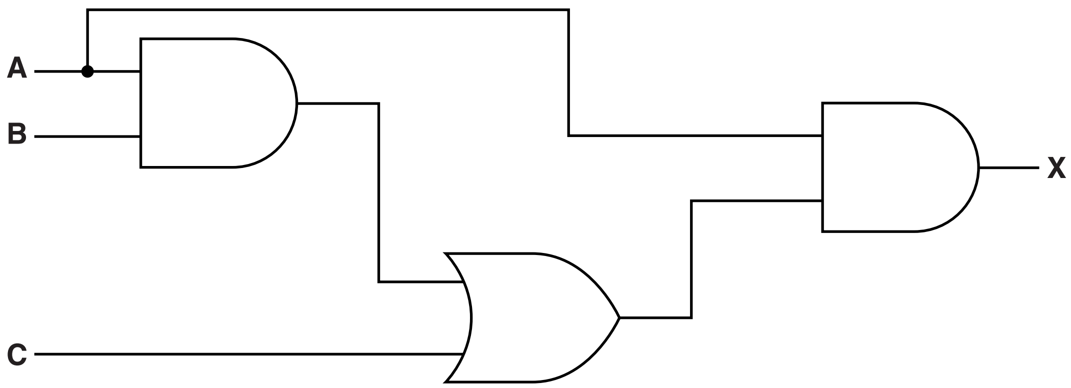
....

....

....

.... [4]

(b) **(i)** Write the Boolean expression corresponding to the following logic circuit:



.... [2]

(ii) Use Boolean algebra to simplify the expression that you gave in **part (b)(i)**.

Show your working.

....

....

....

....

....

.... [3]

5 The TCP/IP protocol suite can be viewed as a stack with four layers.

(a) **(i)** Complete the stack by inserting the names of the three missing layers.

Transport

[3]

(ii) State how each layer of the stack is implemented.

.... [1]

(b) A computer is currently running two processes:

- Process 1 is downloading a web page.
- Process 2 is downloading an email.

(i) Describe **two** tasks that the Transport layer performs to ensure that the incoming data is downloaded correctly.

1 ....

....

....

....

2 ....

....

....

.... [4]

(ii) Name a protocol that will be used by Process 1.

.... [1]

(iii) Name a protocol that will be used by Process 2.

.... [1]

6 **(a)** The table below gives descriptions of three types of malware.

Description	**Term**
Malware that attaches itself to another program.	
Malware that redirects the web browser to a fake website.	
Email that encourages the receiver to access a website and give their banking details.	

Complete the table by adding the correct terms. [3]

(b) Ben wants to send a highly confidential email to Mariah so that only she can read it. Plain text and cipher text will be used in this communication.

(i) Explain the terms plain text and cipher text.

Plain text ....

....

Cipher text ....

.... [2]

(ii) Explain how the use of asymmetric key cryptography ensures that only Mariah can read the email.

....

....

....

....

....

....

....

.... [4]

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

3020014690

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2018**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **12** printed pages.

DC (RW/CT) 151794/4

[Turn over

1 Data types can be defined in a programming language.

The data type, `StudentRecord`, is defined by the code:

```
TYPE StudentRecord
   DECLARE StudentID         : INTEGER
   DECLARE StudentFirstName  : STRING
   DECLARE StudentSurname    : STRING
   DECLARE StudentDOB        : DATE
   DECLARE StudentCourse     : ARRAY[1:10] OF STRING
ENDTYPE
```

A variable, `CollegeStudent`, is declared with the code:

```
   DECLARE CollegeStudent : StudentRecord
```

(a) Write a pseudocode statement to assign 6539 to the `StudentID` of `CollegeStudent`.

....[1]

(b) The type definition for `StudentRecord` is changed.

(i) Students can take six courses from: Computer Science, Engineering, Science, Maths, Physics, Chemistry, Music, Drama and English Language.

Rewrite **one** line from the type definition of `StudentRecord` to implement the change.

`DECLARE` ....

....

....

....[2]

(ii) The values for the field `StudentID` must be between `1` and `8000` inclusive.

Rewrite **one** line from the type definition of `StudentRecord` to implement the change.

`DECLARE` ....[1]

(c) A programmer is asked to write a program to process the assessment data for each student. Students sit one exam in every course they take.

A composite data type, `StudentAssessment`, needs to be defined with the following three fields.

- a student assessment code (a unique code of three letters and two digits)
- the marks for the six exams
- the average mark of the six exams

(i) Write **pseudocode** to define the data type `StudentAssessment`.

....

....

....

....

....

....[4]

(ii) Data about all students and their assessments are stored in a file that uses random organisation. The `StudentID` is used as the key field.

The program allows a user to enter data for a new student.

Explain how the program adds the new data to the file.

....

....

....

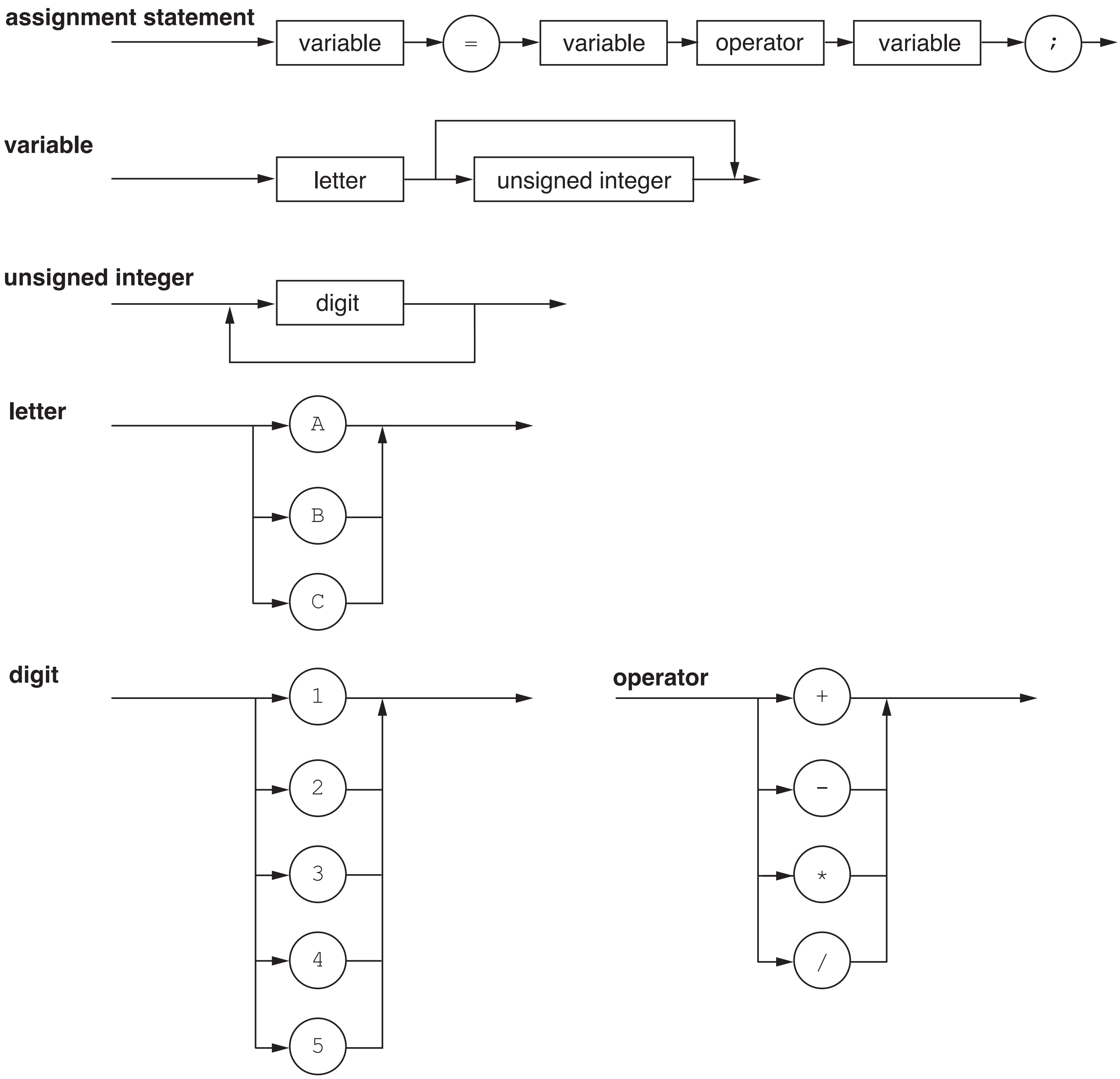
....

....

....[3]

2 The following syntax diagrams show the syntax of:

- an assignment statement
- a variable
- an unsigned integer
- a letter
- a digit
- an operator



(a) The following assignment statements are invalid.

Give the reason in each case.

(i) `A = B + 5;`

Reason ....

....[1]

(ii) `A = B - D;`

Reason ....

....[1]

(iii) `C4 = B2 - A1 + C3;`

Reason ....

....[1]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown on the opposite page.

`<assignment statement> ::=`

....

`<variable> ::=`

....

`<unsigned integer> ::=`

....

`<operator> ::=`

....
[6]

(c) The syntax of **variable** is changed to allow one or more letters followed by an unsigned integer.

Draw a syntax diagram for the new syntax of the variable.

[3]

3 In a computer system, real numbers are stored using normalised-floating point representation with:

- 8 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Calculate the normalised floating-point representation of + 21.75 in this system. Show your working.

Working ....

....

....

....

....

....

Mantissa

Exponent

[3]

(b) Find the denary value for the following binary floating-point number.

Mantissa

1	0	1	1	0	0	0	0

Exponent

1	1	1	0

Show your working.

Working ....

....

....

....

....

Answer ....

[3]

4 The TCP/IP protocol suite is used on the Internet.

(a) The table has statements about transmitting data across the Internet.

Put a tick (✓) in each row to identify whether the responsibility belongs to TCP or IP.

Responsibility	**TCP**	**IP**
Correct routing		
Host to host communication		
Communication between networks		
Retransmitting missing packets		
Reassembling packets into the correct order		

[5]

(b) Identify **two** other internet protocols. State a use for each protocol.

Protocol 1 ....

....

Use ....

....

Protocol 2 ....

....

Use ....

....
[4]

(c) State the name of the TCP/IP layer that uses IP addresses.

....[1]

(d) Emails are transmitted across the Internet using packet switching and routing tables.

(i) Give **four** items of data in an IP data packet.

1 ....

2 ....

3 ....

4 ....
[4]

(ii) Describe **two** benefits of using packet switching.

Benefit 1 ....

....

....

....

Benefit 2 ....

....

....

....
[4]

(iii) Give **two** items of data stored in a routing table.

1 ....

2 ....
[2]

5 Katarina works for a company specialising in the sale of computer parts and accessories. She works in the London office and her colleague Lucy works in the Hong Kong office. Katarina emails confidential information to Lucy so that only Lucy can read the information.

(a) Explain how public and private keys are used to ensure that only Lucy has a readable copy of the confidential information.

....

....

....

....

....

....

....

....[4]

(b) Julio is buying items from the online shop. He already has an account with the shop.

Explain how the use of Secure Socket Layer (SSL) or Transport Layer Security (TLS) helps to keep Julio's confidential information secure.

....

....

....

....

....

....[3]

(c) The manager of the company is concerned about the threat of malware.

State **three** vulnerabilities that a malware can exploit.

1 ....

....

2 ....

....

3 ....

....

[3]

6 **(a)** There are five scenarios on the left and two types of system on the right.

Draw a line to link each scenario to its correct type of system.

Scenario	**System**
Car speed display	
Aeroplane autopilot	
Rollercoaster	Control
Recording the rainfall at a weather station	Monitoring
Robot loading a part onto a conveyor belt	

[2]

(b) Mary has six fish tanks. The temperature of the water in each tank needs to be within a specific range.

Identify **three** items of hardware that Mary can add to her tanks to help maintain the temperature. Describe the purpose of each item.

Item 1 ....

Purpose ....

....

Item 2 ....

Purpose ....

....

Item 3 ....

Purpose ....

....

[6]

(c) A temperature reading is taken from each tank once per minute. The temperature reading is stored as two successive bytes. The format is shown:

Fish tank (**Byte 1**)

7	6	5	4	3	2	1	0

Temperature reading (**Byte 2**)

The fish tank number is indicated by setting one of the bits in **Byte 1** to 1. For example, fish tank number 5 is indicated by setting bit 5 to 1.

Bit 7 of **Byte 1** is a flag:

- the flag's initial value is zero
- when the reading has been processed, the flag's value is set to 1

Bit 0 of **Byte 1** is unused.

Byte 2 contains the temperature reading as a two's complement integer.

(i) After a temperature reading has been taken, the bytes contain the following data.

Byte 1

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0

Byte 2

0	0	0	0	1	0	1	1

Analyse the data contained in the two bytes.

....

....

....

....

....

....[3]

(ii) The system receives a temperature reading of −2 from fish tank number 4.

Complete the bytes to show the values for this reading after it has been processed.

7	6	5	4	3	2	1	0

Byte 1

Byte 2

[2]

(d) A hardware device to affect the temperature of each tank is on or off depending on the value of a bit in memory location `6753`.

If bit 4 is 1, then the hardware device in fish tank 4 is on.

Write **assembly language** instructions to set bit 4 of memory location `6753` to 1 without changing any other bits. Use the instruction set provided.

....

....

....

....[3]

Instruction set

Instruction		Explanation
Op code	**Operand**	
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`STO`	`<address>`	Store the contents of ACC at the given address.
`AND`	`#n`	Bitwise AND operation of the contents of ACC with the operand.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`#n`	Bitwise XOR operation of the contents of ACC with the operand.
`OR`	`#n`	Bitwise OR operation of the contents of ACC with the operand.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>. <address> can be an absolute address or a symbolic address.

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

2847275517

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2017**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

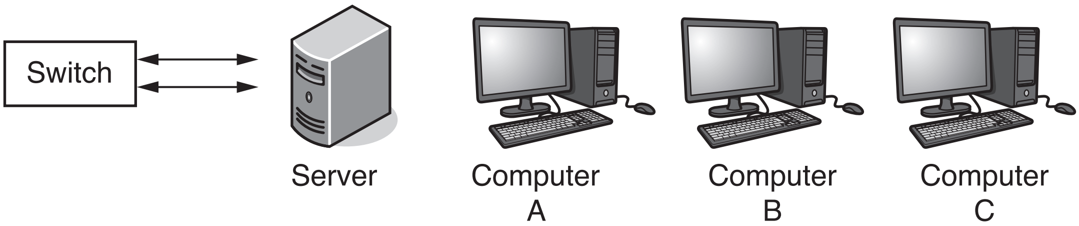
This document consists of **16** printed pages.

DC (NF/FC) 134870/3

CAMBRIDGE International Examinations

1 A Local Area Network (LAN) consists of three computers, one server and a switch. The LAN uses a star topology.

(a) Complete the following diagram to show how the computers, the server and the switch could be connected.



[1]

(b) There are four statements in the following table. For each statement, place a tick (✓) in the appropriate column to indicate whether it is true or false.

Statement	True	False
The server can send packets to Computer B and Computer C at the same time.		
The network software on each computer needs to include collision detection and avoidance.		
Computer B can read a packet sent from the server to Computer C.		
Computer A can send a packet to Computer B and at the same time the server can be sending a packet to Computer C.		

[4]

(c) The LAN shown in **part (a)** will be connected to the Internet.

(i) A router will be attached to one of the devices on the LAN.

State the device used. Give a reason for your choice.

Device ....

Reason ....

....

.... [2]

(ii) Explain why a router is required.

....

....

....

.... [2]

(iii) After the router has been connected, Computer A sends several packets to an internet web server.

Explain how the packets are transmitted from the router to the web server.

....

....

....

....

....

.... [3]

2 **(a)** The following diagram shows four descriptions and four types of computer architecture.

Draw lines to connect each description to the appropriate computer architecture.

Description	**Computer architecture**
Most parallel computer systems use this architecture.	SIMD
Widely used to process 3D graphics in video games.	MIMD
A microprocessor is used to control a washing machine.	MISD
There are a number of processing units. Each processing unit executes the same instruction but on different data.	SISD

[4]

(b) A computer has a single processor that contains four processing units.

Explain why this is **not** an example of a massively parallel computer.

....

....

....

.... [2]

(c) An application has previously executed on a single computer. The application will be transferred onto a massively parallel computer.

The program code used in the application will need to be updated to ensure that the power of the massively parallel computer is fully used.

Explain what changes will be required to the program code.

....

....

....

.... [2]

(d) Explain **one** of the hardware issues that will have to be overcome if a massively parallel computer is to function successfully.

....

....

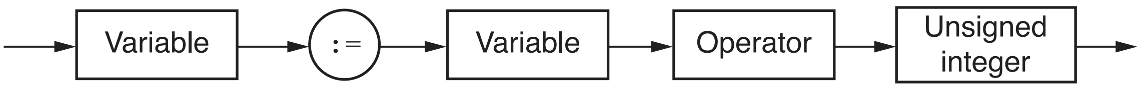
....

.... [2]

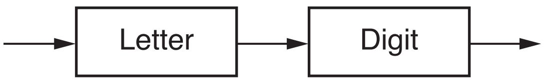
3 The following syntax diagrams for a particular programming language show the syntax of:

- an assignment statement
- a variable
- an unsigned integer
- a letter
- an operator
- a digit.

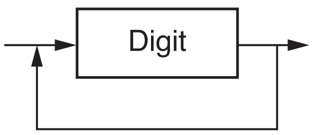
Assignment statement



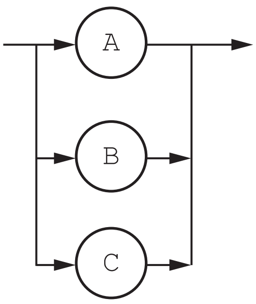
Variable



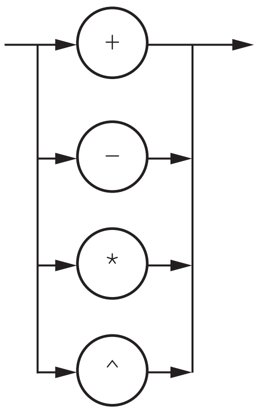
Unsigned integer



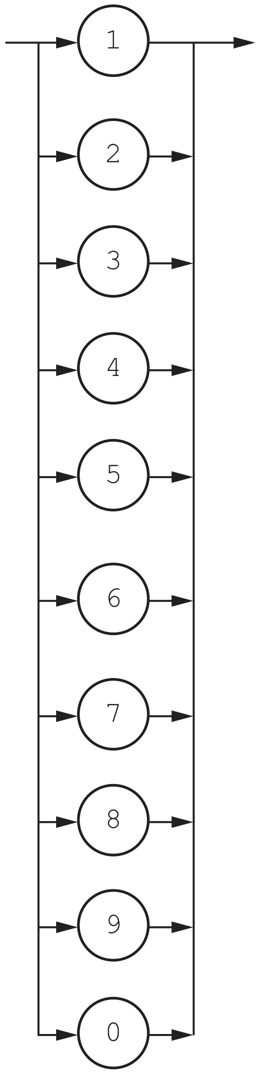
Letter



Operator



Digit



(a) The following assignment statements are invalid.

Give the reason in each case.

(i) `C2 = C3 + 123`

Reason: ....

.... [1]

(ii) `A3 := B1 - B2`

Reason: ....

.... [1]

(iii) `A32 := A2 * 7`

Reason: ....

.... [1]

(b) Complete the Backus-Naur Form (BNF) for the syntax diagrams shown.

`<digit>` has been done for you.

`<assignment_statement> ::=`

....

`<variable> ::=`

....

`<unsigned_integer> ::=`

....

`<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0`

`<letter> ::=`

....

`<operator> ::=`

....
[6]

(c) The definition of `<variable>` is changed to allow:

- one or two letters and
- zero, one or two digits.

Draw an updated version of the syntax diagram for `<variable>`.

Variable

[2]

(d) The definition of `<assignment_statement>` is altered so that its syntax has `<unsigned_integer>` replaced by `<real>`.

A real is defined to be:

- at least one digit before a decimal point
- a decimal point
- at least one digit after a decimal point.

Give the BNF for the revised `<assignment_statement>` and `<real>`.

`<assignment_statement> ::=` ....

....

`<real> ::=` ....

....
[2]

4 The Secure Socket Layer (SSL) protocol and its successor, the Transport Layer Security (TLS) protocol, are used in Internet communications between clients and servers.

(a) (i) Define the term **protocol**.

....

....

....

.... [2]

(ii) Explain the purpose of the TLS protocol.

....

....

....

....

....

.... [3]

(b) A handshake process has to take place before any exchange of data using the TLS protocol. The handshake process establishes details about how the exchange of data will occur. Digital certificates and keys are used.

The handshake process starts with:

- the client sending some communication data to the server
- the client asking the server to identify itself
- the server sending its digital certificate including the public key.

Describe, in outline, the other steps in the handshake process.

....

....

....

....

....

.... [3]

(c) Give **two** applications where it would be appropriate to use the TLS protocol.

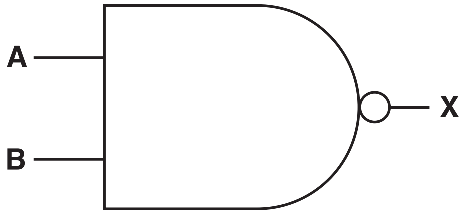
1 ....

....

2 ....

....
[2]

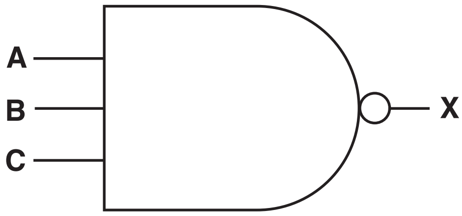
5 **(a)** **(i)** Complete the truth table for this 2-input NAND gate:



A	B	X
0	0	
0	1	
1	0	
1	1	

[1]

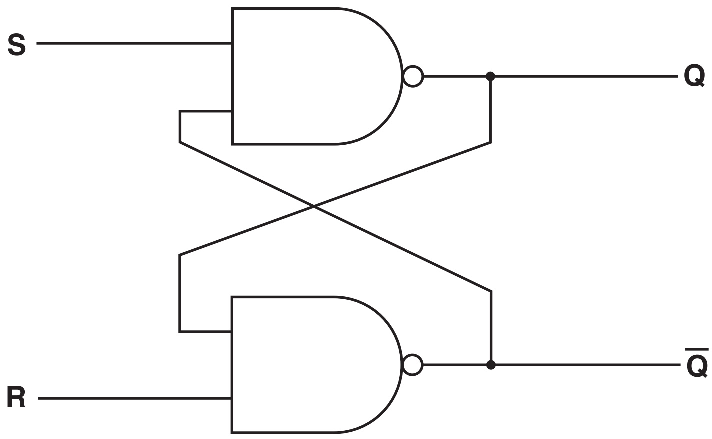
(ii) Complete the truth table for this 3-input NAND gate:



A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

[1]

(b) A SR flip-flop is constructed using two NAND gates.



(i) Complete the truth table for the SR flip-flop:

	S	R	Q	$\overline{Q}$
Initially	1	0	0	1
R changed to 1	1	1		
S changed to 0	0	1		
S changed to 1	1	1		
S and R changed to 0	0	0	1	1

[3]

(ii) The final row in the table in **part b(i)** shows that the output for both **Q** and $\overline{\mathbf{Q}}$ is 1.

Explain why this is a problem.

....

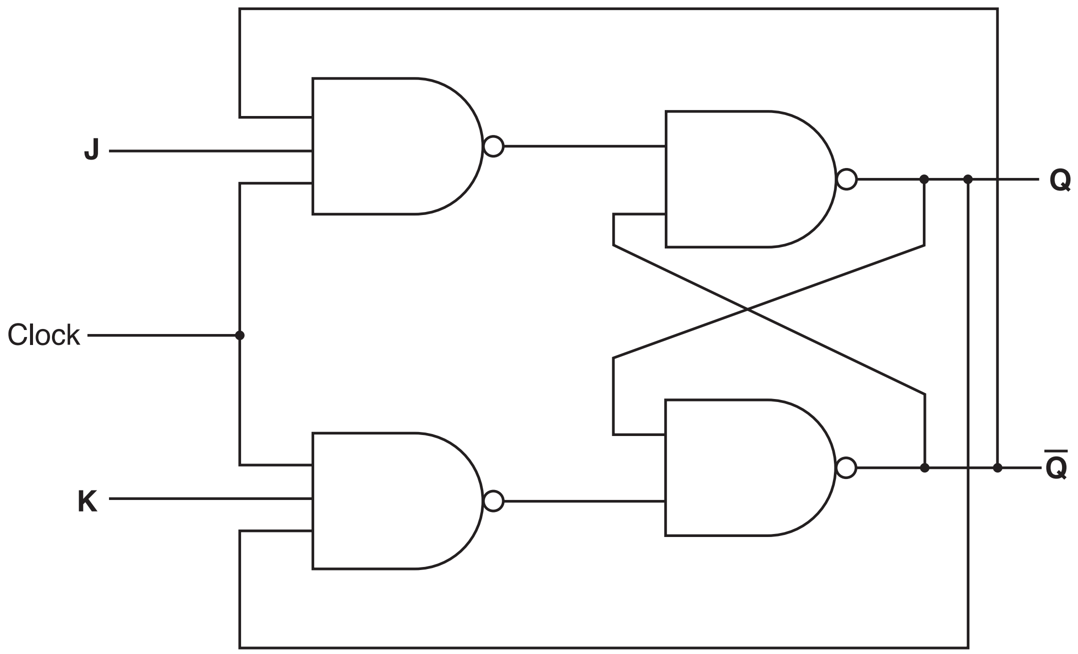
....

....

.... [2]

(c) Another type of flip-flop is the JK flip-flop.

A JK flip-flop is constructed as follows:



(i) Complete this truth table for the JK flip-flop.

J	K	Clock	Working space	Initial values Q	Initial values $\overline{Q}$	Final values Q	Final values $\overline{Q}$
0	0	1		1	0	1	0
0	0	1		0	1	0	1
0	1	1		1	0	0	1
0	1	1		0	1	0	1
1	0	1		1	0		
1	0	1		0	1		
1	1	1		1	0		
1	1	1		0	1		

[4]

(ii) Explain why the JK flip-flop is an improvement on the SR flip-flop.

....

....

....

.... [2]

(d) Explain the role of flip-flops in a computer.

....

....

....

.... [2]

6 A large warehouse stores goods that must be kept above a temperature of 15 degrees Celsius. The warehouse has six temperature sensors which are each placed at a different location in the warehouse.

A computer system is programmed to turn on appropriate heaters when one of the sensors is below the minimum temperature.

(a) **(i)** State the name given to the type of system described.

.... [1]

(ii) Justify your answer to **part (i)**.

....

.... [1]

(b) Sensors and heaters are two types of device used in this system.

State **two** other devices that are used. Justify your choice.

Device 1 ....

Justification ....

....

Device 2 ....

Justification ....

....
[4]

(c) The computer system stores the temperature readings for the six sensors in six 8-bit memory locations.

Six of the bits in an 8-bit register, LOWREG, are used to indicate whether a particular reading is below the minimum temperature. A value of 1 means the reading is below the minimum temperature.

For example:

This pattern of bits in LOWREG shows that sensor 5, sensor 4 and sensor 1 have readings below the minimum temperature.

		6	5	4	3	2	1
Not used	Not used	0	1	1	0	0	1

The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		**Explanation**
Op code	**Operand**	
LDD	<address>	Direct addressing. Load the contents of the given address to ACC.
LDR	#n	Immediate addressing. Load the number n to IX.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
STO	<address>	Store the contents of ACC at the given address.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
ADD	<address>	Add the contents of the given address to the ACC.
OR	<address>	Bitwise OR operation of the contents of ACC with the contents of address.
CMP	#n	Compare the contents of ACC with number n.
CMP	<address>	Compare the contents of ACC with the contents of <address>.
JMP	<address>	Jump to the given address.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JGE	<address>	Following a compare instruction, jump to <address> if the content of ACC is greater than or equal to the number used in the compare instruction.

Question 6(c) continues on the next page.

Part of the assembly language code for updating `LOWREG` is:

Label	Op code	Operand
LOWTEMP:		15
LOWREG:		B00000000
COUNTER:		1
START:	LDR	#0
LOOP:	LDX	8000
	CMP	LOWTEMP
	JGE	TEMPOK
	LDD	LOWREG
	OR	COUNTER
	STO	LOWREG
TEMPOK:	LDD	COUNTER
Q1:	CMP	#32
	JPE	HEATON
	ADD	COUNTER
	STO	COUNTER
	INC	IX
	JMP	LOOP
HEATON:	LDD	LOWREG

(i) The code uses six memory locations to store the temperature readings. It stores readings for sensors 1 to 6 at addresses `8000` to `8005`.

At a particular time, the memory locations store the following data.

8000	8001	8002	8003	8004	8005
17	14	15	15	16	14

Dry run the assembly language code starting at `START` and finishing when the loop has been processed twice.

LOWTEMP	LOWREG	COUNTER	ACC	IX
15	B00000000	1		

[4]

(ii) Explain why the operand of the instruction labelled `Q1` has the value 32.

....

....

....

.... [2]

(iii) The code beginning at the instruction labelled `HEATON` must make the system turn on the heaters in those areas that are below the minimum temperature.

Describe what this code will have to do.

....

....

....

....

....

.... [3]

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

4757018559

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **May/June 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

DC (LK/CB) 205788/1

1 Real numbers are stored in a computer system using floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- Two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised floating-point representation of –7.25 in this system.
Show your working.

Mantissa

Exponent

Working ....

....

....

....

....

....

....

....
[3]

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	0	1	1	1

Exponent

0	0	0	1	1	1

Working ....

....

....

....

....

Answer ....
[3]

(c) The given binary floating-point number is not normalised.

Normalise the floating-point number. Show your working.

Mantissa

0	0	0	0	0	0	0	1	1	1

Exponent

1	0	0	1	1	1

Mantissa

Exponent

Working ....

....

....

....

....

....
[3]

(d) The denary number 513 cannot be stored accurately as a normalised floating-point number in this computer system.

(i) Explain the reason for this.

....

....

....

....

....

.... [3]

(ii) Describe an alteration to the way floating-point numbers are stored to enable this number to be stored accurately using the same total number of bits.

....

....

....

.... [2]

2 **(a)** Describe the purpose of a user-defined data type.

....

....

....

....

.... [2]

(b) Define, using pseudocode, the following enumerated data types:

(i) `SchoolDay` to hold data about the days students are usually in school.

....

.... [1]

(ii) `WeekEnd` to hold data about the days that are not school days.

....

.... [1]

(c) Define, using pseudocode, the composite data type `ClubMeet`. This will hold data about club members that includes:

- first name and last name
- the two days they attend:
  - one on a school day
  - one not on a school day.

Use the enumerated types you created in **part (b)**.

....

....

....

....

....

....

....

.... [4]

3 **(a)** Draw **one** line to connect each **Operating System (OS)** term to the **most appropriate** description about it.

OS term	**Description**
Multi-tasking	Using secondary storage to simulate additional main memory
Paging	Managing the processes running on the CPU
Interrupt handling	Managing the execution of many programs that appear to run at the same time
Scheduling	Locating non-contiguous blocks of data and relocating them
Virtual memory	Transferring control to another routine when a service is required
	Reading/writing same-size blocks of data from/to secondary storage when required

[5]

(b) Explain how an interpreter executes a program without producing a complete translated version of it.

....

....

....

....

....

....

....

.... [4]

4 **(a)** **(i)** Explain why Reverse Polish Notation (RPN) is used to carry out the evaluation of expressions.

....

....

....

.... [2]

(ii) Identify, with reasons, a data structure that could be used to evaluate an expression in RPN.

....

....

....

.... [2]

(b) Write the infix expression in RPN.

```
(a − b) * (a + c) / 7
```

....

.... [1]

(c) Write the RPN expression as an infix expression.

```
a b / 4 * a b + -
```

....

.... [1]

(d) Evaluate the RPN expression:

```
a b + c d / /
```

where `a = 17, b = 3, c = 48 and d = 12`.

Show your working.

....

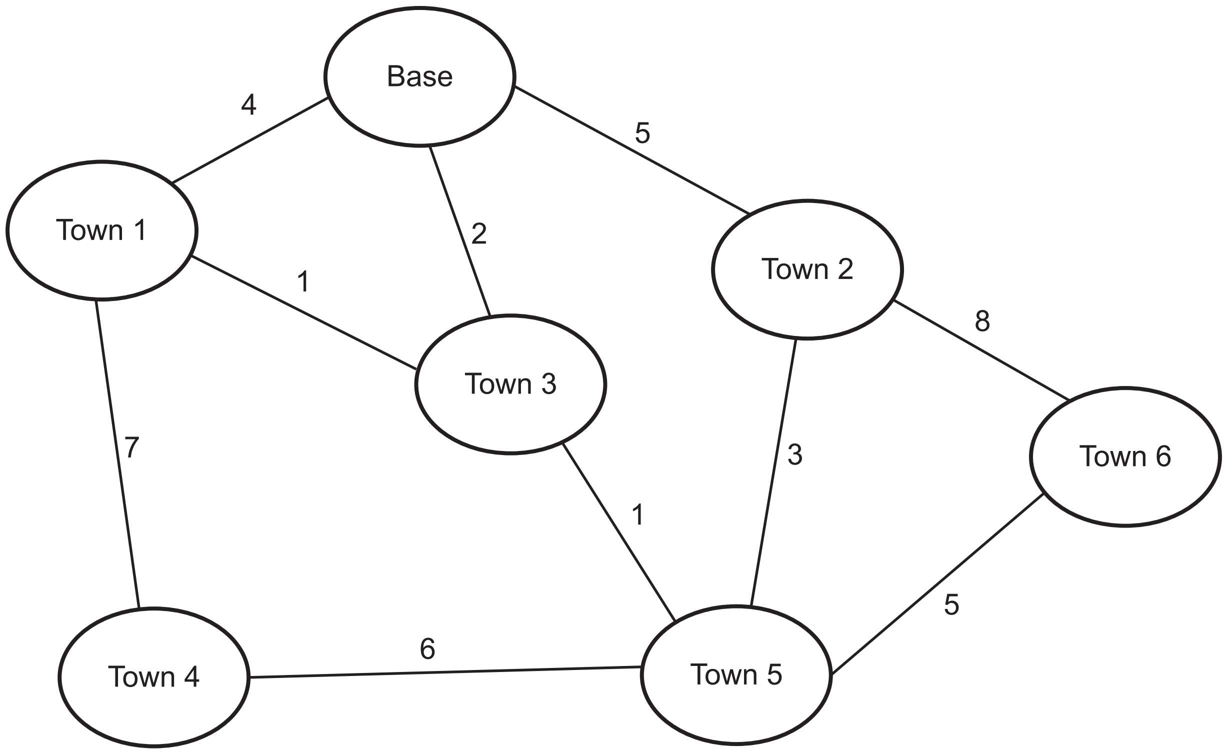
....

....

.... [2]

5 (a) Calculate the shortest distance between the base and each of the other towns in the diagram using Dijkstra's algorithm.

Show your working **and** write your answers in the table provided.



Working ....

....

....

....

....

....

....

....

Answers

Town 1	Town 2	Town 3	Town 4	Town 5	Town 6

[5]

(b) Explain the use of graphs to aid Artificial Intelligence (AI).

....

....

....

....

....

.... [3]

6 Give **two** benefits **and two** drawbacks of packet switching.

Benefit 1 ....

....

Benefit 2 ....

....

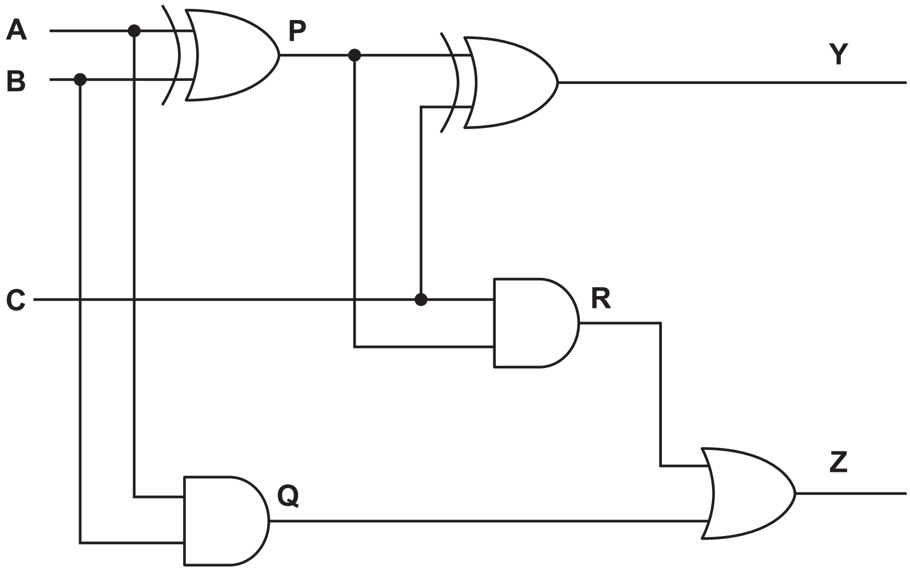
Drawback 1 ....

....

Drawback 2 ....

....
[4]

7 The diagram shows a logic circuit.



(a) Complete the truth table for the given logic circuit. Show your working.

Inputs			Working space			Outputs	
A	**B**	**C**	**P**	**Q**	**R**	**Y**	**Z**
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

[3]

(b) State the name of the logic circuit.

.... [1]

(c) Write the Boolean expressions for the two outputs **Y** and **Z** in the truth table as sum-of-products **and** state the purpose of each output.

Y = ...

Purpose ....

Z = ...

Purpose ....

[4]

[Turn over

8 **(a)** State **two** factors that may affect the performance of a sorting algorithm.

....

....

....

.... [2]

(b) The given algorithm is a simple bubble sort that arranges a set of scores stored in a one-dimensional array into **descending** order, and orders the corresponding students' names stored into a two-dimensional array in the same order as the scores. All the arrays are indexed from 1.

The contents of both arrays after sorting are shown.

	Score
1	98
2	97
...	
248	5
249	3

	Name	
	1	**2**
1	Smithfield	Tom
2	Johnson	Jane
...		
248	Peters	Jade
249	Allen	John

```
YearSize ← 249
Flag ← TRUE
WHILE Flag = TRUE
    Flag ← FALSE
    FOR Student ← 1 TO YearSize - 1
        IF Score[Student] < Score[Student + 1] THEN
           Temp1 ← Score[Student]
           Temp2 ← Name[Student,1]
           Temp3 ← Name[Student,2]
           Score[Student] ← Score[Student + 1]
           Name[Student,1] ← Name[Student + 1,1]
           Name[Student,2] ← Name[Student + 1,2]
           Score[Student + 1] ← Temp1
           Name[Student + 1,1] ← Temp2
           Name[Student + 1,2] ← Temp3
           Flag ← TRUE
        ENDIF
    NEXT Student
ENDWHILE
```

Write an algorithm, using pseudocode, that will perform the same task using an insertion sort.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

9 **(a)** Describe what is meant by **an imperative (procedural)** programming language.

....

....

....

.... [2]

(b) Describe what is meant by **a declarative** programming language.

....

....

....

.... [2]

(c) Identify the programming paradigm for each of these program code examples.

Program code example	**Programming paradigm**
`male(john).` `female(ethel).` `parent(john, ethel).`	
`FOR Counter = 1 TO 20` `    X = X * Counter` `NEXT Counter`	
`Start: LDD Counter` `       INC ACC` `       STO Counter`	
`public class Vehicle` `{` `    private speed;` `    public Vehicle()` `    {` `        speed = 0;` `    }` `}`	

[4]

Cambridge International AS & A Level

Cambridge Assessment International Education
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME	

CENTRE NUMBER		CANDIDATE NUMBER	

4657474381

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 2019**

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (SC/CB) 168338/4

Cambridge Assessment International Education

1 Each student at CIE University needs a printing account to print documents from university computers.

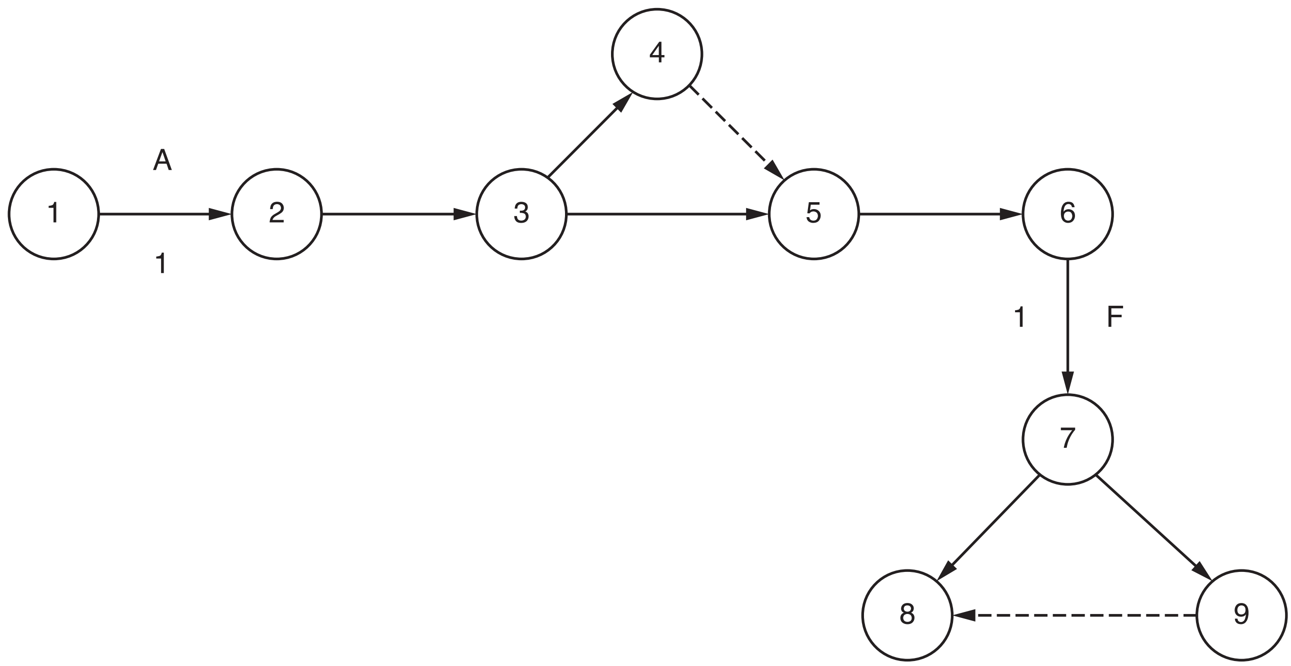
The university is developing software to manage each student's printing account and the printing process.

(a) Developing the software will include the following activities.

Activity	Description	Time in weeks	Predecessor
A	Identify requirements	1	-
B	Produce design	3	A
C	Write code	10	B
D	Test modules	7	B
E	Final system black-box testing	3	C, D
F	Install software	1	E
G	Acceptance testing	2	F
H	Create user documentation	2	F

(i) Add the correct activities and times to the following Program Evaluation Review Technique (PERT) chart for the software development.

Two activities and times have been done for you.



[6]

(ii) State what is meant by the **critical path** in a PERT chart.

....

.... [1]

(iii) Identify **and** describe a project planning technique, other than a PERT chart.

....

....

....

.... [2]

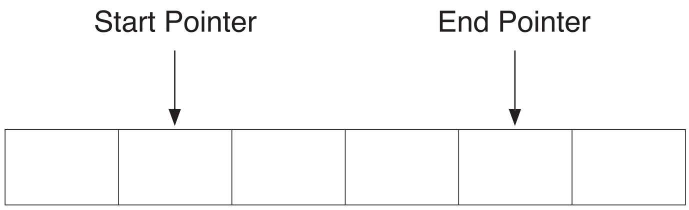
(b) When a student prints a document, a print job is created. The print job is sent to a print server.

The print server uses a queue to hold each print job waiting to be printed.

(i) The queue is circular and has six spaces to hold jobs.

The queue currently holds four jobs waiting to be printed. The jobs have arrived in the order A, B, D, C.

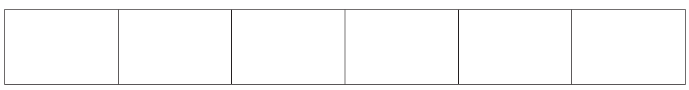
Complete the diagram to show the current contents of the queue.



[1]

(ii) Print jobs A and B are now complete. Four more print jobs have arrived in the order E, F, G, H.

Complete the diagram to show the current contents and pointers for the queue.



[3]

(iii) State what would happen if another print job is added to the queue in the status in **part (b)(ii)**.

....

.... [1]

(iv) The queue is stored as an array, `Queue`, with six elements. The following algorithm removes a print job from the queue and returns it.

Complete the following **pseudocode** for the function `Remove`.

```
FUNCTION Remove RETURNS STRING
   DECLARE PrintJob : STRING
   IF .................................................................... = EndPointer
      THEN
         RETURN "Empty"
      ELSE
         PrintJob ← Queue[....................................................................]
         IF StartPointer = ....................................................................
            THEN
               StartPointer ← ...............................................................
            ELSE
               StartPointer ← StartPointer + 1
         ENDIF
         RETURN PrintJob
   ENDIF
ENDFUNCTION
```

[4]

(v) Explain why the circular queue could not be implemented as a stack.

....

....

....

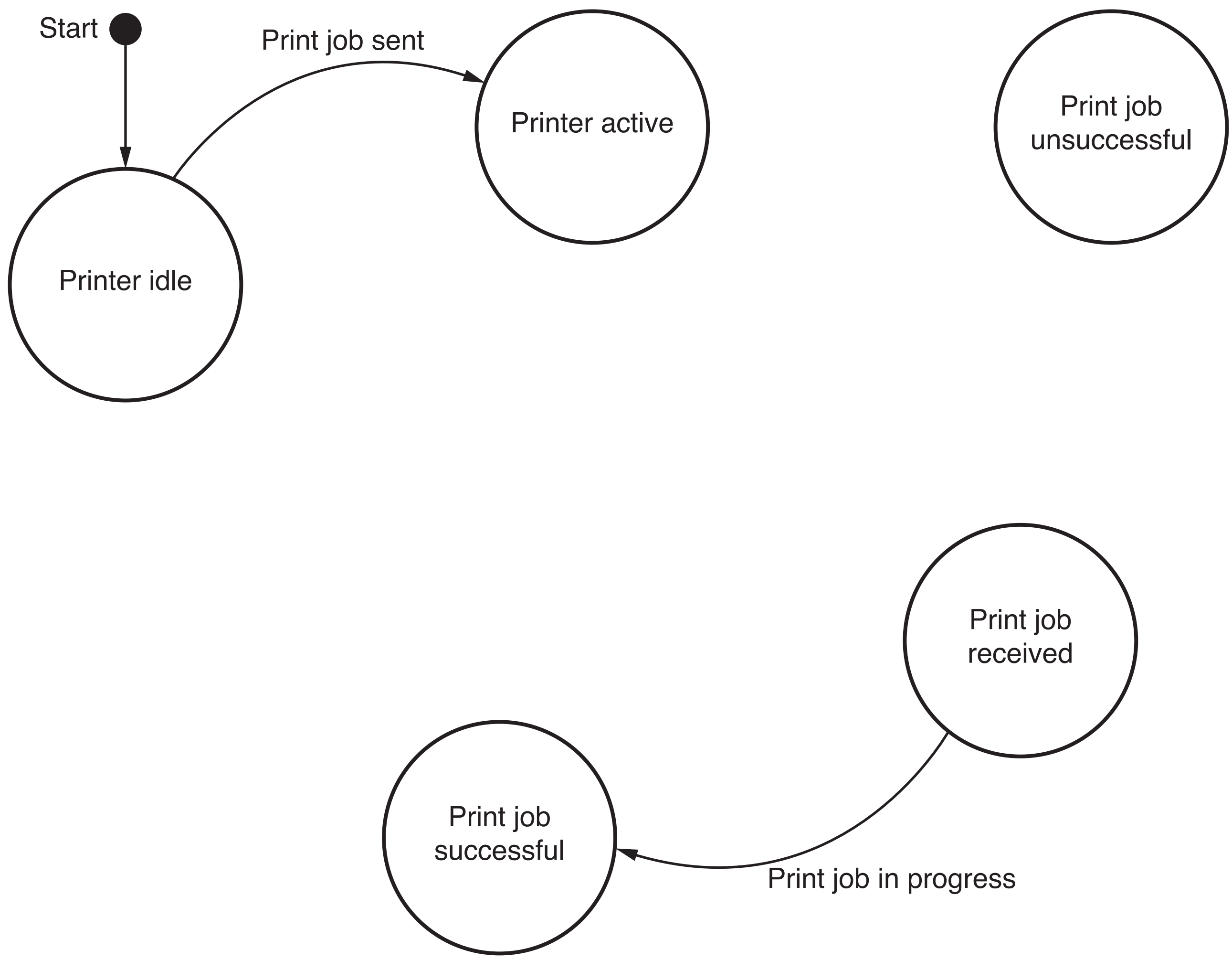
.... [2]

(c) The university wants to analyse how a printer and a print server deal with the print jobs.

The following table shows the transitions from one state to another for the process.

Current state	Event	Next state
Printer idle	Print job sent	Printer active
Printer active	Print job added to queue	Print job received
Print job received	Print job in progress	Print job successful
Print job received	Print job in progress	Print job unsuccessful
Print job successful	Check print queue	Printer active
Print job unsuccessful	Error message displayed	Printer active
Printer active	Timeout	Printer idle

Complete the state-transition diagram for the table.



[5]

(d) The university wants to assess troubleshooting issues with a printer. It wants to use a decision table to do this.

The troubleshooting actions are:

- check the connection from computer to printer, if the error light is flashing **and** the document has not been printed
- check the ink status, if the quality is poor
- check whether there is a paper jam, if the error light is flashing **and** the document has not been printed
- check the paper size selected, if the paper size is incorrect.

(i) Describe the purpose of a decision table.

....

....

....

.... [2]

(ii) Complete the rules for the actions in the following decision table.

		Rules							
Conditions	Document printed but the quality is poor	Y	Y	Y	Y	N	N	N	N
	Error light is flashing on printer	Y	Y	N	N	Y	Y	N	N
	Document printed but paper size is incorrect	Y	N	Y	N	Y	N	Y	N
Actions	Check connection from computer to printer								
	Check ink status								
	Check if there is a paper jam								
	Check the paper size selected								

[4]

(iii) Simplify your solution by removing redundancies.

		Rules							
Conditions	Document printed but the quality is poor								
	Error light is flashing on printer								
	Document printed but paper size is incorrect								
Actions	Check connection from computer to printer								
	Check ink status								
	Check if there is a paper jam								
	Check the paper size selected								

[5]

(e) There are 1000 students at the university. They will each require a printing account.

Students need to buy printing credits that will be added to their account. Each page printed uses one printing credit.

The university needs software to keep track of the number of printing credits each student has in their account. The university has decided to implement the software using object-oriented programming (OOP).

The following diagram shows the design for the class `PrintAccount`. This includes the attributes and methods.

```
                        PrintAccount
FirstName  :  STRING    //  parameter sent to Constructor()
LastName   :  STRING    //  parameter sent to Constructor()
PrintID    :  STRING    //  parameter sent to Constructor()
Credits    :  INTEGER   //  initialised to 50

Constructor()     //  instantiates an object of the PrintAccount class,
                  //  and assigns initial values to the attributes
GetName()         //  returns FirstName and LastName concatenated
                  //  with a space between them
GetPrintID()      //  returns PrintID
SetFirstName()    //  sets the FirstName for a student
SetLastName()     //  sets the LastName for a student
SetPrintID()      //  sets the PrintID for a student
AddCredits()      //  increases the number of credits for a student
RemoveCredits()   //  removes credits from a student account
```

(i) Write **program code** for the `Constructor()` method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

.... [4]

(ii) Write **program code** for the `SetFirstName()` method.

Programming language ....

Program code

....

....

....

....

.... [2]

(iii) Write **program code** for the `GetName()` method.

Programming language ....

Program code

....

....

....

....

....

.... [2]

(iv) The method `AddCredits()` calculates the number of printing credits a student buys and adds the printing credits to the student's account.

- Credits cost $1 for 25 credits.
- If a student buys $20 or more of credits in a single payment, they receive an extra 50 credits.
- If a student buys between $10 and $19 (inclusive) of credits in a single payment, they receive an extra 25 credits.

Payment from a student is stored in the variable `MoneyInput`. This is passed as a parameter.

Write **program code** for `AddCredits()`. Use constants for the values that do not change.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

(v) A global array, `StudentAccounts`, stores 1000 instances of `PrintAccount`.

Write **pseudocode** to declare the array `StudentAccounts`.

....

.... [2]

(vi) The main program has a procedure, `CreateID()`, that:

- takes the first name and last name as parameters
- creates `PrintID` that is a concatenation of:
  - the first three letters of the first name in lower case
  - the first three letters of the last name in lower case
  - the character '1'
    for example, the name Bill Smith would produce `"bilsmi1"`
- checks if the `PrintID` created already exists in the global array `StudentAccounts`:
  - If `PrintID` does not exist, it creates an instance of `PrintAccount` in the next free index in `StudentAccounts`.
  - If `PrintID` does exist, the number is incremented until a unique ID is created, for example, `"bilsmi2"`. It then creates an instance of `PrintAccount` in the next free index in `StudentAccounts`.

The global variable `NumberStudents` stores the number of print accounts that have currently been created.

Write **program code** for the procedure `CreateID()`. Do not write the procedure header.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

2 The following table shows part of the instruction set for a processor, which has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		**Explanation**
Op code	**Operand**	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents of this calculated address to ACC.
`LDR`	`#n`	Immediate addressing. Load the number n to IX.
`STO`	`<address>`	Store the contents of ACC at the given address.
`STX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents from ACC to this calculated address.
`ADD`	`<address>`	Add the contents of the given address to the ACC.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`JMP`	`<address>`	Jump to the given address.
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`CMP`	`#n`	Compare the contents of ACC with number n.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`AND`	`#n`	Bitwise AND operation of the contents of ACC with the operand.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`#n`	Bitwise XOR operation of the contents of ACC with the operand.
`XOR`	`<address>`	Bitwise XOR operation of the contents of ACC with the contents of <address>.
`OR`	`#n`	Bitwise OR operation of the contents of ACC with the operand.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>. <address> can be an absolute address or a symbolic address.
`LSL`	`#n`	Bits in ACC are shifted n places to the left. Zeros are introduced on the right hand end.
`LSR`	`#n`	Bits in ACC are shifted n places to the right. Zeros are introduced on the left hand end.
`IN`		Key in a character and store its ASCII value in ACC.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

A programmer writes a program that multiplies two numbers together and outputs the result. The numbers are stored as `NUMONE` and `NUMTWO`.

The programmer has started to write the program in the following table. The comment column contains explanations for some of the missing program instructions and data.

Complete the program using the given instruction set.

Label	Op code	Operand	Comment
LOOP:			// load the value from ANSWER
			// add the value from NUMONE
			// load the value from COUNT
			// increment the Accumulator
			// is NUMTWO = COUNT ?
			// if false, jump to LOOP
			// load the value from ANSWER
			// output ANSWER to the screen
			// end of program
NUMONE:	2		
NUMTWO:	4		
COUNT:	0		
ANSWER:	0		

[9]

3 Software may not perform as expected. One reason for this is that a syntax error exists in the code.

Identify **three other** reasons why software may not perform as expected.

1 ....

....

2 ....

....

3 ....

....

[3]

4 The following table contains definitions related to testing terminology.

Complete the table with the correct testing term to match the definition.

Definition	**Term**
Software is tested by an in-house team of dedicated testers.	....
Software is tested by the customer before it is signed off.	....
Software is tested by a small selection of users before general release.	....

[3]

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

9438678873

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2015**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **12** printed pages.

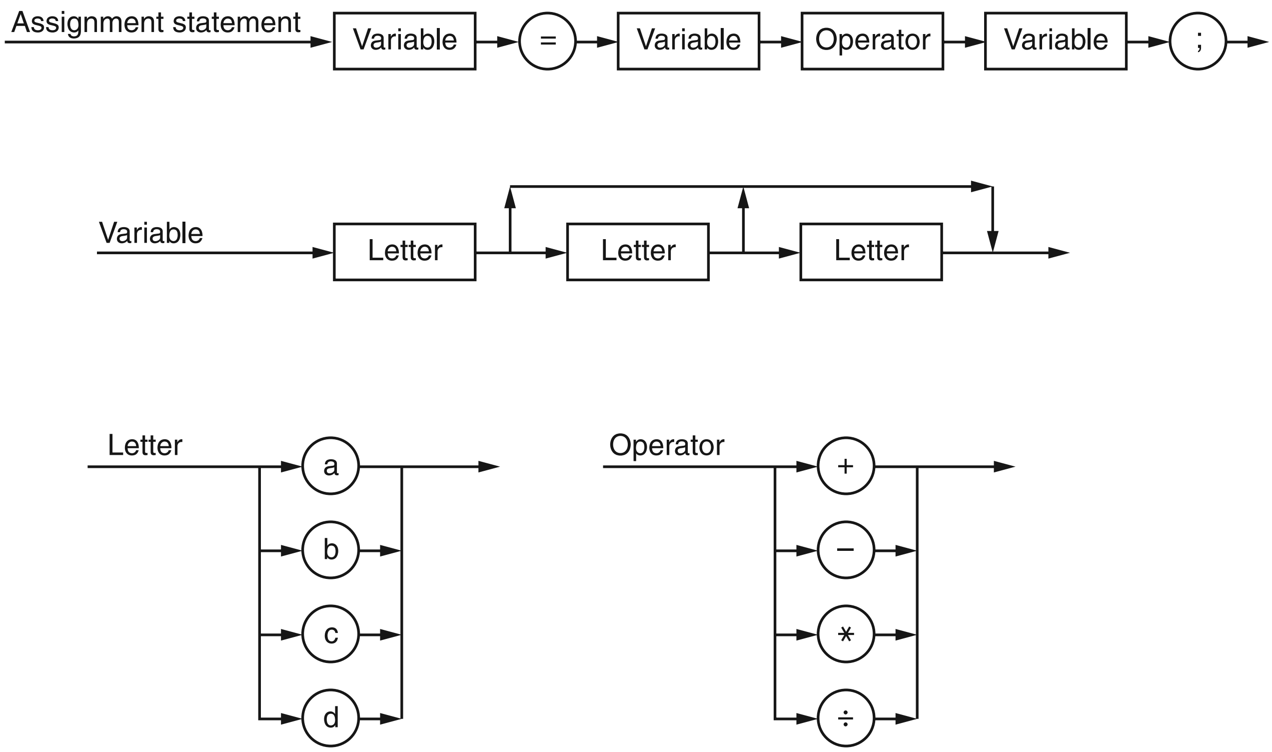
DC (ST/SW) 95541/2

CAMBRIDGE
International Examinations

[Turn over

1 The following syntax diagrams, for a particular programming language, show the syntax of:

- an assignment statement
- a variable
- a letter
- an operator



(a) The following assignment statements are invalid.

Give the reason in each case.

(i) `a = b + c`

Reason ....

....[1]

(ii) `a = b - 2;`

Reason ....

....[1]

(iii) `a = dd * cce;`

Reason ....

....[1]

(b) Write the Backus-Naur Form (BNF) for the syntax diagrams shown on the opposite page.

`<assignmentstatement> ::=`

....

`<variable> ::=`

....

`<letter> ::=`

....

`<operator> ::=`

....[6]

(c) Rewrite the BNF rule for a variable so that it can be any number of letters.

`<variable> ::=`

....[2]

(d) Programmers working for a software development company use both interpreters and compilers.

(i) The programmers prefer to debug their programs using an interpreter.

Give **one** possible reason why.

....

....[1]

(ii) The company sells compiled versions of its programs.

Give a reason why this helps to protect the security of the source code.

....

....[1]

2 The incomplete table below shows descriptions and terms relating to malware.

(a) Complete the table with appropriate descriptions and terms.

	Description	**Term**
A	Unsolicited emails containing advertising material sent to a distribution list.	....
B	A standalone piece of malicious software that can reproduce itself automatically.	....
C	....	Pharming
D	....	Phishing

[4]

(b) For one of the terms, describe:

- a problem that might arise for a user
- a possible solution to the problem

Choose between the terms:

A / B (circle your choice)

Problem ....

....

Solution ....

....[2]

(c) Explain the following terms:

Encryption ....

....

....

....

Public key ....

....

....

....[2]

(d) A user downloads software from the Internet.

(i) State what should be part of the download to provide proof that the software is authentic.

....[1]

(ii) Describe the process for ensuring that the software is both authentic and has not been altered.

....

....

....

....

....

....

....

....[4]

3 **(a)** A particular programming language allows the programmer to define their own data types.

`ThisDate` is an example of a user-defined structured data type.

```
TYPE ThisDate
   DECLARE ThisDay    :  (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                         13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
                         24, 25, 26, 27, 28, 29, 30, 31)
   DECLARE ThisMonth  :  (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug,
                         Sep, Oct, Nov, Dec)
   DECLARE ThisYear   :  INTEGER
ENDTYPE
```

A variable of this new type is declared as follows:

```
DECLARE DateOfBirth : ThisDate
```

(i) Name the non-composite data type used in the `ThisDay` and `ThisMonth` declarations.

....[1]

(ii) Name the data type of `ThisDate`.

....[1]

(iii) The month value of `DateOfBirth` needs to be assigned to the variable `MyMonthOfBirth`.

Write the required statement.

....[1]

(b) Annual rainfall data from a number of locations are to be processed in a program.

The following data are to be stored:

- location name
- height above sea level (to the nearest metre)
- total rainfall for each month of the year (centimetres to 1 decimal place)

A user-defined, composite data type is needed. The programmer chooses `LocationRainfall` as the name of this data type.

A variable of this type can be used to store all the data for one particular location.

(i) Write the definition for the data type `LocationRainfall`.

....

....

....

....

....

....

....[5]

(ii) The programmer decides to store all the data in a file. Initially, data from 27 locations will be stored. More rainfall locations will be added over time and will never exceed 100.

The programmer has to choose between two types of file organisation. The two types are serial and sequential.

Give **two** reasons for choosing serial file organisation.

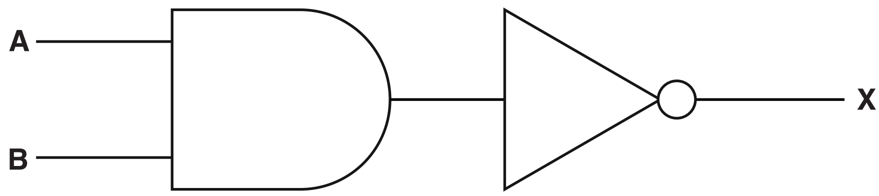
....

....

....

....[2]

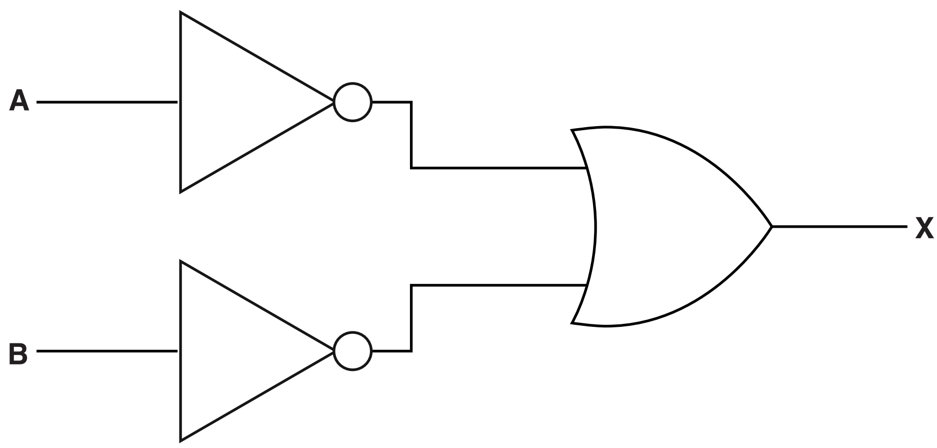
4 **(a)** **(i)** Complete the truth table for this logic circuit:



A	B	Working space	X
0	0		
0	1		
1	0		
1	1		

[1]

(ii) Complete the truth table for this logic circuit:



A	B	Working space	X
0	0		
0	1		
1	0		
1	1		

[1]

(b) A student decides to write an equation for **X** to represent the full behaviour of each logic circuit.

(i) Write the Boolean expression that will complete the required equation for **X** for each circuit:

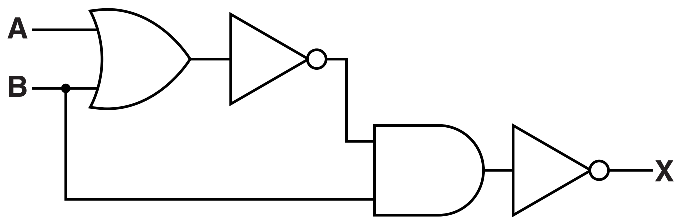
Circuit 1: **X** = ...

Circuit 2: **X** = ...[2]

(ii) Write the De Morgan's Law which is shown by your answers to **part (a)** and **part (b)(i)**.

....[1]

(c) Write the Boolean algebraic expression corresponding to the following logic circuit:



....[3]

(d) Using De Morgan's laws and Boolean algebra, simplify your answer to **part (c)**.

Show all your working.

....

....

....

....

....

....[3]

5 A gardener grows vegetables in a greenhouse. For the vegetables to grow well, the temperature needs to always be within a particular range.

The gardener is not sure about the actual temperatures in the greenhouse during the growing season. The gardener installs some equipment. This records the temperature every hour during the growing season.

(a) Name the type of system described.

....[1]

(b) Identify **three** items of hardware that would be needed to acquire and record the temperature data. Justify your choice for each.

Item 1 ....

Justification ....

....

Item 2 ....

Justification ....

....

Item 3 ....

Justification ....

....[6]

(c) The equipment records temperatures in the greenhouse. It does this for seven locations.

Each recording is stored as two successive bytes. The format is shown below:

Greenhouse location

7	6	5	4	3	2	1	0

Byte 1

Temperature reading

Byte 2

The location is indicated by the setting of one of the seven bits in byte 1. For example, location 4 is indicated by setting bit 4.

Bit 0 of byte 1 acts as a flag:

- the initial value is zero
- when the reading has been processed it is set to 1

Byte 2 contains the temperature reading (two's complement integer).

(i) Interpret the data in byte 1 shown below:

7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	1

Byte 1

0	0	0	1	1	0	0	0

Byte 2

....

....

....

....[2]

(ii) The system receives a temperature reading of –5 degrees from sensor 6.

Complete the boxes below to show the two bytes for this recording. The reading has not yet been processed.

7	6	5	4	3	2	1	0

Byte 1

Byte 2

[2]

(d) **(i)** The accumulator is loaded with the value of byte 1 from location 106.

Write the assembly language instruction to check whether the reading in byte 2 came from location 4.

```
LDD 106            // data loaded from address 106
```

....[4]

(ii) Write the assembly language instruction to set the flag (bit 0) of the byte contained in the accumulator to 1.

....[2]

6 **(a)** Four descriptions and three protocols are shown below.

Draw a line to connect each description to the appropriate protocol.

Description	**Protocol used**
email client downloads an email from an email server	HTTP
email is transferred from one email server to another email server	POP3
email client sends email to email server	SMTP
browser sends a request for a web page to a web server	

[4]

(b) Downloading a file can use the client-server model. Alternatively, a file can be downloaded using the BitTorrent protocol.

Name the model used.

....[1]

(c) For the BitTorrent protocol, explain the function of each of the following:

(i) Tracker ....

....

....[2]

(ii) Seed ....

....

....[2]

(iii) Swarm ....

....

....[2]

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME	
CENTRE NUMBER	
CANDIDATE NUMBER	

5664833112

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 2021**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **20** pages. Any blank pages are indicated.

DC (PQ/CT) 212836/4

1 An array, `NumberArray`, stores 100 integer values. The array needs to be sorted into ascending numerical order.

(a) Describe how an insertion sort will sort the data in `NumberArray`.

....

....

....

....

....

....

....

....

....

.... [4]

(b) Another type of sorting algorithm is a bubble sort.

The procedure `Bubble()` takes an array as a parameter. It performs a bubble sort on the array. The sorting algorithm stops as soon as all the elements are in ascending order.

Complete the procedure `Bubble()`.

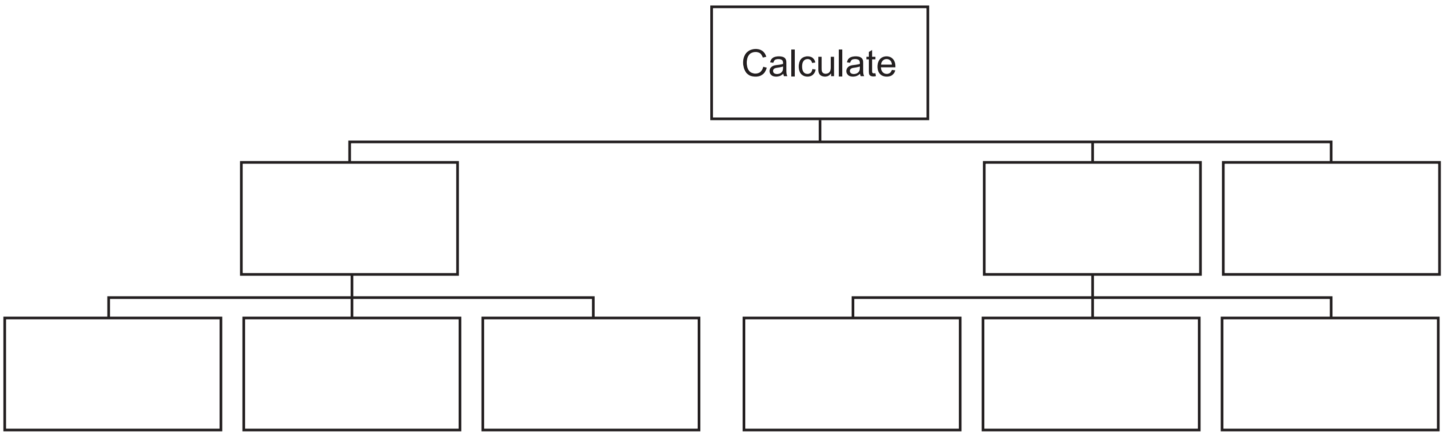
```
PROCEDURE Bubble(BYREF NumberArray : ARRAY[0 : 99] OF INTEGER)

   DECLARE Outer : INTEGER

   DECLARE Swap : BOOLEAN

   DECLARE Inner : INTEGER

   DECLARE Temp : INTEGER

   Outer ← LENGTH(NumberArray) - 1

   REPEAT

      Inner ← ............................................

      Swap ← FALSE

      REPEAT

         IF NumberArray[Inner] > NumberArray[Inner + 1]

            THEN

               Temp ← NumberArray[Inner]

               NumberArray[Inner] ← NumberArray[Inner + 1]

               NumberArray[Inner + 1] ← Temp

               Swap ← ............................................

         ENDIF

         Inner ← Inner + 1

      UNTIL Inner = ............................................

      Outer ← Outer - 1

   UNTIL Swap = ............................................ OR Outer = ............................................

ENDPROCEDURE
```

[5]

2 Complete the JSP structure diagram for the following pseudocode procedure.

```
PROCEDURE Calculate()
   INPUT Number1
   INPUT Number2
   INPUT Command
   IF Command = 1
      THEN
         Value ← Function1(Number1, Number2)
      ELSE
         IF Command = 2
            THEN
               Value ← Function2(Number1, Number2)
            ELSE
               Value ← Function3(Number1, Number2)
         ENDIF
   ENDIF
   OUTPUT Value
ENDPROCEDURE
```

JSP structure diagram



[4]

3 A user has to choose a new password to create an account. It is recommended that the password has at least two of the following elements:

- upper-case letter
- numeric character
- symbol.

The system outputs:

- `"Strong"` if there are at least two of the elements
- `"Medium"` if there is only one of the elements
- `"Weak"` if there are none of the elements.

Complete the following decision table for the password system described.

		Rules							
Conditions	One or more upper-case letters	N	Y	N	Y	N	Y	N	Y
	One or more numeric characters	N	N	Y	Y	N	N	Y	Y
	One or more symbols	N	N	N	N	Y	Y	Y	Y
Actions	Strong								
	Medium								
	Weak								

[3]

4 Each node of a binary tree is a record. Each record has a left pointer, an integer data value between 0 and 100 inclusive, and a right pointer.

For example:

Item	**Example data**
`LeftPointer`	`2`
`Data`	`34`
`RightPointer`	`3`

(a) Write **pseudocode** to declare the record with the identifier `Node`.

....

....

....

....

....

....

.... [2]

(b) Write **pseudocode** to declare a new node, `Node100`, **and** assign 100 to its data value, 1 to the left pointer and 4 to the right pointer.

....

....

....

....

....

....

.... [3]

(c) The ordered binary tree is stored as a 1D global array named `BinaryTree` of type `Node`.

`RootNode` and `FreePointer` are declared as global variables.

A null pointer is represented by `−1`.

The current state of the binary tree is shown in the following table:

RootNode	0
FreePointer	6

Index	**LeftPointer**	**Data**	**RightPointer**
[0]	1	23	3
[1]	-1	5	2
[2]	-1	8	4
[3]	5	100	-1
[4]	-1	9	-1
[5]	-1	88	-1
[6]	-1	null	-1
[7]	-1	null	-1

(i) State the purpose of the free pointer.

....

.... [1]

(ii) Identify an appropriate integer value to represent null data.

.... [1]

(iii) Draw the current state of the binary tree.

[2]

(iv) The procedure `AddData()`:

- takes the node to be added to the tree as a parameter
- finds the location for the node to be stored
- stores the node in the next free array index
- stores `-1` in the new node's `LeftPointer` and `RightPointer`
- updates the pointers in the other nodes
- updates `FreePointer`.

Complete the pseudocode for the procedure `AddData()`.

```
PROCEDURE AddData(NewNode)
    BinaryTree[FreePointer] ← ............................................
    BinaryTree[FreePointer].LeftPointer ← -1
    BinaryTree[FreePointer].RightPointer ← -1
    DECLARE PositionFound : BOOLEAN
    DECLARE PointerCounter : INTEGER
    PositionFound ← ............................................
    PointerCounter ← RootNode
    WHILE NOT ............................................
        IF ............................................ < BinaryTree[PointerCounter].Data
            THEN
                IF BinaryTree[PointerCounter].LeftPointer = -1
                    THEN
                        BinaryTree[PointerCounter].LeftPointer ← FreePointer
                        PositionFound ← TRUE
                    ELSE
                        PointerCounter ← BinaryTree[PointerCounter].LeftPointer
                ENDIF
            ELSE
                IF BinaryTree[PointerCounter].RightPointer = -1
                    THEN
                        BinaryTree[PointerCounter].RightPointer ← FreePointer
                        PositionFound ← TRUE
                    ELSE
                        PointerCounter ← BinaryTree[PointerCounter].RightPointer
                ENDIF
        ENDIF
    ENDWHILE
    FreePointer ← FreePointer ............................................
ENDPROCEDURE
```

[5]

5 Study the following recursive pseudocode algorithm.

```
FUNCTION Recursive(Num1, Num2 : INTEGER) RETURNS INTEGER
   IF Num1 > Num2
      THEN
         RETURN 10
      ELSE
         IF Num1 = Num2
            THEN
               RETURN Num1
            ELSE
               RETURN Num1 + Recursive(Num1 * 2, Num2)
         ENDIF
   ENDIF
ENDFUNCTION
```

(a) The function is called as follows:

```
Recursive(1, 15)
```

Dry run the function and complete the trace table. Give the final return value.

Trace table:

Function call	Num1	Num2	Return value

Final return value ....

Working ....

....

....

....

....

[4]

(b) Rewrite the function `Recursive()` in **pseudocode**, using an **iterative algorithm**.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [7]

6 Details of errors generated in a program are stored in a stack.

Details of each error are stored in a record structure, `Error`.

(a) State which error will be the first retrieved from the stack.

....

.... [1]

(b) The stack is implemented as a 1D array with the identifier `ErrorArray`.

The pointer `LastItem` stores the position of the last error in the array.

(i) The function, `AddItemToStack`, takes the next error, the array, and pointer as parameters.

If the stack is full, the function returns `FALSE`; otherwise it adds the error to the stack, changes the pointer's value and returns `TRUE`.

Complete the following pseudocode for the function `AddItemToStack`.

```
FUNCTION AddItemToStack(BYREF ErrorArray : ARRAY[0 : 99] OF Error,
                        BYREF LastItem : INTEGER,
                        BYVALUE Error1 : Error) RETURNS BOOLEAN

   IF LastItem = ............................................

      THEN

         RETURN ............................................

      ELSE

         ErrorArray[LastItem + 1] ← ............................................

         LastItem ← ............................................

         RETURN ............................................

   ENDIF

ENDFUNCTION
```

[4]

(ii) Explain the reasons why `ErrorArray` and `LastItem` are passed by reference, but `Error1` is passed by value.

....

....

....

....

....

.... [3]

(iii) The function `RemoveItem` takes the next error from the stack and returns it.

If there are no errors in the stack, it returns the global record `NullError`.

Complete the pseudocode algorithm `RemoveItem`.

```
FUNCTION RemoveItem(BYREF ErrorArray : ARRAY[0 : 99] OF Error,
                    BYREF LastItem : INTEGER) RETURNS Error

   DECLARE ItemToRemove : Error

   IF ..............................................

      THEN

         RETURN ..............................................

      ELSE

         ItemToRemove ← ErrorArray[..............................................]

         LastItem ← LastItem - 1

         RETURN ..............................................

ENDFUNCTION
```

[3]

(iv) The errors that have been processed are stored in a global queue, `ErrorComplete`.

The function `Enqueue` adds a record to `ErrorComplete`:

```
Enqueue(ErrorToAdd)
```

`Enqueue()` returns `TRUE` if the record is successfully added to the queue, and returns `FALSE` if the queue is full.

The procedure `RunError()` should:

- remove a record from the stack using the function `RemoveItem()`
- output an appropriate message if there were no records in the stack
- if an error record is returned, add the record to the queue using the function `Enqueue()`
- if the record is added to the queue, output an appropriate message
- if the record is not added to the queue, output an appropriate message.

Complete the pseudocode procedure `RunError()`.

```
PROCEDURE RunError(BYREF ErrorComplete : ARRAY[0 : 99] OF Error,
                   BYREF ErrorArray : ARRAY[0 : 99] OF Error)
```

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

```
ENDPROCEDURE
```

[5]

7 A treasure box is hidden within a computer game.

The box has a code that needs to be entered to allow the user into the box. The box contains up to 10 objects that are defined as being of the class `FieldObject`. The definition for the class `Box` is:

Box	
`Size : STRING`	`// small, medium or large`
`Contents : ARRAY[0 : 9] OF FieldObject`	`// the 10 items the box holds`
`Lock : STRING`	`// the code to unlock the box`
`Strength : INTEGER`	`// the strength of the box` `// decreases by 1 each time an` `// incorrect code is entered`
`Constructor()`	`// instantiates an object of the Box` `// class and assigns initial values` `// to the attributes`
`Unlock()`	`// checks if the code is correct to` `// unlock the box`
`GetContents()`	`// returns the array`
`SetSize()`	`// sets the size of the box`
`SetContents()`	`// sets the contents of the box`
`SetLock()`	`// sets the lock code`
`SetStrength()`	`// sets the strength`

(a) The constructor creates a new instance of a box. It takes the size of the box, one item of content and the lock code as parameters. The strength is initialised to `100`.

Write **program code** to create the class and constructor for `Box`.

Do **not** write the code for `Unlock()` or any of the set or get methods. Use the constructor for your chosen language.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(b) The player inputs the code to unlock the box. Each time they enter an incorrect code, the strength of the box decreases by 1. If the strength of the box becomes 0, the box automatically unlocks.

The class `Box` has a method `Unlock()` that:

- takes the code entered as a parameter and checks if it matches the code to unlock the box
- returns `TRUE` if the parameter matches the unlock code
- subtracts 1 from `Strength` if the parameter does not match the unlock code
- checks if the new value of `Strength` is less than 1
- returns `TRUE` if the new value of `Strength` is less than 1, otherwise it returns `FALSE`.

Write **program code** for the method `Unlock()`.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(c) The text file, `Progress.txt`, stores data about the player's previous progress.

The procedure `LoadGame()`:

- opens the text file in read mode
- takes the data from the file and stores the data in the variable `GameData`
- raises an exception with an appropriate message output if it cannot find the file.

Write **program code** for the procedure `LoadGame()`.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

8 A game stores details about characters.

A declarative programming language is used to represent the following knowledge base:

```
01 hair(blonde).

02 hair(black).

03 hair(red).

04 face(glasses).

05 face(moustache).

06 face(beard).

07 person(ismail).

08 person(anisha).

09 person(kim).

10 person(kyle).

11 has(kyle, glasses).

12 has(kyle, beard).

13 has(anisha, red).

14 has(kyle, black).
```

These clauses have the following meaning:

Clause	**Explanation**
`01`	Hair can be blonde
`04`	Glasses can be on the face
`08`	Anisha is a person
`12`	Kyle has a beard
`13`	Anisha has red hair

A person, `X`, is a selected person if they have black hair and either a moustache or a beard.

Write a rule to represent this condition.

```
SelectedPerson(X)

IF
```

....

....

....

.... [2]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

* 5 5 6 9 4 6 3 9 9 4 *

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **October/November 2023**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (EF/CT) 318340/2

1 **(a)** Real numbers are stored in a computer using floating point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

Write the normalised floating-point representation of –96.75 in this system.

Show your working.

Mantissa											**Exponent**					

Working ....

....

....

....

....

.... [3]

(b) Explain why a binary representation is sometimes only an approximation to the real number it represents.

....

....

....

....

....

.... [3]

2 Describe what is meant by **composite** and **non-composite** data types.

Composite ....

....

....

....

Non-composite ....

....

....

....
[4]

3 The location of a record in a random file is determined using a hashing algorithm.

A collision may occur during the process of adding a record.

(a) Outline what is meant by the term **collision** in this context.

....

....

....

.... [2]

(b) Explain how a collision can be dealt with when writing records to a random file.

....

....

....

....

....

.... [3]

4 Complete the following paragraph about a **protocol suite**, using words from the given list.

Some words are **not** used.

BitTorrent	**circuit switching**	**layered**	**link**	**list**
peer-to-peer	**queue**	**stack**	**star**	**TCP/IP**

The protocols in a .... determine the interconnectivity rules for a .... network model such as the .... model.

[3]

5 **(a)** Outline the reasons why an operating system may need to use virtual memory.

....

....

....

.... [2]

(b) Explain the circumstances in which disk thrashing could occur.

....

....

....

....

....

.... [3]

6 **(a)** The Reverse Polish Notation (RPN) expression:

a b * 2 / c d / *

is to be evaluated where a = 20, b = 3, c = 10 and d = 5.

Show the changing contents of the following stack as the RPN expression is evaluated.

[4]

(b) Explain how an expression stored in RPN can be evaluated.

....

....

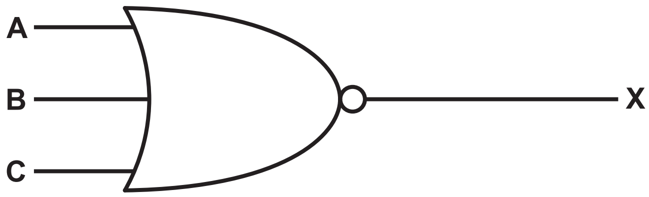
....

....

....

.... [3]

7 **(a)** This logic circuit represents the Boolean expression: $\mathbf{X = \overline{A + B + C}}$



Complete this truth table for the given logic circuit.

A	**B**	**C**	**X**
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

[1]

(b) Apply De Morgan's laws to the expression: $\mathbf{X = \overline{A + B + C}}$

X = ... [1]

(c) Simplify the following expression using Boolean algebra.

Show all the stages in your simplification.

$\mathbf{T = X.Y.Z + X.\overline{Y}.Z + \overline{X}}$

....

....

....

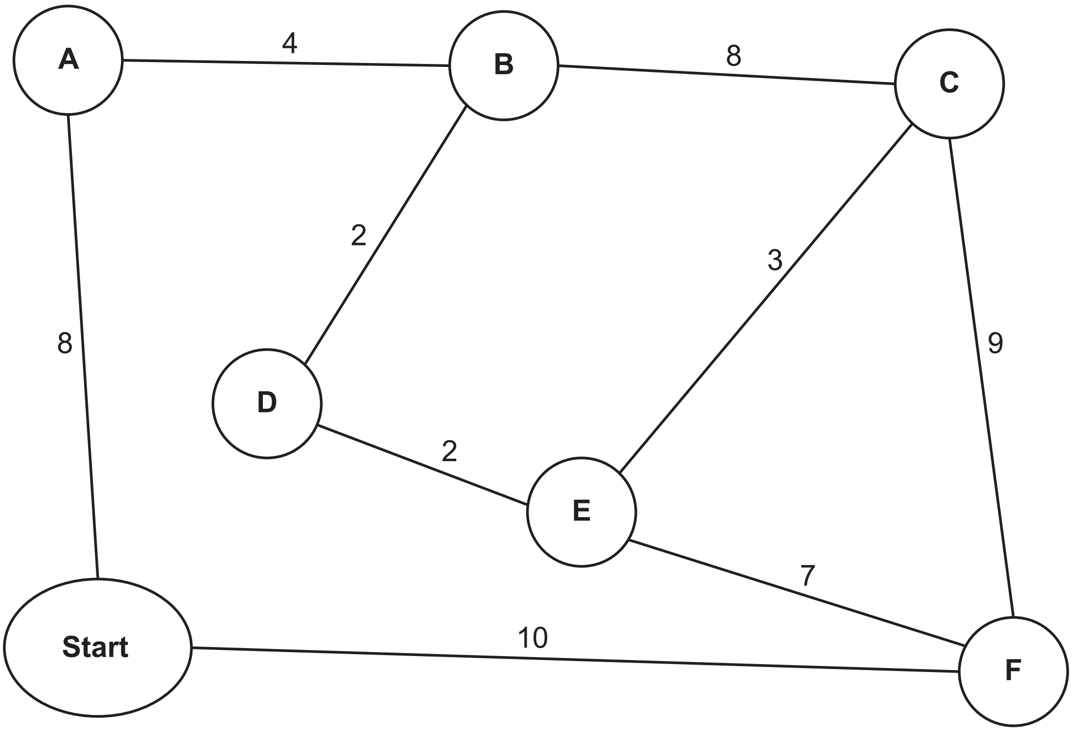
....

....

.... [3]

8 Calculate the shortest distance between the **Start** and each of the destinations in the diagram using Dijkstra's algorithm.

Show your working **and** write your answers in the table provided.



Working ....

....

....

....

....

....

Answers:

A	B	C	D	E	F

[5]

9 (a) A stack Abstract Data Type (ADT) is to be implemented using pseudocode, with procedures to initialise it and to push new items onto the stack.

A 1D array `Stack` stores the contents of the stack.

(i) Study the pseudocode in **part (a)(ii)** and complete the table of identifiers by writing the missing data types and descriptions.

Identifier	Data type	Description
`BasePointer`		
`TopPointer`		
`Stack`	`REAL`	

[2]

(ii) Complete the pseudocode.

```
CONSTANT MaxSize = 40
DECLARE BasePointer : INTEGER
DECLARE TopPointer : INTEGER
DECLARE Stack : ARRAY[1:40] OF REAL

// initialisation of stack
PROCEDURE Initialise()

   ......................................................................... ← 1

   ......................................................................... ← 0
ENDPROCEDURE

// push an item onto the stack
PROCEDURE Push(NewItem : REAL)

   ......................................................................... MaxSize THEN

      ...............................................................................................

      Stack[TopPointer] ← ...........................................................
   ENDIF
ENDPROCEDURE
```

[5]

(b) Justify the use of a linked list instead of an array to implement a stack.

....

....

....

.... [2]

(c) Explain how a compiler makes use of a stack when translating recursive programming code.

....

....

....

....

....

....

....

.... [4]

10 Describe the features of the SIMD and MISD computer architectures.

SIMD ....

....

....

....

MISD ....

....

....

....

[4]

11 A **declarative** programming language is used to represent some facts about people and their hobbies.

```
01 hobby(music).
02 hobby(caving).
03 hobby(climbing).
04 hobby(camping).
05 hobby(baking).
06 hobby(travelling).
07 person(toby).
08 person(natasha).
09 person(fatima).
10 person(joseph).
11 person(elijah).
12 person(nina).
13 enjoys(natasha, travelling).
14 enjoys(toby, climbing).
15 enjoys(nina, climbing).
16 enjoys(elijah, camping).
17 enjoys(fatima, baking).
18 enjoys(joseph, camping).
19 dislikes(toby, caving).
```

These clauses have the meanings:

Clause	Meaning
01	Music is a hobby
07	Toby is a person
13	Natasha enjoys travelling
19	Toby dislikes caving

(a) Carlos is a person who enjoys the hobby of cycling but does not like music.

Write additional clauses to represent this information.

20 ....

21 ....

22 ....

23 ....

[4]

(b) Using the variable `P`, the goal:

```
enjoys(P, camping)
```

returns

```
P = elijah, joseph
```

Write the result returned by the goal:

```
enjoys(P, climbing)
```

`P =` .... [1]

(c) N is a person who might enjoy H if H is a hobby and N does not dislike H.

Write this as a rule.

`might_enjoy(N, H)`

`IF` ....

....

....

.... [4]

12 **(a)** Describe, with an example, what is meant by an **exception**.

....

....

....

.... [2]

(b) A pseudocode algorithm searches for a customer record in a random file `AccountRecord.dat`. A user inputs the name of the customer.

The records are stored using the user-defined data type `TAccount`.

```
TYPE TAccount
    DECLARE AccountNumber : INTEGER
    DECLARE Name : STRING
    DECLARE Address : STRING
    DECLARE Telephone : STRING
ENDTYPE
```

If the record is found, it is output, otherwise an error message is displayed.

Complete the file handling pseudocode.

```
DECLARE Customer : TAccount
DECLARE Location : INTEGER
DECLARE MaxSize : INTEGER
DECLARE FoundFlag : BOOLEAN
DECLARE SearchCustomer : STRING
MaxSize ← 1000

OPENFILE ..................................................................................................
Location ← 1

.......................................................................................................... ← FALSE
OUTPUT "Enter the customer's name"

..........................................................................................................

............................................................................ AND Location <= MaxSize

   ................................ "AccountRecord.dat", .........................................
   GETRECORD "AccountRecord.dat", Customer
   IF SearchCustomer = Customer.Name THEN
      OUTPUT "Customer found: "
      OUTPUT Customer            // output customer record
      FoundFlag ← TRUE
   ENDIF
   Location ← Location + 1
ENDWHILE
IF NOT FoundFlag THEN

   OUTPUT "..................................................................................................."
ENDIF
```

[7]

BLANK PAGE

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

4622048361

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **May/June 2015**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **12** printed pages.

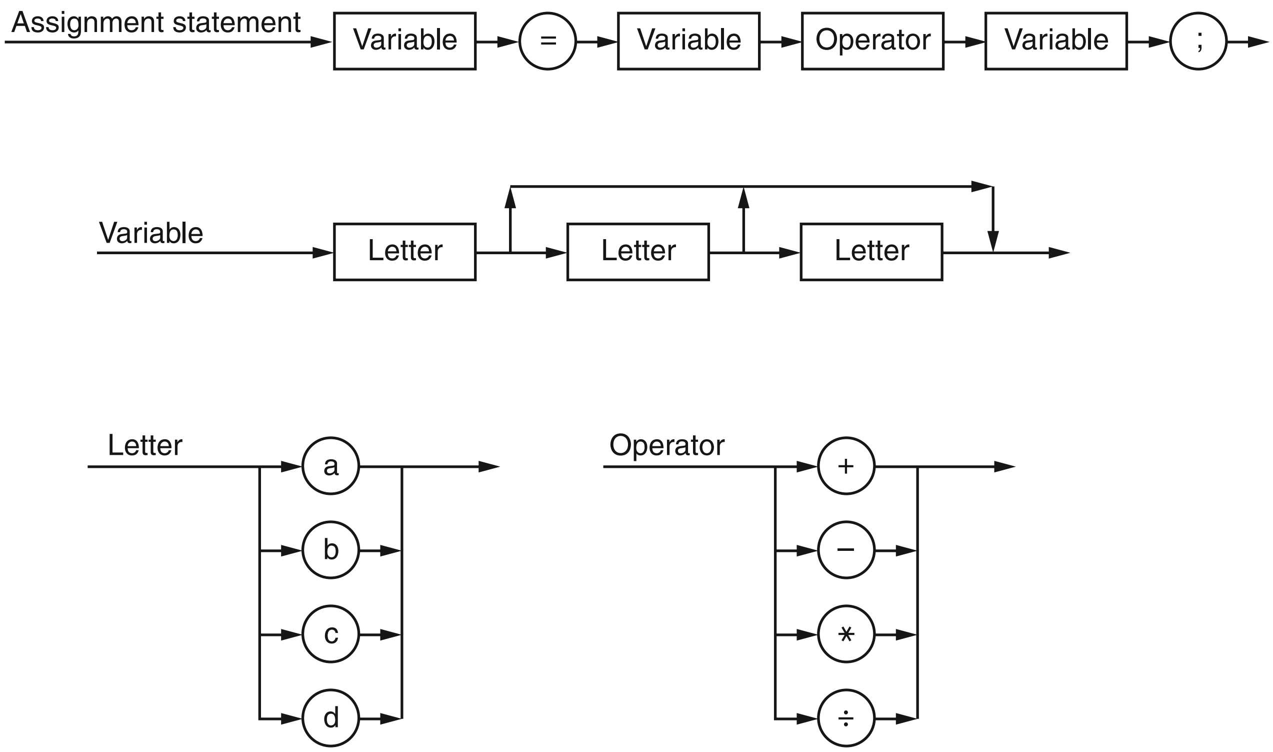
DC (FD) 110069

CAMBRIDGE International Examinations

[Turn over

1 The following syntax diagrams, for a particular programming language, show the syntax of:

- an assignment statement
- a variable
- a letter
- an operator



(a) The following assignment statements are invalid.

Give the reason in each case.

(i) `a = b + c`

Reason ....

....[1]

(ii) `a = b - 2`;

Reason ....

....[1]

(iii) `a = dd * cce`;

Reason ....

....[1]

(b) Write the Backus-Naur Form (BNF) for the syntax diagrams shown on the opposite page.

`<assignmentstatement> ::=`

....

`<variable> ::=`

....

`<letter> ::=`

....

`<operator> ::=`

....[6]

(c) Rewrite the BNF rule for a variable so that it can be any number of letters.

`<variable> ::=`

....[2]

(d) Programmers working for a software development company use both interpreters and compilers.

(i) The programmers prefer to debug their programs using an interpreter.

Give **one** possible reason why.

....

....[1]

(ii) The company sells compiled versions of its programs.

Give a reason why this helps to protect the security of the source code.

....

....[1]

2 The incomplete table below shows descriptions and terms relating to malware.

(a) Complete the table with appropriate descriptions and terms.

	Description	Term
A	Unsolicited emails containing advertising material sent to a distribution list.	....
B	A standalone piece of malicious software that can reproduce itself automatically.	....
C	....	Pharming
D	....	Phishing

[4]

(b) For one of the terms, describe:

- a problem that might arise for a user
- a possible solution to the problem

Choose between the terms:

A / B (circle your choice)

Problem ....

....

Solution ....

....[2]

(c) Explain the following terms:

Encryption ....

....

....

....

Public key ....

....

....

....[2]

(d) A user downloads software from the Internet.

(i) State what should be part of the download to provide proof that the software is authentic.

....[1]

(ii) Describe the process for ensuring that the software is both authentic and has not been altered.

....

....

....

....

....

....

....

....[4]

3 **(a)** A particular programming language allows the programmer to define their own data types.

`ThisDate` is an example of a user-defined structured data type.

```
TYPE ThisDate
   DECLARE ThisDay     :  (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                        13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
                        24, 25, 26, 27, 28, 29, 30, 31)
   DECLARE ThisMonth   :  (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug,
                        Sep, Oct, Nov, Dec)
   DECLARE ThisYear    :  INTEGER
ENDTYPE
```

A variable of this new type is declared as follows:

```
DECLARE DateOfBirth : ThisDate
```

(i) Name the non-composite data type used in the `ThisDay` and `ThisMonth` declarations.

....[1]

(ii) Name the data type of `ThisDate`.

....[1]

(iii) The month value of `DateOfBirth` needs to be assigned to the variable `MyMonthOfBirth`.

Write the required statement.

....[1]

(b) Annual rainfall data from a number of locations are to be processed in a program.

The following data are to be stored:

- location name
- height above sea level (to the nearest metre)
- total rainfall for each month of the year (centimetres to 1 decimal place)

A user-defined, composite data type is needed. The programmer chooses `LocationRainfall` as the name of this data type.

A variable of this type can be used to store all the data for one particular location.

(i) Write the definition for the data type `LocationRainfall`.

....

....

....

....

....

....

....[5]

(ii) The programmer decides to store all the data in a file. Initially, data from 27 locations will be stored. More rainfall locations will be added over time and will never exceed 100.

The programmer has to choose between two types of file organisation. The two types are serial and sequential.

Give **two** reasons for choosing serial file organisation.

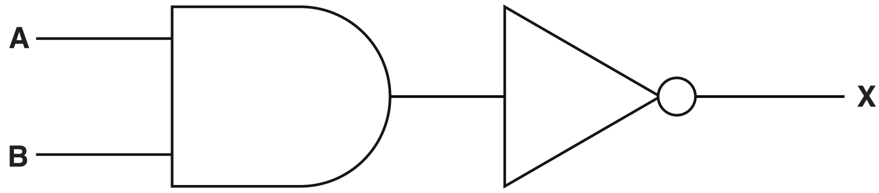
....

....

....

....[2]

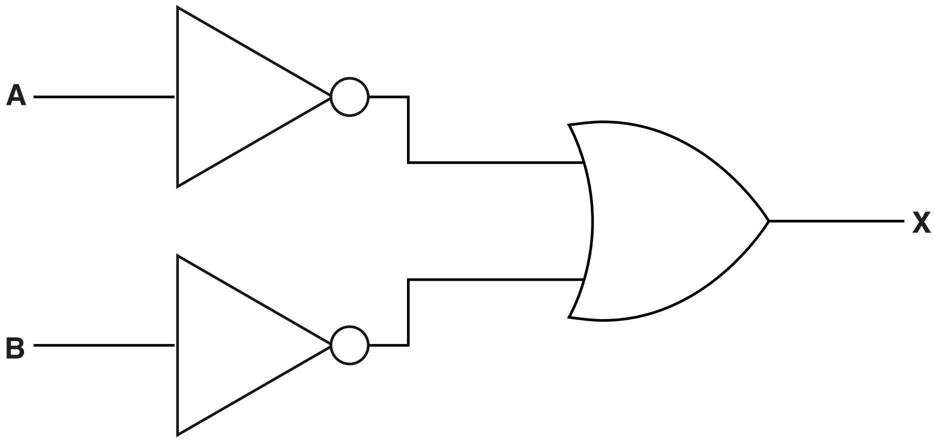
4 (a) (i) Complete the truth table for this logic circuit:



A	B	Working space	X
0	0		
0	1		
1	0		
1	1		

[1]

(ii) Complete the truth table for this logic circuit:



A	B	Working space	X
0	0		
0	1		
1	0		
1	1		

[1]

(b) A student decides to write an equation for **X** to represent the full behaviour of each logic circuit.

(i) Write the Boolean expression that will complete the required equation for **X** for each circuit:

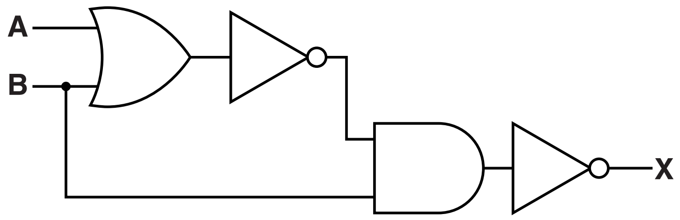
Circuit 1: **X** = ...

Circuit 2: **X** = ...[2]

(ii) Write the De Morgan's Law which is shown by your answers to **part (a)** and **part (b)(i)**.

....[1]

(c) Write the Boolean algebraic expression corresponding to the following logic circuit:



....[3]

(d) Using De Morgan's laws and Boolean algebra, simplify your answer to **part (c)**.

Show all your working.

....

....

....

....

....

....[3]

5 A gardener grows vegetables in a greenhouse. For the vegetables to grow well, the temperature needs to always be within a particular range.

The gardener is not sure about the actual temperatures in the greenhouse during the growing season. The gardener installs some equipment. This records the temperature every hour during the growing season.

(a) Name the type of system described.

....[1]

(b) Identify **three** items of hardware that would be needed to acquire and record the temperature data. Justify your choice for each.

Item 1 ....

Justification ....

....

Item 2 ....

Justification ....

....

Item 3 ....

Justification ....

....[6]

(c) The equipment records temperatures in the greenhouse. It does this for seven locations.

Each recording is stored as two successive bytes. The format is shown below:

Greenhouse location

7	6	5	4	3	2	1	0

Byte 1

Temperature reading

Byte 2

The location is indicated by the setting of one of the seven bits in byte 1. For example, location 4 is indicated by setting bit 4.

Bit 0 of byte 1 acts as a flag:

- the initial value is zero
- when the reading has been processed it is set to 1

Byte 2 contains the temperature reading (two's complement integer).

(i) Interpret the data in byte 1 shown below:

7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	1

Byte 1

0	0	0	1	1	0	0	0

Byte 2

....

....

....

....[2]

(ii) The system receives a temperature reading of –5 degrees from sensor 6.

Complete the boxes below to show the two bytes for this recording. The reading has not yet been processed.

7	6	5	4	3	2	1	0

Byte 1

Byte 2

[2]

(d) **(i)** The accumulator is loaded with the value of byte 1 from location 106.

Write the assembly language instruction to check whether the reading in byte 2 came from location 4.

```
LDD 106            // data loaded from address 106
```

....[4]

(ii) Write the assembly language instruction to set the flag (bit 0) of the byte contained in the accumulator to 1.

....[2]

6 **(a)** Four descriptions and three protocols are shown below.

Draw a line to connect each description to the appropriate protocol.

Description	Protocol used
email client downloads an email from an email server	HTTP
email is transferred from one email server to another email server	POP3
email client sends email to email server	SMTP
browser sends a request for a web page to a web server	

[4]

(b) Downloading a file can use the client-server model. Alternatively, a file can be downloaded using the BitTorrent protocol.

Name the model used.

....[1]

(c) For the BitTorrent protocol, explain the function of each of the following:

(i) Tracker ....

....

....[2]

(ii) Seed ....

....

....[2]

(iii) Swarm ....

....

....[2]

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

1378702323

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **October/November 2018**

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 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.
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.

The maximum number of marks is 75.

This document consists of **15** printed pages and **1** blank page.

DC (ST/SW) 148547/3

CAMBRIDGE
International Examinations

1 A bank provides bank accounts to customers.

The following pseudocode represents the operation of the bank account.

```
CALL OpenAccount()
CALL AccountLifeTime()
CALL CloseAccount()

PROCEDURE AccountLifeTime()
   REPEAT
      CALL Transactions()
   UNTIL AccountClosed() = TRUE
ENDPROCEDURE

PROCEDURE CloseAccount()
   IF ReopenAccount() = TRUE
      THEN
         CALL FlagToReopen()
      ELSE
         CALL DeletePermanently()
   ENDIF
ENDPROCEDURE
```

(a) Complete the JSP structure diagram for this bank account from the pseudocode given.

[6]

(b) A transaction can be a credit (deposit) or a debit (withdrawal).

There are two types of transaction that are credits (deposits). These are:

- SWIFT payment
- BACS payment

There are three types of transaction that are debits (withdrawals). These are:

- Debit card payment
- Cheque payment
- Online payment

Complete the JSP structure diagram to represent these additional requirements.

[4]

2 A declarative language is used to represent facts and rules about dogs that perform tasks to help people.

```
01 type(pointer, gundog).
02 type(flushing, gundog).
03 type(retriever, gundog).
04
05 is_a(labrador, retriever).
06 is_a(newfoundland, retriever).
07 is_a(cocker_spaniel, flushing).
08 is_a(springer_spaniel, flushing).
09 is_a(king_charles, flushing).
10 is_a(english_setter, pointer).
11 is_a(irish_setter, pointer).
12
13 fav_bird(pointer, grouse).
14 fav_bird(flushing, pheasant).
15 fav_bird(retriever, waterfowl).
16
17 type(X, Y) IF is_a(Z, X) AND type(Z, Y).
```

These clauses have the following meaning:

Clause	Explanation
01	A pointer is a type of gundog.
05	A labrador is an example of a retriever.
13	The favourite bird of a pointer is a grouse.
17	X is a type of Y if Z is an example of X and Z is a type of Y.

(a) More facts are to be included.

A **standard poodle** is an example of a waterdog. A waterdog is a type of gundog.

Write the additional clauses to record these facts.

18 ....

19 ....

[2]

(b) Using the variable `P`, the goal

```
is_a(P, retriever)
```

returns

```
P = labrador, newfoundland
```

Write the result returned by the goal

```
is_a(H, pointer)
```

H = ... [2]

(c) Write a query, using the variable `W`, to find out what an **irish setter** is an example of.

.... [2]

(d) `Y` is the favourite bird of dog `X`.

Complete the following rule:

fav_bird(X, Y) IF ....

.... [3]

(e) State the value returned by the goal

```
NOT(is_a(labrador, retriever))
```

.... [1]

3 A bubble sort algorithm is used to sort an integer array, `List`. This algorithm can process arrays of different lengths.

(a) Write **pseudocode** to complete the bubble sort algorithm shown.

```
01 FOR Outer ← ............................................ TO 0 STEP - 1
02   FOR Inner ← 0 TO (...............................................)
03     IF ................................  >  ...........................
04       THEN
05         Temp ← ..................................
06         List[Inner] ← ..................................
07         List[Inner + 1] ← ....................................
08     ENDIF
09   ENDFOR
10 ENDFOR
```

[7]

(b) **(i)** State the order of the sorted array.

.... [1]

(ii) State which line of the algorithm you would change to sort the array into the opposite order.

State the change you would make.

Line ....

Change ....

....

[1]

(c) Use **pseudocode** to write an alternative version of this bubble sort algorithm that will exit the algorithm when the list is fully sorted.

.... [4]

4 A circus is made up of performers. There are three types of performer: clown, acrobat and aerial.

The following data are stored for each performer.

- First name
- Last name
- Secondary role (that can be edited)
- Stage name (that can be edited)
- Type of performer (PerfType)

The following statements apply to performers.

- An acrobat may or may not use fire in his or her act.
- An aerial performer can be one of two types: either catcher or flyer.
- Each clown has an item, such as a water-spraying flower or a unicycle.
- Each clown also has a musical instrument, such as a guitar or an oboe.

Each of the three types of performer has a method that will display all of the information about that performer in a specific format. For example:

Sally Superstar (real name Sally Smith) is an acrobat. Fire is part of Sally Superstar's act. When not performing, Sally Superstar is a set changer.

(a) Complete the following class diagram to show the **attributes**, **methods** and **inheritance** for the program.

You do not need to write the get and set methods.

Performer
FirstName : STRING LastName : STRING SecondaryRole : STRING StageName : STRING PerfType : STRING
Constructor() EditSecondaryRole() EditStageName()

Acrobat
UseFire : BOOLEAN
Constructor() PerformerInfo()

Clown
....
Constructor() ....

Aerial
....
Constructor() ....

[4]

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

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(c) The program will display the acrobat information as follows:

Sally Superstar (real name Sally Smith) is an acrobat. Fire is part of Sally Superstar's act. When not performing, Sally Superstar is a set changer.

Write **program code** for the `Acrobat` class.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

(d) Information about a performer is as follows:

Amazing Alex (real name Alex Tan) is an acrobat. Fire is part of Alex's act. When not performing, Amazing Alex is a popcorn seller.

(i) Write **program code** to create an instance of an object with the identifier `Acrobat_1`.

All attributes of the instance should be fully initialised.

....

....

....

.... [3]

(ii) Explain **inheritance** with reference to the circus example.

....

....

....

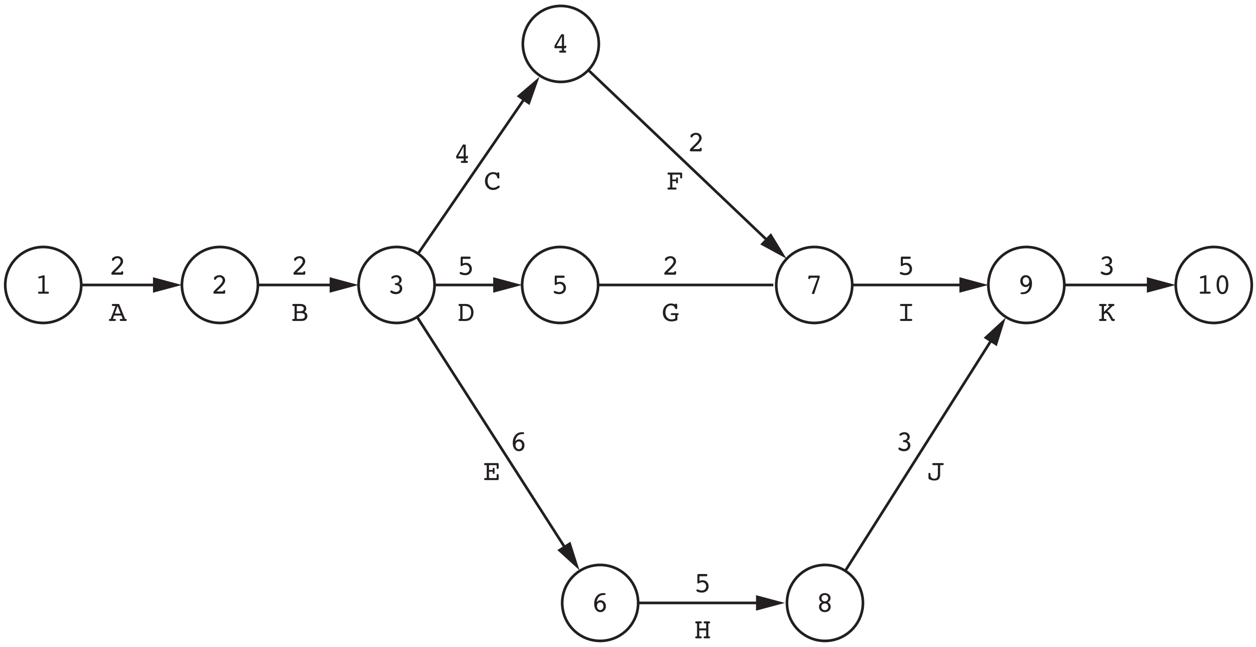
....

....

....

.... [2]

5 A Program Evaluation Review Technique (PERT) chart has been constructed for a project that is at the planning stage.



(a) Complete the following GANTT chart using the information in the PERT chart.

A																									
B																									
C																									
D																									
E																									
F																									
G																									
H																									
I																									
J																									
K																									
Week number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**

[5]

(b) There are three teams working on the project. Each team is able to work on any of the activities.

Explain, with reference to the PERT chart, how work can be allocated to the three teams.

....

....

....

....

....

.... [2]

(c) The PERT chart is used to calculate the critical path for the project.

(i) List the activities that form the critical path using the given PERT chart on page 12.

.... [1]

(ii) Explain the importance of the critical path for project delivery.

....

....

....

....

.... [2]

6 A linked list abstract data type (ADT) is created. This is implemented as an array of records. The records are of type `ListElement`.

An example of a record of `ListElement` is shown in the following table.

Data Item	**Value**
`Country`	`"Scotland"`
`Pointer`	`1`

(a) **(i)** Use **pseudocode** to write a definition for the record type, `ListElement`.

....

....

....

....

.... [3]

(ii) Use **pseudocode** to write an array declaration to reserve space for only 15 nodes of type `ListElement` in an array, `CountryList`. The lower bound element is `1`.

.... [2]

(b) The program stores the position of the last node in the linked list in `LastNode`. The last node always has a `Pointer` value of `-1`. The position of the node at the head of the list is stored in `ListHead`.

After some processing, the array and variables are in the following state.

ListHead
1
LastNode
3

CountryList

	Country	**Pointer**
1	"Wales"	2
2	"Scotland"	4
3		-1
4	"England"	5
5	"Brazil"	6
6	"Canada"	7
7	"Mexico"	8
8	"Peru"	9
9	"China"	10
10		11
11		12
12		13
13		14
14		15
15		3

A **recursive** algorithm searches the list for a value, deletes that value, and updates the required pointers. When a node value is deleted, it is set to empty `""` and the node is added to the end of the list.

A node value is deleted using the pseudocode statement

```
CALL DeleteNode("England", 1, 0)
```

Complete the following **pseudocode** to implement the `DeleteNode` procedure.

```
PROCEDURE DeleteNode(NodeValue: STRING, ThisPointer : INTEGER,
                                                 PreviousPointer : INTEGER)

 IF CountryList[ThisPointer].Value = NodeValue

   THEN

     CountryList[ThisPointer].Value ← ""

     IF ListHead = ..................................................

       THEN

         ListHead ← ..................................................

       ELSE

         CountryList[PreviousPointer].Pointer ← CountryList[ThisPointer].Pointer

     ENDIF

     CountryList[LastNode].Pointer ← ..................................................

     LastNode ← ThisPointer

     ..................................................

   ELSE

     IF CountryList[ThisPointer].Pointer <> -1

       THEN

          CALL DeleteNode(NodeValue, ..................................................,
                                                                     ThisPointer)

       ELSE

          OUTPUT "DOES NOT EXIST"

     ENDIF

 ENDIF

ENDPROCEDURE
```

[5]

BLANK PAGE

Cambridge International A Level

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME			
CENTRE NUMBER		CANDIDATE NUMBER	

5986150518

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **October/November 2015**

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 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.
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.

The maximum number of marks is 75.

DC (CW/SW) 95226/1

CAMBRIDGE International Examinations

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

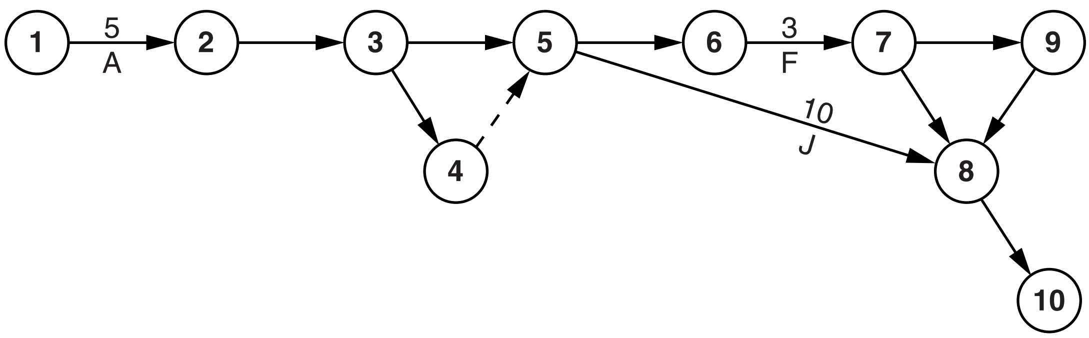
Complete the statement to indicate which high-level programming language you will use.

Programming language ....

1 A large software house has been asked to supply a computerised solution for a business. The project manager has drawn up a list of activities and their likely duration.

Activity	**Description**	**Weeks to complete**
A	Write requirement specification	5
B	Produce program design	5
C	Write module code	15
D	Module testing	10
E	Integration testing	5
F	Alpha testing	3
G	Install software and acceptance testing	5
H	Write end user training guide	5
J	Write technical documentation	10
K	End user training	4
L	Sign off final system	1

(a) The project manager decides to construct a Program Evaluation Review Technique (PERT) chart from this data.



(i) Complete the PERT chart. [7]

(ii) State the critical path.

....[2]

(iii) Calculate the minimum number of weeks for the completion of this solution.

....[1]

(b) For activity J:

(i) State the earliest start time.

Week number ....[1]

(ii) State the latest start time.

Week number ....[1]

(c) Give a reason why the project manager used a PERT chart.

....

....[1]

2 A declarative programming language is used to represent the following facts and rules:

```
01  male(ali).
02  male(raul).
03  male(ahmed).
04  male(philippe).
05  female(meena).
06  female(aisha).
07  female(gina).
08  parent(ali, raul).
09  parent(meena, raul).
10  parent(ali, ahmed).
11  parent(meena, ahmed).
12  parent(ali, aisha).
13  parent(meena, aisha).
14  father(A, B) IF male(A) AND parent(A, B).
```

These clauses have the following meaning:

Clause	Explanation
`01`	Ali is male
`05`	Meena is female
`08`	Ali is a parent of Raul
`14`	`A` is the father of `B` if `A` is male and `A` is a parent of `B`

(a) More facts are to be included.

Philippe and Gina are the parents of Meena.

Write the additional clauses to record this.

`15` ....

`16` ....[2]

(b) Using the variable `P`, the goal

```
parent(P, raul)
```

returns

```
P =  ali, meena
```

Write the result returned by the goal

```
parent(ali, C)
```

`C =` ....[2]

(c) Use the variable `F` to write the goal to find the father of Ahmed.

....[1]

(d) Write the rule to show that `X` is the mother of `Y`.

```
mother(X, Y)
```

`IF` ....

.... [2]

(e) `W` is a grandparent of `Z` if `W` is a parent of one of `Z`'s parents.
Complete the following rule:

```
grandparent(W, Z)
```

`IF` ....

....

....[2]

(f) Complete the rule to show that `G` is a grandfather of `K`.

```
grandfather(G, K)
```

`IF` ....

....

....[2]

3 A lending library stocks two types of item for loan: books and CDs.

All stock items have a title, the date the item was acquired and whether the item is currently out on loan.

Books have an author and ISBN. CDs have an artist and play time in minutes.

The library needs a program to process data about the stock items. The program will use an object-oriented programming language.

(a) Complete the class diagram showing the appropriate properties and methods.

StockItem
`Title: STRING`
....
....
....
`ShowTitle()`
....
....
....

Book
`Author: STRING`
....
....
....
`Constructor()`
`ShowAuthor()`
....
....

CD
....
....
....
....
....
....
....
....

[7]

(b) Write **program code**

(i) for the class definition for the superclass `StockItem`.

Programming language ....

....

....

....

....

....

....

....

....

....

.... [3]

(ii) for the class definition for the subclass `Book`.

Programming language ....

....

....

....

....

....

....

....

....

.... [3]

(iii) to create a new instance of `Book` with:

- identifier `NewBook`
- title "Computers"
- author A.Nyone
- ISBN 099111
- acquired on 12/11/2001
- not out on loan

Programming language ....

....

....

....

....

....

....

....

....[3]

Question 4 begins on page 10.

4 A binary tree Abstract Data Type (ADT) has these associated operations:

- create the tree (`CreateTree`)
- add an item to tree (`Add`)
- output items in ascending order (`TraverseTree`)

(a) Show the final state of the binary tree after the following operations are carried out.

```
CreateTree
Add("Dodi")
Add("Farai")
Add("Elli")
Add("George")
Add("Ben")
Add("Celine")
Add("Ali")
```

[4]

(b) The binary tree ADT is to be implemented as an array of nodes. Each node consists of data and two pointers.

Using pseudocode, a record type, `Node`, is declared as follows:

```
TYPE Node
   DECLARE Name : STRING
   DECLARE LeftPointer : INTEGER
   DECLARE RightPointer : INTEGER
ENDTYPE
```

The statement

```
DECLARE Tree : ARRAY[1:10] OF Node
```

reserves space for 10 nodes in array `Tree`.

The `CreateTree` operation links all nodes into a linked list of free nodes. It also initialises the `RootPointer` and `FreePointer`.

Show the contents of the `Tree` array and the values of the two pointers, `RootPointer` and `FreePointer`, after the operations given in **part (a)** have been carried out.

`RootPointer`

`FreePointer`

`Tree`

	Name	LeftPointer	RightPointer
[1]			
[2]			
[3]			
[4]			
[5]			
[6]			
[7]			
[8]			
[9]			
[10]			

[7]

(c) A programmer needs an algorithm for outputting items in ascending order. To design this, the programmer writes a recursive procedure in pseudocode.

(i) Complete the pseudocode:

```
01 PROCEDURE TraverseTree(BYVALUE Root: INTEGER)
02    IF Tree[Root].LeftPointer ..............................................................
03       THEN
04          TraverseTree(.............................................................)
05    ENDIF
06    OUTPUT ..................................................................................Name
07    IF .........................................................................................<> 0
08       THEN
09          TraverseTree(.............................................................)
10    ENDIF
11 ENDPROCEDURE
```

[5]

(ii) Explain what is meant by a recursive procedure. Give a line number from the code above that shows procedure `TraverseTree` is recursive.

....

....

....

Line number ....[2]

(iii) Write the pseudocode call required to output all names stored in `Tree`.

....

....[1]

Question 5 begins on page 14.

5 Data about sports club members are stored in a random file of records.

- The key field of a member record is the member ID (range 1000 to 9999).
- Other member data are stored.
- A hashing function is used to calculate a record address.
- The random file initially consists of dummy records.
- Dummy records are shown by member ID set to 0.

```
FUNCTION Hash(MemberID : INTEGER) RETURNS INTEGER

   Address ← MemberID MOD 100

   RETURN Address

ENDFUNCTION
```

(a) New members with the following member IDs have joined the sports club:

1001, 3005, 4096, 2098, 7002

Indicate where each record should be stored by deleting the zero and writing the member ID in the correct cell.

MembershipFile

Address	MemberID	Other member data
0	0	
1	0	
2	0	
3	0	
4	0	
5	0	
6	0	
7	0	
8	0	
:		
:		
96	0	
97	0	
98	0	
99	0	

[2]

(b) (i) The program stores a new member's data in the record variable `NewMember`. The field `MemberID` stores the member ID.

Complete the pseudocode:

```
// generate record address

NewAddress ← ..........................................................

// move pointer to the disk address for the record

SEEK ..................................................................

PUTRECORD "MembershipFile", ...........................................
```

[4]

(ii) Before records can be saved to the file `MembershipFile`, the file needs to be opened.

Complete the pseudocode.

```
TRY
    OPENFILE ................................................ FOR RANDOM
EXCEPT
    ..................................................................
ENDTRY
```

[2]

(iii) A record with member ID 9001 is to be stored.

Explain the problem that occurs when this record is saved.

....

....

....

....[2]

(iv) Describe a method, without changing the function `Hash`, to handle the problem identified in **part (b)(iii)**.

....

....

....

....[2]

(v) Write **pseudocode** to implement the method you described in **part (b)(iv)**.

Choose line numbers to indicate where your pseudocode should be inserted in the pseudocode of **part (b)(i)**.

....

....

....

....

....

....

....

....

....

....

....

....[4]

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

6366226742 0

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2017**

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 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.
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.

The maximum number of marks is 75.

This document consists of **17** printed pages and **3** blank pages.

DC (CW/SW) 129963/3

1 The following table shows part of the instruction set for a processor which has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction		**Explanation**
Op code	**Operand**	
`LDM`	`#n`	Immediate addressing. Load the number n to ACC.
`LDD`	`<address>`	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`STO`	`<address>`	Store the contents of ACC at the given address.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`CMP`	`<address>`	Compare the contents of ACC with the contents of <address>.
`JMP`	`<address>`	Jump to the given address.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`AND`	`<address>`	Bitwise AND operation of the contents of ACC with the contents of <address>.
`XOR`	`<address>`	Bitwise XOR operation of the contents of ACC with the contents of <address>.
`OR`	`<address>`	Bitwise OR operation of the contents of ACC with the contents of <address>.
`IN`		Key in a character and store its ASCII value in ACC.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

(a) A programmer writes a program that:

- reads two characters input from the keyboard (you may assume they will be capital letters in ascending alphabetical sequence)
- outputs the alphabetical sequence of characters from the first to the second character. For example, if the characters 'B' and 'F' are input, the output is:

  BCDEF

The programmer has started to write the program in the following table. The Comment column contains descriptions for the missing program instructions, labels and data.

Complete the following program. Use op codes from the given instruction set.

Label	Op code	Operand	Comment
START:			// INPUT character
			// store in CHAR1
			// INPUT character
			// store in CHAR2
			// initialise ACC to ASCII value of CHAR1
			// output contents of ACC
			// compare ACC with CHAR2
			// if equal jump to end of FOR loop
			// increment ACC
			// jump to LOOP
ENDFOR:	END		
CHAR1:			
CHAR2:			

[9]

(b) The programmer now starts to write a program that:

- converts a positive integer, stored at address NUMBER1, into its negative equivalent in two's complement form
- stores the result at address NUMBER2

Complete the following program. Use op codes from the given instruction set.
Show the value stored in NUMBER2.

Label	Op code	Operand	Comment
START:			
		MASK	// convert to one's complement
			// convert to two's complement
	END		
MASK:			// show value of mask in binary here
NUMBER1:	B00000101		// positive integer
NUMBER2:			// negative equivalent

[6]

2 An ordered binary tree Abstract Data Type (ADT) has these associated operations:

- create tree
- add new item to tree
- traverse tree

The binary tree ADT is to be implemented as a linked list of nodes.

Each node consists of data, a left pointer and a right pointer.

(a) A null pointer is shown as Ø.

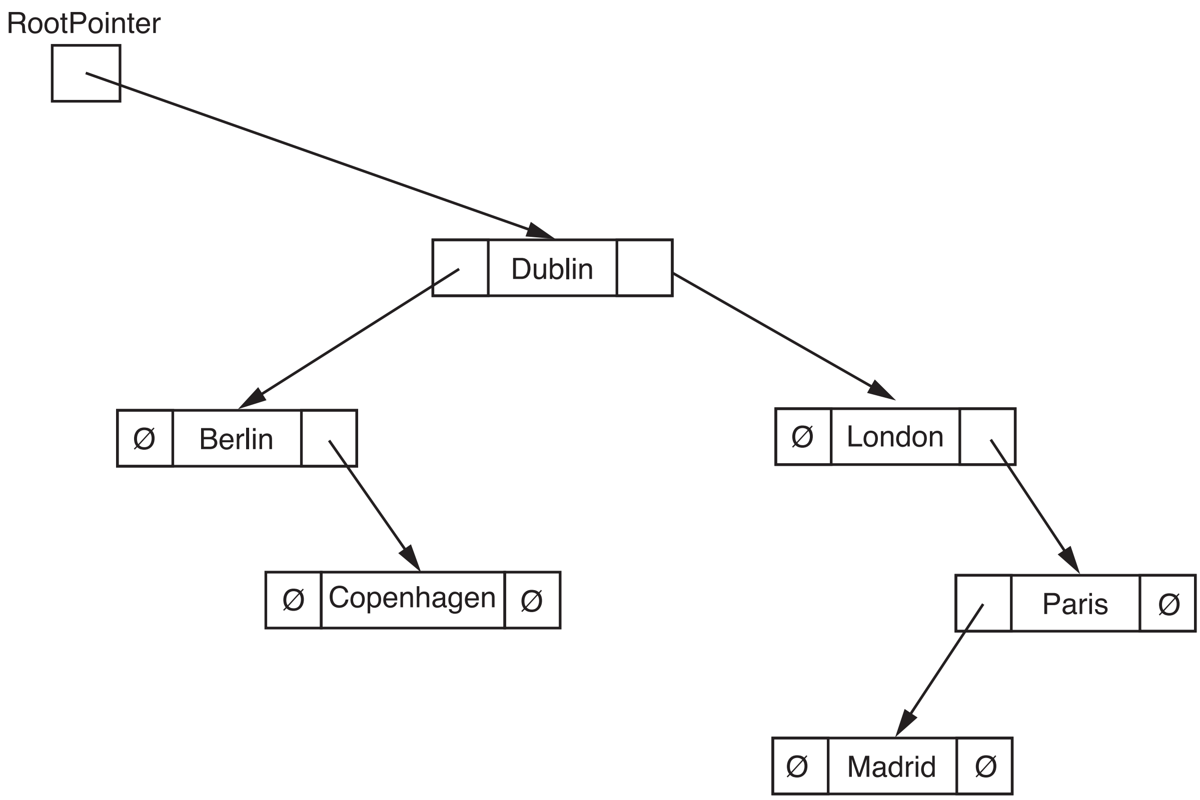
Explain the meaning of the term **null pointer**.

....

....[1]

(b) The following diagram shows an ordered binary tree after the following data have been added:

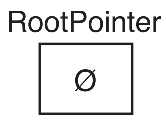
Dublin, London, Berlin, Paris, Madrid, Copenhagen



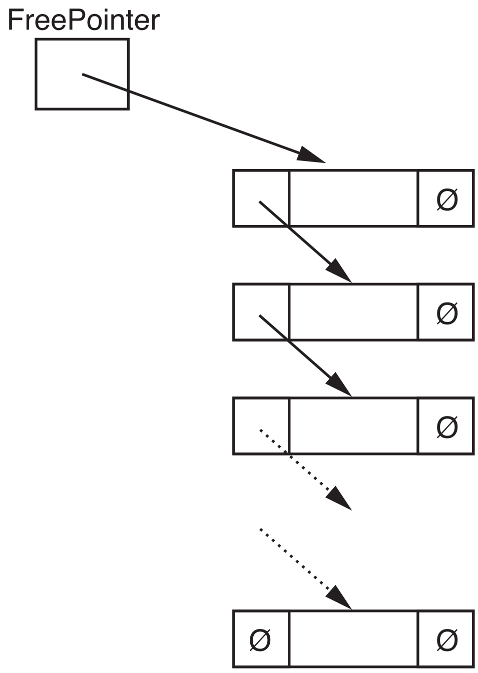
Another data item to be added is Athens.

Make the required changes to the diagram when this data item is added. [2]

(c) A tree without any nodes is represented as:



Unused nodes are linked together into a free list as shown:



The following diagram shows an array of records that stores the tree shown in **part (b)**.

(i) Add the relevant pointer values to complete the diagram.

RootPointer: `0`

FreePointer: ____

	LeftPointer	Tree data	RightPointer
[0]		Dublin	
[1]		London	
[2]		Berlin	
[3]		Paris	
[4]		Madrid	
[5]		Copenhagen	
[6]		Athens	
[7]			
[8]			
[9]			

[5]

(ii) Give an appropriate numerical value to represent the null pointer for this design. Justify your answer.

....

....

....

....[2]

(d) A program is to be written to implement the tree ADT. The variables and procedures to be used are listed below:

Identifier	**Data type**	**Description**
`Node`	`RECORD`	Data structure to store node data and associated pointers.
`LeftPointer`	`INTEGER`	Stores index of start of left subtree.
`RightPointer`	`INTEGER`	Stores index of start of right subtree.
`Data`	`STRING`	Data item stored in node.
`Tree`	`ARRAY`	Array to store nodes.
`NewDataItem`	`STRING`	Stores data to be added.
`FreePointer`	`INTEGER`	Stores index of start of free list.
`RootPointer`	`INTEGER`	Stores index of root node.
`NewNodePointer`	`INTEGER`	Stores index of node to be added.
`CreateTree()`		Procedure initialises the root pointer and free pointer and links all nodes together into the free list.
`AddToTree()`		Procedure to add a new data item in the correct position in the binary tree.
`FindInsertionPoint()`		Procedure that finds the node where a new node is to be added. Procedure takes the parameter `NewDataItem` and returns two parameters: • `Index`, whose value is the index of the node where the new node is to be added • `Direction`, whose value is the direction of the pointer ("`Left`" or "`Right`").

(i) Complete the pseudocode to create an empty tree.

```
TYPE Node

    ..............................................................................

    ..............................................................................

    ..............................................................................

ENDTYPE

DECLARE Tree : ARRAY[0 : 9] ..................................................

DECLARE FreePointer : INTEGER

DECLARE RootPointer : INTEGER

PROCEDURE CreateTree()

   DECLARE Index : INTEGER

    ..............................................................................

    ..............................................................................

   FOR Index ← 0 TO 9   // link nodes

       ..........................................................................

       ..........................................................................

   ENDFOR

    ..............................................................................

ENDPROCEDURE
```

[7]

(ii) Complete the pseudocode to add a data item to the tree.

```
PROCEDURE AddToTree(BYVALUE NewDataItem : STRING)

// if no free node report an error

   IF FreePointer ..................................................

      THEN

         OUTPUT("No free space left")

      ELSE // add new data item to first node in the free list

         NewNodePointer ← FreePointer

         ..................................................

         // adjust free pointer

         FreePointer ← ..................................................

         // clear left pointer

         Tree[NewNodePointer].LeftPointer ← ..................................................

         // is tree currently empty ?

         IF ..................................................

            THEN // make new node the root node

               ..................................................

            ELSE   // find position where new node is to be added

               Index ← RootPointer

               CALL FindInsertionPoint(NewDataItem, Index, Direction)

               IF Direction = "Left"

                  THEN   // add new node on left

                     ..................................................

                  ELSE   // add new node on right

                     ..................................................

               ENDIF

         ENDIF

   ENDIF

ENDPROCEDURE
```

[8]

(e) The traverse tree operation outputs the data items in alphabetical order. This can be written as a recursive solution.

Complete the pseudocode for the recursive procedure `TraverseTree`.

```
PROCEDURE TraverseTree(BYVALUE Pointer : INTEGER)

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

.......................................................................................

ENDPROCEDURE
```

[5]

3 A programmer is writing a treasure island game to be played on the computer. The island is a rectangular grid, 30 squares by 10 squares. Each square of the island is represented by an element in a 2D array. The top left square of the island is represented by the array element [0, 0]. There are 30 squares across and 10 squares down.

The computer will:

- generate three random locations where treasure will be buried
- prompt the player for the location of one square where the player chooses to dig
- display the contents of the array by outputting for each square:
  - `'.'` for only sand in this square
  - `'T'` for treasure still hidden in sand
  - `'X'` for a hole dug where treasure was found
  - `'O'` for a hole dug where no treasure was found.

Here is an example display after the player has chosen to dig at location [9, 3]:

```
..............................
..............................
..............................
..............................
..............................
........T.....................
..............................
..............................
.........T....................
...X..........................
```

The game is to be implemented using object-oriented programming.

The programmer has designed the class `IslandClass`. The identifier table for this class is:

Identifier	Data type	Description
`Grid`	`ARRAY[0 : 9, 0 : 29] OF CHAR`	2D array to represent the squares of the island
`Constructor()`		instantiates an object of class `IslandClass` and initialises all squares to sand
`HideTreasure()`		generates a pair of random numbers used as the grid location of treasure and marks the square with `'T'`
`DigHole(Row, Column)`		takes as parameters a valid grid location and marks the square with `'X'` or `'O'` as appropriate
`GetSquare(Row, Column)`	`CHAR`	takes as parameter a valid grid location and returns the grid value for that square from the `IslandClass` object

(a) The programmer designed the pseudocode for the main program as follows:

```
DECLARE Island : IslandClass.Constructor()      // instantiate object
CALL DisplayGrid()                              // output island squares
FOR Treasure ← 1 TO 3                             // hide 3 treasures
   CALL Island.HideTreasure()
ENDFOR
CALL StartDig()                        // user to input location of dig
CALL DisplayGrid()                              // output island squares
```

Write **program code** to implement this pseudocode.

Programming language used ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....[3]

(b) Write **program code** to declare the `IslandClass` and write the constructor method.

The value to represent sand should be declared as a constant.

Programming language used ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(c) The procedure `DisplayGrid` shows the current grid data. `DisplayGrid` makes use of the getter method `GetSquare` of the `Island` class.

An example output is:

```
..............................
..............................
..............................
..............................
..............................
........T.....................
..............................
..............................
.........T....................
...X..........................
```

(i) Write **program code** for the `GetSquare(Row, Column)` getter method.

....

....

....

....

....[2]

(ii) Write **program code** for the `DisplayGrid` procedure.

....

....

....

....

....

....

....

....

....

....

....[4]

(d) Write **program code** for the `HideTreasure` method. Your method should check that the random location generated does not already contain treasure.

The value to represent treasure should be declared as a constant.

Programming language used ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(e) (i) The `DigHole` method takes two integers as parameters. These parameters form a valid grid location. The location is marked with `'X'` or `'O'` as appropriate.

Write **program code** for the `DigHole` method. The values to represent treasure, found treasure and hole should be declared as constants.

Programming language used ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....[3]

(ii) The `StartDig` procedure:

- prompts the player for a location to dig
- validates the user input
- calls the `DigHole` method from **part (e)(i)**.

Write **program code** for the `StartDig` procedure. Ensure that the user input is fully validated.

Programming language used ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[5]

(f) **(i)** The squares in the `IslandClass` grid could have been declared as objects of a `Square` class.

State the term used to describe the relationship between `IslandClass` and `Square`.

....

....[1]

(ii) Draw the appropriate diagram to represent this relationship. Do not list the attributes and methods of the classes.

[2]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

* 3 3 0 3 4 4 7 8 8 5 *

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **October/November 2021**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages.

1 Data types can be defined using pseudocode.

The data type, `ComputerRecord`, is defined by the following pseudocode:

```
TYPE ComputerRecord
   DECLARE ComputerID       : INTEGER
   DECLARE ComputerType     : (Laptop, Desktop, Tablet)
   DECLARE ComputerLocation : (Lab1, Lab2, Lab3, Mobile)
   DECLARE DateTested       : DATE
ENDTYPE
```

A variable, `SchoolComputer`, is declared in pseudocode as:

```
DECLARE SchoolComputer : ComputerRecord
```

(a) Write **pseudocode** statements to assign `1234` to the `ComputerID` of `SchoolComputer` **and** `Lab2` to the `ComputerLocation` of `SchoolComputer`.

....

.... [2]

(b) The type definition for `ComputerRecord` is changed.

(i) The definition has been extended to include the student identification numbers, `StudentID`, for up to 20 students who can use that computer. Each student identification number is an integer.

Write the extra line of **pseudocode** needed in the type definition for `ComputerRecord`.

....

.... [1]

(ii) The values for the field `ComputerID` must be between `1000` and `1999` inclusive.

Rewrite **one pseudocode** line from the type definition of `ComputerRecord` to implement the change.

....

.... [1]

(c) Data about all the computers are stored in a file that uses random file organisation.

`ComputerID` is used as the key field.

Explain how a program could search for a record stored in this file.

....

....

....

....

....

.... [3]

2 The TCP/IP protocol suite can be viewed as a stack with four layers.

(a) Complete the diagram by writing the names of the **three** missing layers.

Layer

Transport

[3]

(b) State the purpose of each of the following protocols.

HTTP ....

....

....

FTP ....

....

....

POP3 ....

....

....

SMTP ....

....

....

[4]

3 Hamish is constructing a Local Area Network (LAN) using Ethernet with CSMA/CD.

(a) Identify **and** draw a diagram of the most appropriate topology for this LAN.

Topology ....

Diagram:

[3]

(b) Explain how devices on the LAN use CSMA/CD.

....

....

....

....

....

....

....

.... [4]

4 **(a)** The truth table for a logic circuit with four inputs is shown.

INPUT				OUTPUT
P	**Q**	**R**	**S**	**X**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

(i) Write the Boolean algebraic expression for the truth table as a sum-of-products.

X = ... [2]

(ii) Complete the following Karnaugh Map (K-map) for the truth table.

		PQ			
		00	**01**	**11**	**10**
RS	**00**				
	01				
	11				
	10				

[2]

(iii) The K-map can be used to simplify the expression in **part (a)(i)**.

Draw loop(s) around appropriate groups in the K-map to produce an optimal sum-of-products. [2]

(iv) Write the simplified sum-of-products from the K-map.

X = ... [2]

(b) Simplify the expression for **X**, as represented by the truth table in **part (a)**, using Boolean Algebra.

....

....

....

.... [2]

5 **(a)** Flora has written a program that uses the variables `a`, `b`, `c` and `d`.

Part of the program contains the following calculations:

```
a = 2
b = 5
c = 7
d = a * b − (a + b + c)
```

(i) Write the Reverse Polish Notation (RPN) for the expression:

```
a * b − (a + b + c)
```

.... [2]

(ii) Show the changing contents of the stack as the value for variable `d` is calculated from the RPN expression.

[4]

(b) Convert the following RPN expression back to its original infix form.

```
d b * b c d + − + a /
```

....

....

....

....

.... [3]

(c) Explain why expressions are evaluated using RPN.

....

....

....

....

....

.... [3]

6 Mohammad is working away from his company's head office. He wants to send a secure message over a computer network to the head office.

(a) **(i)** Explain the way in which a digital signature for the message would be produced.

....

....

....

....

....

....

....

.... [4]

(ii) State **two** reasons why a digital signature for the message is required.

1 ....

....

2 ....

....
[2]

(b) The message is encrypted using asymmetric key cryptography before it is sent and decrypted when it arrives at the head office.

(i) Describe this process of encrypting and decrypting the message.

....

....

....

....

....

....

....

....

....

.... [5]

(ii) State **one** reason for using asymmetric key cryptography.

....

.... [1]

7 A large apartment block has 20 floors. On each floor there is one security camera and four sensors.

The image from each security camera is output to a display screen for that floor.

- There are 20 display screens in the reception area on the lowest floor.
- The data from the sensors are read and processed by a computer system.
- Warning messages can also be displayed on each display screen.

(a) **(i)** Identify the type of system described.

.... [1]

(ii) Justify your answer to **part (a)(i)**.

....

....

....

.... [2]

(iii) Identify **two** types of sensor that could be used by this system.
State a reason for the use of each sensor.

Sensor 1 ....

Reason ....

....

Sensor 2 ....

Reason ....

....
[4]

(b) A program regularly checks each sensor's readings. If the value of the reading is out of range, a warning message is displayed on the screen for that floor.

A pseudocode algorithm to output the warnings has been written using these identifiers.

Identifier	Data type	Description
`FloorNumber`	`INTEGER`	Floor number
`SensorNumber`	`INTEGER`	Sensor number
`Always`	`BOOLEAN`	Value to ensure continuous loop

The pseudocode algorithm uses:

- the function `CheckSensor(Floor, Sensor)` that returns `TRUE` if the sensor reading is out of range and `FALSE` otherwise
- the procedure `ScreenOut(Floor, Sensor)` that outputs the warning message `"Problem on Floor"` to the appropriate screen.

(i) Complete the **pseudocode** algorithm.

```
01 Always ← ..........................................................................
02 REPEAT
03   FOR FloorNumber ← 1 TO ..........................................................................
04     FOR SensorNumber ← 1 TO ..........................................................................
05        IF CheckSensor(FloorNumber, SensorNumber)
06          THEN
07            ScreenOut(FloorNumber, SensorNumber)
08        ENDIF
09     ENDFOR
10   ENDFOR
11
12 // delay loop
13 // delay loop
14 UNTIL ..........................................................................
```

[4]

(ii) Write a delay loop in **pseudocode** for lines `12` and `13` of the pseudocode algorithm.

....

....

.... [2]

(c) If a sensor reading is out of range, a bit is set in a memory location allocated to that floor. The addresses for the memory locations are 401 to 420.
For example, memory location 401 is used to store the status of the sensors 1 to 4 on floor 1, memory location 402 is used to store the status of the sensors 1 to 4 on floor 2.

The table shows data for some of the floors, with sensor 1 on floor 1 set, sensor 2 on floor 2 set and sensors 3 and 4 on floor 20 set.

Memory location	**Bits**								**Floor**
					Sensor number				
					1	**2**	**3**	**4**	
401	0	0	0	0	1	0	0	0	**1**
402	0	0	0	0	0	1	0	0	**2**
…	…	…	…	…	…	…	…	…	**…**
420	0	0	0	0	0	0	1	1	**20**

(i) The data in memory location 410 is shown.

410	0	0	0	0	0	1	0	1

State what this data represents.

....

.... [2]

(ii) Explain the way in which the data from sensor 3 on floor 7 can be checked.

....

....

....

....

....

....

....

....

....

.... [5]

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME	

CENTRE NUMBER		CANDIDATE NUMBER	

9740267570

COMPUTER SCIENCE **9618/32**

Paper 3 Advanced Theory **October/November 2022**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages. Any blank pages are indicated.

DC (RW/CGW) 302516/4

1 Normalised floating-point numbers are stored in a computer system using two's complement for both the mantissa and the exponent with:

- 11 bits for the mantissa
- 5 bits for the exponent.

(a) Write the largest positive two's complement binary number that can be stored in this system.

Mantissa

Exponent

[1]

(b) Calculate the denary value of the given binary floating-point number.
Show your working.

Mantissa

1	0	1	1	0	0	1	0	0	1	1

Exponent

0	1	0	0	1

Working ....

....

....

....

....

Answer ....

[3]

(c) State when underflow occurs in a binary floating-point system.

....

....

....

.... [2]

2 Lexical analysis and syntax analysis are stages in the compilation of a program.

(a) Identify **two other** stages that take place during the compilation of a program.

1 ....

....

2 ....

....
[2]

(b) Outline the purpose of syntax analysis.

....

....

....

.... [2]

3 **(a)** Explain why a protocol is used in communication between computers.

....

....

....

.... [2]

(b) The TCP/IP protocol implementation can be viewed as a stack.

Complete the diagram for the TCP/IP protocol stack.

Transport
Link

[2]

(c) Describe the purpose of the IMAP protocol.

....

....

....

.... [2]

4 A program to manage regular flight details at an airport requires some user-defined data types.

(a) Write **pseudocode** statements to declare the enumerated data type `Aircraft` to hold data about the types of aircraft used for a flight.

These types of aircraft are: C300, C350, D242, E757, X380.

....

....

....

.... [2]

(b) Write **pseudocode** statements to declare the composite data type `Flight` to hold data about flights to a specific destination. These include:

- flight number, which could be any combination of letters and numbers
- destination
- date of departure
- type of aircraft used.

Use the enumerated data type you created in **part (a)**.

....

....

....

....

....

....

....

.... [4]

(c) **(i)** Write the **pseudocode** statement to set up a variable for one record of the composite data type `Flight`.

....

.... [1]

(ii) Write **pseudocode** to store the details of the following flight in the variable you set up in **part (c)(i)**.

Field	Data
flight number	XA782
destination	Cambridge
date of departure	12/12/2022
type of aircraft used	C350

Use the field names you created in **part (b)**.

....

....

....

....

....

.... [3]

5 Describe what is meant by a **virtual machine**.
Include in your answer **two** benefits and **two** drawbacks of using a virtual machine.

Description ....

....

....

....

Benefit 1 ....

....

Benefit 2 ....

....

Drawback 1 ....

....

Drawback 2 ....

....
[6]

6 **(a)** State **two** differences between symmetric and asymmetric encryption.

....

....

....

.... [2]

(b) Explain the process by which an organisation may acquire its digital certificate.

....

....

....

....

....

....

....

.... [4]

7 Supervised and unsupervised learning are two categories of machine learning.

Describe supervised learning and unsupervised learning.

Supervised learning ....

....

....

....

Unsupervised learning ....

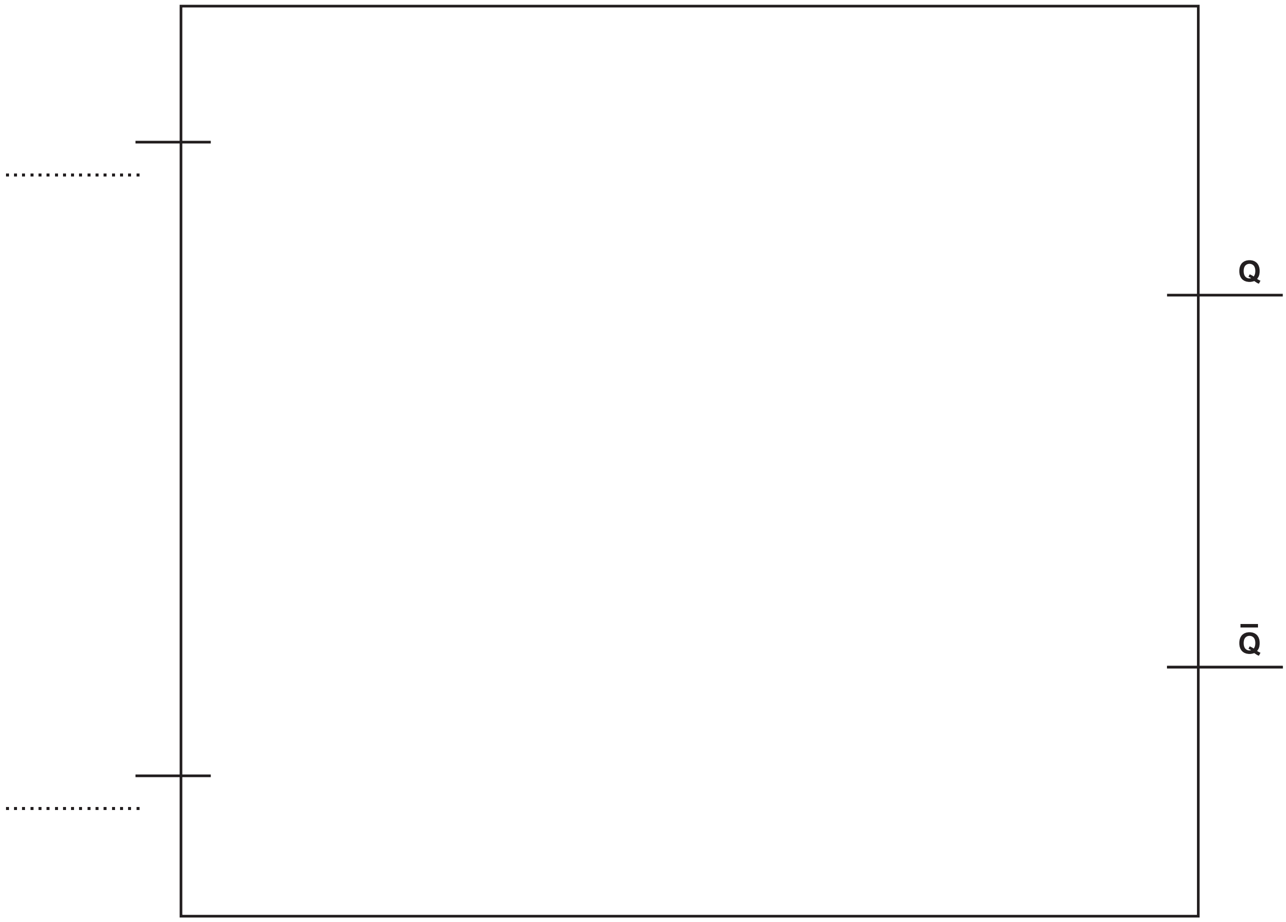
....

....

....

[4]

8 **(a)** Draw a logic circuit for an SR flip-flop **and** label the inputs.



[4]

(b) State the purpose of a flip-flop.

....

.... [1]

(c) Simplify the following expression using Boolean algebra, including De Morgan's laws.
Show your working.

$$\overline{\overline{(\overline{A}.\overline{B})}.\overline{(\overline{A}.C)}.\overline{(B.D)}}$$

Working ....

....

....

....

....

Answer ....

[3]

9 **(a)** Explain the need for scheduling in process management.

....

....

....

....

....

.... [3]

(b) Describe these scheduling routines **and** identify a benefit for each one.

Shortest job first ....

....

....

....

Round robin ....

....

....

....

First come first served ....

....

....

....

[6]

10 (a) Define these Object-Oriented Programming (OOP) terms:

Encapsulation ....

....

Getter ....

....

Setter ....

.... [3]

(b) A school has a program written using OOP to maintain its staff and student records.

The object `SubstituteTeacher` allows the details of the school's substitute teachers to be stored. This includes their full name, telephone number and whether or not they are in school today. For example:

SubName	Sarah Jones
Telephone	01223658721
InSchool	TRUE

Complete the diagram for the object `SubstituteTeacher`, including appropriate properties and their getters and setters.

SubstituteTeacher
SubName : STRING .... InSchool : BOOLEAN
.... SetTelephone(Tel : STRING) .... GetSubName() ....

[3]

11 A simplified linked list is used to store the names of flowers in alphabetical order. It is implemented using two 1D arrays:

- `Flower` stores the names of the flowers.
- `NextPointer` stores the pointer to the next flower name in the list.

`HeadPointer` indicates the index of the first flower name in the linked list.

`HeadPointer` 6

When the end of the linked list is reached, the next pointer has the value of 0.

The following table shows the initial content of the arrays.

Index	Flower	NextPointer
1	Rose	7
2	Marigold	1
3	Foxglove	10
4	Iris	9
5	Daisy	3
6	Dahlia	5
7	Saxifrage	0
8	Lupin	2
9	Lily	8
10	Hydrangea	4

(a) Several flower names have been deleted from the linked list. These are crossed out in the following table.

Complete the table to show the new values of `HeadPointer` and `NextPointer` to keep the remaining flower names in alphabetical order.

`HeadPointer`

Index	Flower	NextPointer
1	Rose	
2	~~Marigold~~	
3	Foxglove	
4	Iris	
5	Daisy	
6	~~Dahlia~~	
7	~~Saxifrage~~	
8	Lupin	
9	Lily	
10	~~Hydrangea~~	

[3]

(b) Complete the pseudocode algorithm so that it achieves the following when applied to the arrays:

- The flower name is input.
- The linked list is searched, in order, for the flower name.
- If the flower name is found, an appropriate message is output to indicate it has been found.
- If the flower name is not found, an appropriate message is output to indicate it has not been found.
- The algorithm terminates when the next pointer value is 0.

```
Pointer ← HeadPointer
Found ← 0
OUTPUT "Enter a flower name "

..................................................................

..................................................................
   IF Flower[Pointer] = FlowerName THEN
      Found ← Pointer
      Pointer ← 0
   ELSE

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

..................................................................
   OUTPUT Flower[Found], " is found"
ELSE

   ..................................................................
ENDIF
```

[5]

(c) Explain how you could improve the simplified linked list structure.

....

....

....

.... [2]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME											
CENTRE NUMBER							CANDIDATE NUMBER				

1587717403

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2021**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS
- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **20** pages. Any blank pages are indicated.

1 A vending machine allows users to insert coins to purchase an item.

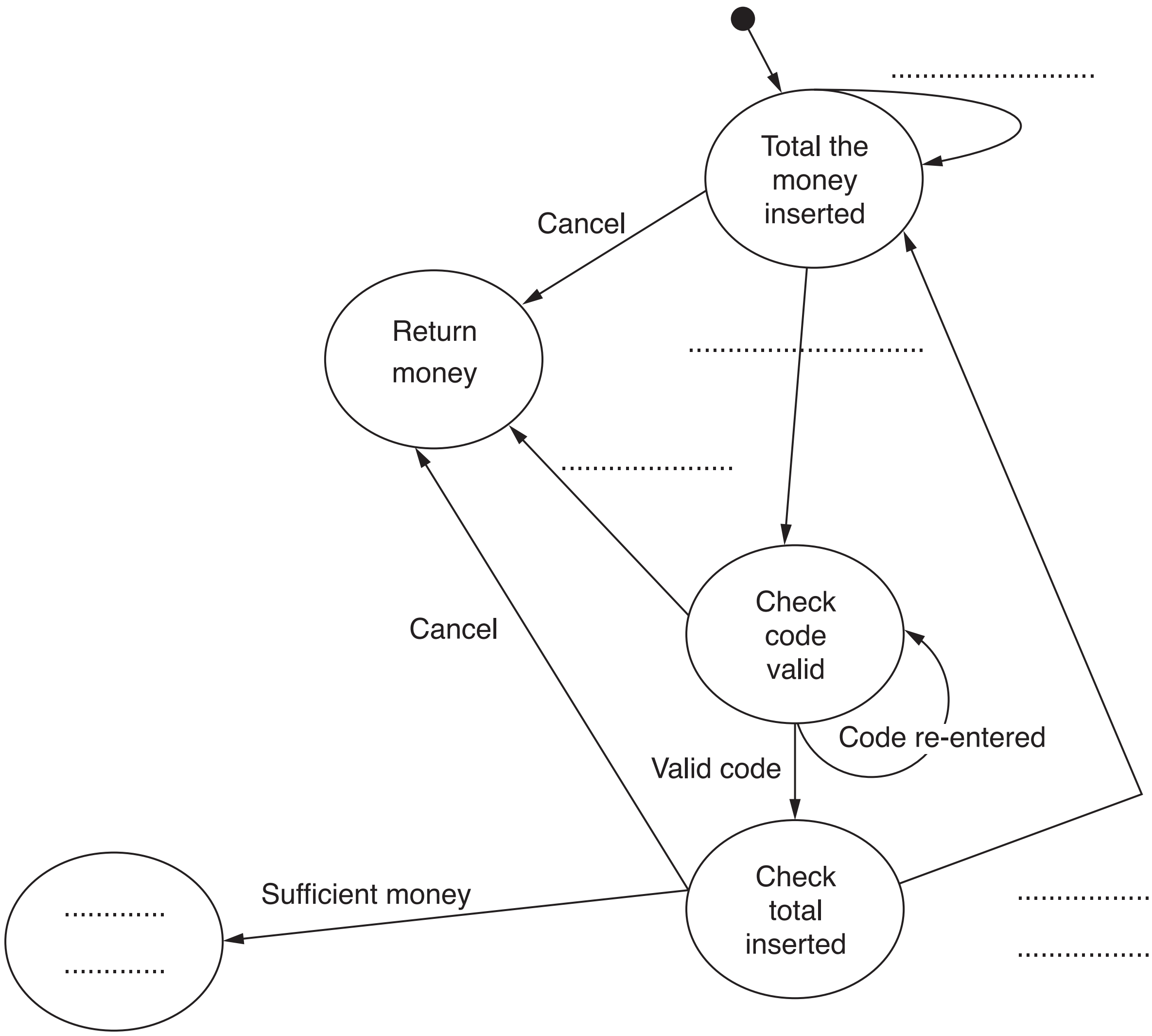
The user then enters the code for the item they would like the machine to dispense (give out). The user must re-enter the code until it is valid.

If the code is valid but the user has not inserted enough money for the item chosen, the machine waits for more coins to be inserted. The user then has to re-enter the code.

The user can press cancel at any time to return the money inserted into the machine.

(a) The state-transition diagram shows the different states of the vending machine.

Complete the state-transition diagram.



[5]

(b) The vending machine is part of a program that is written using object-oriented programming (OOP). The vending machine makes use of two classes that are described in the following tables.

All attributes are declared as private.

foodItem	
`name : STRING`	`// the name of the item of food`
`code : STRING`	`// the code to be entered for that item to be` `// selected`
`cost : REAL`	`// the cost of the item`
`constructor(nameP, codeP,` `costP)`	`// creates an instance of foodItem` `// takes the name, code and cost as parameters`
`getCode()` `getCost()` `getName()`	`// returns the code for the item` `// returns the cost of the item` `// returns the name of the item`

vendingMachine	
`items : ARRAY[0:3] OF foodItem` `moneyIn : REAL`	`// stores four items of type foodItem` `// stores the total money inserted by the` `// user, initialised to 0 in the constructor`
`constructor(item1, item2,` `item3, item4)`	`// creates an instance of vendingMachine,` `// takes four objects of type foodItem as` `// parameters and stores them in array items`
`insertMoney()`	`// takes the value of the coin as a parameter` `// and adds it to moneyIn`
`checkValid ()`	`// takes a code as a parameter and checks it is` `// valid against the food item codes`
`getItemName()`	`// takes the array index as a parameter and` `// returns the name of the food items`

(i) Write **program code** to declare the class `vendingMachine`. You are only required to write program code for the attribute declarations and the constructor.

If you are writing in Python, include attribute declarations using comments.

Use your programming language's constructor method.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

.... [4]

(ii) The method `checkValid()` takes the food item code as a parameter. It checks the code against each element in `items` and returns:

- `-1` if the code is not valid
- `-2` if the code is valid, but the `moneyIn` is less than the cost of the item
- the index of the item, if the code is valid and the `moneyIn` is greater than or equal to the cost of the item.

Write **program code** for the method `checkValid()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

(iii) Four objects of type `foodItem` are declared with the identifiers:

`chocolate, sweets, sandwich, apple`

Write **program code** to declare an instance of `vendingMachine` with the identifier `machineOne` and the objects: `chocolate, sweets, sandwich, apple`.

Programming language ....

Program code

....

....

....

.... [2]

2 Peter uses a record structure, `customer`, to store data about customers. The data includes:

- a unique customer ID between 10 000 and 99 999
- the customer's first name
- the customer's last name
- the customer's telephone number (for example, +44 1234567891).

(a) Write **pseudocode** to define the record type `customer`.

....

....

....

....

....

....

....

.... [3]

(b) The customer records are stored in a random file. The location of each record is calculated as a hash value using:

```
(customer.customerID modulus 1000) + 2
```

(i) Calculate the hash value for each of the customer IDs in the following table.

Customer ID	Hash value
40 125	
10 131	

[1]

(ii) Two or more records could have the same hash value that results in a collision.

Explain how the hashing algorithm can be designed to handle collisions.

....

....

....

....

....

.... [3]

(iii) The function, `getCustomer()`:

- takes the customer ID as a parameter
- passes the customer ID to the function `getRecordLocation()`, which returns the calculated hash value
- reads and returns the record from the hashed location in the file `customerRecords.dat`

You can assume that both the file and the record being accessed exist.

Write **pseudocode** for the function `getCustomer()`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

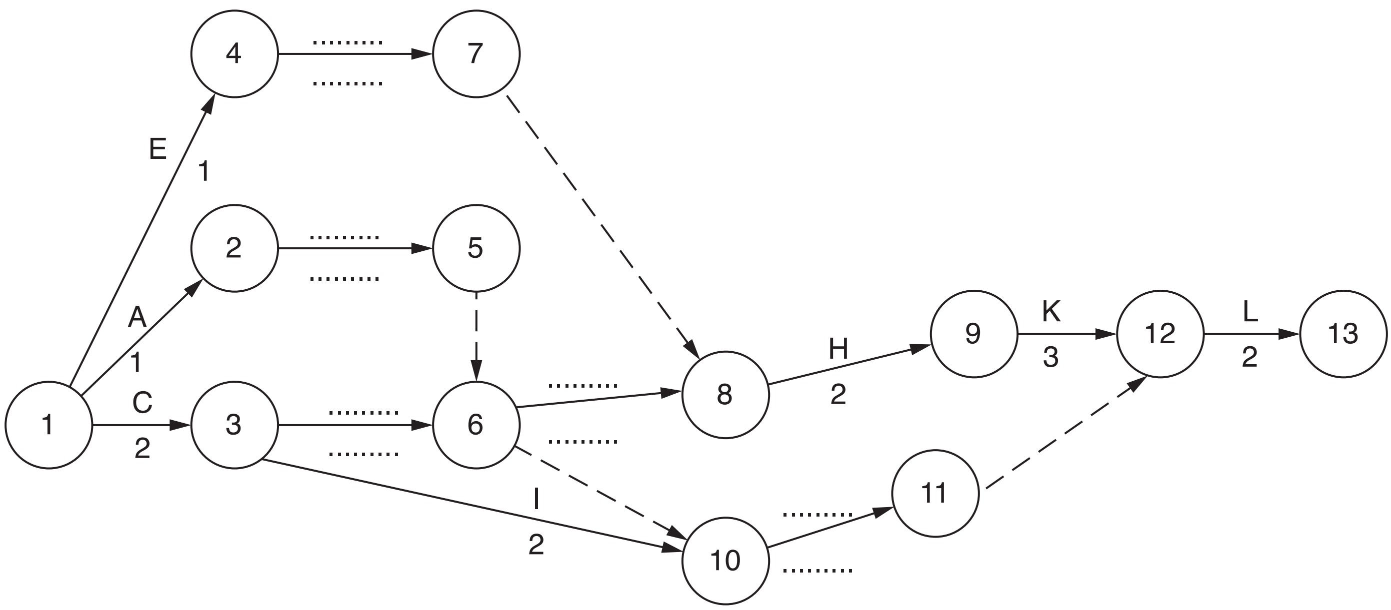
.... [5]

3 Alix manages a team of programmers who are creating a new computer game.

Alix has listed some of the tasks, along with their estimated time to complete and their immediate predecessors in the following table:

Task	Description	Predecessors	Time to complete (weeks)
A	Design character	–	1
B	Program character movement	A	1
C	Design level 1	–	2
D	Program level 1	C	2
E	Design robot	–	1
F	Program robot movement	E	1
G	Integrate character in level 1	B, D	2
H	Integrate robot in level 1	F, G	2
I	Design level 2	C	2
J	Program level 2	D, I	2
K	Test level 1	H	3
L	Integrate character and robot into level 2	J, K	2

(a) Complete the Program Evaluation Review Technique (PERT) chart for the tasks in the table.



[5]

(b) Explain how the tasks in the table can be divided between the team to allow concurrency of tasks.

....

....

....

....

....

.... [2]

(c) Explain the benefits of the team using program libraries in the development of the program.

....

....

....

....

....

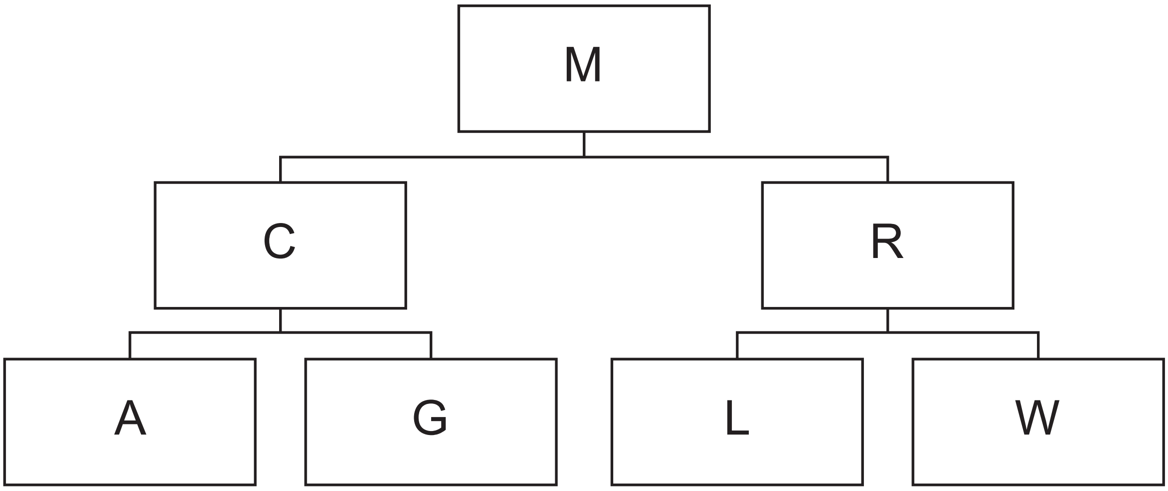
.... [3]

(d) Identify **two** features in an editor that the developers can use to help them create their programs.

Feature 1 ....

Feature 2 ....
[2]

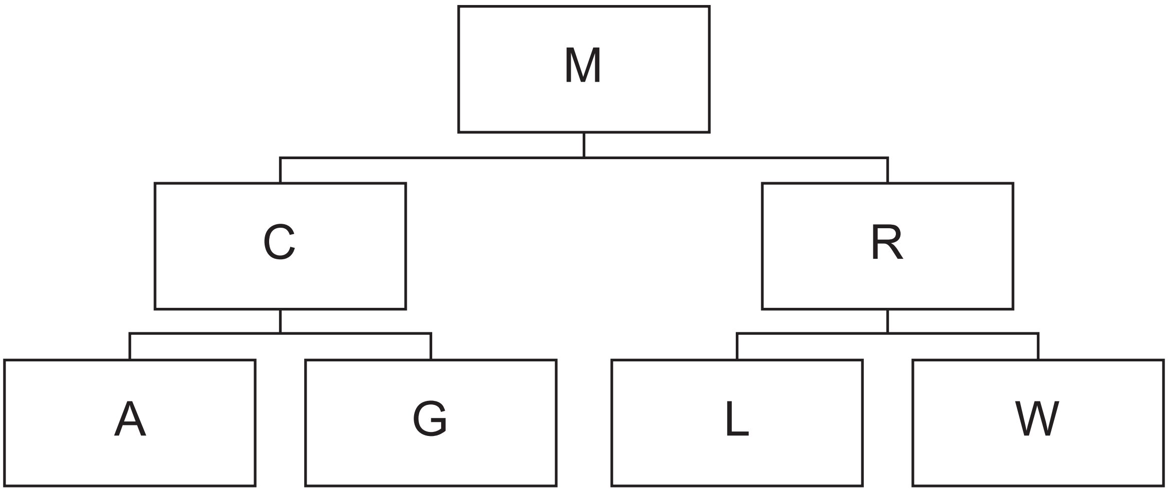
4 Chon creates a binary tree structure to store options that the user can select from a menu (M) in his program.



(a) There are four new options that need to be added.

If option G is selected, the user must choose either option D or option H. If option L is selected, the user must choose either option J or option P.

Complete the following binary tree by adding options D, H, J and P.



[2]

(b) Each node in the binary tree is stored using the following record structure:

```
TYPE node
   leftPointer : INTEGER
   data : STRING
   rightPointer : INTEGER
ENDTYPE
```

The tree is stored as a 1D array, `binaryTree`. Null pointers are represented by `−1`.

(i) The table shows the contents of the three fields in each record stored in the 1D array `binaryTree`.

Complete the table to show the contents of `binaryTree` from **part (a)**.

rootPointer	
freePointer	

Index	**leftPointer**	**data**	**rightPointer**
0		M	
1		C	
2		A	
3		L	
4		G	
5		R	
6		W	
7		J	
8		D	
9		P	
10		H	
11			

[4]

(ii) Write **pseudocode** to declare the array `binaryTree` to store up to 100 objects of type `node`.

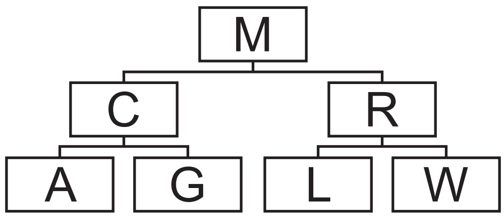
....

....

....

.... [2]

(iii) A pre-order traversal on the following tree would output M C A G R L W



The pre-order traversal can be written as a recursive procedure:

1. output the root node
2. follow the left pointer and repeat from step 1
3. follow the right pointer and repeat from step 1.

Complete the **pseudocode** recursive procedure `preOrder()`.

```
PROCEDURE preOrder(BYVALUE rootPointer : INTEGER)
```

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

```
ENDPROCEDURE
```

[6]

5 A binary search algorithm searches for data in a sorted array.

(a) The pseudocode function `binarySearch()` performs a binary search to find a given value in the global array, `dataArray`. If the value is found, the function returns its index. If the value is not found, the function returns `−1`.

Complete the **pseudocode** for the function `binarySearch()`.

```
FUNCTION binarySearch(BYVALUE upper, lower, searchValue : INTEGER)
                                                       RETURNS INTEGER

  DECLARE flag : INTEGER

  DECLARE mid : INTEGER

  flag ← -2

  mid ← 0

  WHILE flag <> -1

    mid ← lower + ((upper – lower) ............................................)

    IF upper < lower

      THEN

        RETURN ............................................

      ELSE

        IF dataArray(mid) < searchValue

          THEN

            ............................................ ← ............................................

          ELSE

            IF dataArray(mid) > searchValue

              THEN

                ............................................ ← ............................................

              ELSE

                RETURN ............................................

            ENDIF

        ENDIF

    ENDIF

  ENDWHILE

ENDFUNCTION
```

[4]

(b) The binary search algorithm can be written recursively.

Write **program code** for a recursive function `recursiveBinarySearch()`.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [5]

6 The table shows assembly language instructions for a processor that has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

	Instruction		Explanation
Label	**Op code**	**Operand**	
	LDM	#n	Immediate addressing. Load the number n to ACC
	LDD	<address>	Direct addressing. Load the contents of the location at the given address to ACC
	LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the Index Register. Copy the contents of this calculated address to ACC
	LDR	#n	Immediate addressing. Load the number n to IX
	STO	<address>	Store contents of ACC at the given address
	ADD	<address>	Add the contents of the given address to ACC
	INC	<register>	Add 1 to the contents of the register (ACC or IX)
	AND	<address>	Bitwise AND operation of the contents of ACC with the contents of <address>
	XOR	<address>	Bitwise XOR operation of the contents of ACC with the contents of <address>
	OR	<address>	Bitwise OR operation of the contents of ACC with the contents of <address>
	OUT		Output to screen the character whose ASCII value is stored in ACC
	CMP	<address>	Compare the contents of ACC with the contents of <address>
	CMP	#n	Compare the contents of ACC with number n
	JPE	<address>	Following a compare instruction, jump to <address> if the compare was True
	JPN	<address>	Following a compare instruction, jump to <address> if the compare was False
	JMP	<address>	Jump to the given address
	END		Return control to the operating system
<label>:	<op code>	<operand>	Labels an instruction
<label>:	<data>		Gives a symbolic address <label> to the memory location with contents <data>

An algorithm takes each letter of a stored 5-letter word and checks if the letter is upper case.

If the letter is upper case, it outputs the letter.

If the letter is not upper case, it converts the letter to upper case and then outputs it.

All ASCII upper case letters have 010 as the three most significant bits.

Assume each letter is alphabetic.

Complete the assembly language program for the algorithm described using the instruction set provided on the previous page.

Instruction			Comment
Label	Op code	Operand	
	LDR	#0	// load zero to IX
			// load count and check if it is 5
	JPE	endP	// jump to end
	LDX	word	// load letter from indexed address word
			// check if it is upper case
	CMP	#0	
	JPE	output	// jump to output if it is upper case
	LDX	word	// load letter from indexed address word
			// convert to upper case
output:	OUT		// output the character
			// increase count by 1
	INC	IX	// increase IX by 1
	JMP	start	// return to start
endP:	end		// end the program
word:	B01001000		
	B01101111		
	B01110101		
	B01110011		
	B01100101		
mask1:	B00100000		
mask2:	B11011111		
count:	0		

[6]

7 Giles is writing a program that uses a stack.

The stack stores up to 1000 integers in the 1D array, `stackArray`.

(a) The procedure `setUpStack()` takes two parameters:

- the array, `stackArray`
- a pointer to the last element pushed onto the stack, `topOfStack`

The procedure initialises all array elements to `−1` and the pointer to `−1`.

Write **pseudocode** for the procedure `setUpStack()`.

....

....

....

....

....

....

....

....

.... [3]

(b) The function `pop()` pops and returns the item from the top of the stack. If the stack is empty, it returns `−1`.

Write **pseudocode** for the function `pop()`.

....

....

....

....

....

....

....

....

....

.... [3]

BLANK PAGE

BLANK PAGE

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME		
CENTRE NUMBER		CANDIDATE NUMBER

5224848084

COMPUTER SCIENCE **9608/31**

Paper 3 Advanced Theory **May/June 2017**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **16** printed pages.

DC (ST/SG) 133411/3

CAMBRIDGE International Examinations

1 **(a)** Consider the following user-defined data type:

```
TYPE LibraryBookRecord
   DECLARE ISBN       : INTEGER
   DECLARE Title      : STRING
ENDTYPE
```

(i) Write a pseudocode statement to declare a variable, `Book`, of type `LibraryBookRecord`.

....[1]

(ii) Write a pseudocode statement that assigns 'Dune' to the `Title` of `Book`.

....[1]

(b) The user-defined data type `LibraryBookRecord` needs to be modified by adding the following fields:

- a field called `Genre` which can take two values, fiction or non-fiction
- a field called `NumberOfLoans` which can be an integer value in the range 1 to 99

Write the updated version of `LibraryBookRecord`.

....

....

....

....

....

....

....[3]

(c) A pointer is a variable that stores the address of a variable of a particular type.

Consider the code on page 3, which uses the following identifiers:

Identifier	Data type	Description
`IntPointer`	`^INTEGER`	pointer to an integer
`IntVar`	`INTEGER`	an integer variable
`Temp1`	`INTEGER`	an integer variable
`Temp2`	`INTEGER`	an integer variable

```
IntVar ← 57             // assigns the value 57 to the integer
                        // variable IntVar
IntPointer ← @IntVar    // assigns to IntPointer the address of the
                        // integer variable IntVar
Temp2 ← IntPointer^     // assigns to variable Temp2 the value at an
                        // address pointed at by IntPointer
IntPointer^ ← Temp1     // assigns the value in the variable Temp1 to
                        // the memory location pointed at by IntPointer
```

The four assignment statements are executed. The diagram shows the memory contents after execution.

Variable	**Memory address**	**Contents**
	...	
	8217	
`IntVar`	8216	`88`
	8215	
	8214	
	...	
	7307	
`IntPointer`	7306	`8216`
	7305	
	...	
	6717	
`Temp1`	6716	`88`
`Temp2`	6715	`57`
	6714	
	...	

Use the diagram to state the current values of the following expressions:

(i) `@Temp2` ....[1]

(ii) `IntPointer` ....[1]

(iii) `IntPointer^` ....[1]

(iv) `IntPointer^ = Temp2 + 6` ....[1]

(d) Write pseudocode statements that will achieve the following:

(i) Assign the value `22` to the variable `Temp2`.

....[1]

(ii) Place the address of `Temp1` in `IntPointer`.

....[1]

(iii) Copy the value in `Temp2` into the memory location currently pointed at by `IntPointer`.

....[1]

2 The following incomplete table shows descriptions and terms relating to malware.

(a) Complete the table with appropriate description and terms.

	Description	Term	
(i)	A standalone piece of malicious software that can replicate itself using a network.	....	[1]
(ii)	Use email to attempt to obtain an individual's confidential data.	....	[1]
(iii)	....	Virus	[2]

(b) State **two** vulnerabilities that the malware in **part (a)(i)** or **part (a)(iii)** can exploit.

Vulnerability 1 ....

....

Vulnerability 2 ....

....
[2]

Question 2 continues on the next page.

(c) Anna has to send an email to Bob containing confidential information. Bob and Anna have never sent emails to each other before.

Bob and Anna both have public and private keys.

The first step is for Anna to request that Bob sends her one of his keys.

(i) State the key that Bob sends. ....[1]

(ii) Explain how Anna can be sure that it is Bob who has sent the key.

....

....

....

....[2]

(iii) Anna has received the key from Bob.

The following incomplete table shows the sequence of actions between Anna and Bob to communicate the confidential information.

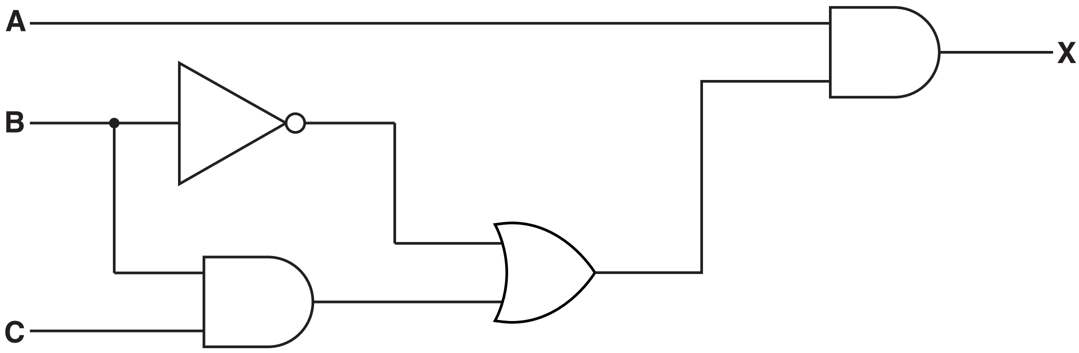
Complete the table.

The person performing the action	**What that person does**
Anna	Requests Bob's <answer to **part (c)(i)**> key.
Bob	....
Anna	....
Anna	Sends the email to Bob.
Bob	....

[4]

Question 3 begins on page 8.

3 Consider the following logic circuit, which contains a redundant logic gate.



(a) Write the Boolean algebraic expression corresponding to this logic circuit.

X = ...[3]

(b) Complete the truth table for this logic circuit.

A	B	C	Working space	X
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

[2]

(c) **(i)** Complete the Karnaugh Map (K-map) for the truth table in **part (b)**.

		AB 00	01	11	10
C	**0**				
	1				

[1]

The K-map can be used to simplify the expression in **part (a)**.

(ii) Draw loop(s) around appropriate groups to produce an optimal sum-of-products. [2]

(iii) Write a simplified sum-of-products expression, using your answer to **part (ii)**.

X = ...[2]

(d) One Boolean identity is:

$$A + \overline{A}.B = A + B$$

Simplify the expression for X in **part (a)** to the expression for X in **part (c)(iii)**. You should use the given identity.

....

....

....

....[2]

4 A bank has 95 000 customers. Each customer has a unique ID.

When a customer uses an Automated Teller Machine (ATM) to obtain cash, their current balance is checked. The balance is stored in a file which has the following fields:

- the customer ID (6-digit number in the range 100000 to 999999)
- an encrypted PIN
- the current balance

The file can store a maximum of 100 000 records.

(a) Give a reason why a random organisation would be appropriate for this file.

....

....[1]

(b) An algorithm for inserting a new record in this file uses the following hash function:

`RecordKey ← CustomerID MOD 100000`

where `RecordKey` is the record position in the file.

(i) Complete the table to show the values generated by the hash function for the given customer IDs.

CustomerID	RecordKey
802139	2139
700004	
689998	
102139	

[1]

(ii) State the range of possible values for `RecordKey`.

Minimum value of `RecordKey`: ....

Maximum value of `RecordKey`: ....
[2]

(iii) A procedure is written to insert a new record into the file.

Complete the algorithm for this procedure.

```
PROCEDURE InsertRecord(CustomerID : INTEGER)
   RecordKey ← CustomerID MOD 100000
   Success ← FALSE
   // Find position for new record and insert it
   REPEAT
      IF record at position RecordKey is ..........................
         THEN
            Insert new record at position RecordKey
            Success ← TRUE
         ELSE
            IF RecordKey = ..........................
               THEN
                  RecordKey ← ..........................
               ELSE
                  RecordKey ← .......................... + 1
            ENDIF
      ENDIF
   UNTIL Success = TRUE
ENDPROCEDURE
```

[4]

(c) (i) Explain why an encrypted version of the PIN is stored in the file.

....

....

....

....[2]

(ii) A customer attempts to withdraw cash from an ATM. An algorithm is used to check if the customer has entered the correct PIN.

Complete the algorithm.

```
1. Customer ID is read from card.

2. Customer enters PIN.

3. Customer PIN is ..........................................................................

4. ...........................................................................................

5. Customer record is located in file.

6. ...........................................................................................

7. If match then transaction can proceed.
```

[3]

5 **(a)** A web browser is used to request and display a page stored on an internet web server.

Explain how each of the following items is used in this event.

(i) Packet: ....

....

....

....[2]

(ii) Router: ....

....

....

....[2]

(iii) TCP/IP: ....

....

....

....[2]

(b) The Internet can be used for video conferencing. Data can be transmitted over the Internet using either packet switching or circuit switching.

(i) State **two** problems that could arise if video conferencing were to use packet switching.

Problem 1 ....

....

Problem 2 ....

....

[2]

(ii) Explain what is meant by **circuit switching**.

....

....

....

....[2]

(iii) Explain how the use of circuit switching overcomes the problems you have identified in **part (i)**.

....

....

....

....

....

....[3]

6 A computer system is used to manage some of the functions in a vehicle. The vehicle has a number of sensors and actuators. One sensor is used to monitor the moisture on the screen. If the moisture exceeds a pre-set value, the windscreen wiper motor turns on automatically.

The software used in the computer system is dedicated to the sensor management functions. When the system starts, the software runs some initial tasks. It then loops continuously until the system is switched off.

(a) **(i)** State the name given to the type of system described.

....[1]

(ii) Explain your answer to **part (i)**.

....

....[1]

(b) Within the software loop, the value of each sensor is read in turn. The value read from the sensor is then processed.

State **two** drawbacks with this method of reading and processing sensor data.

Drawback 1 ....

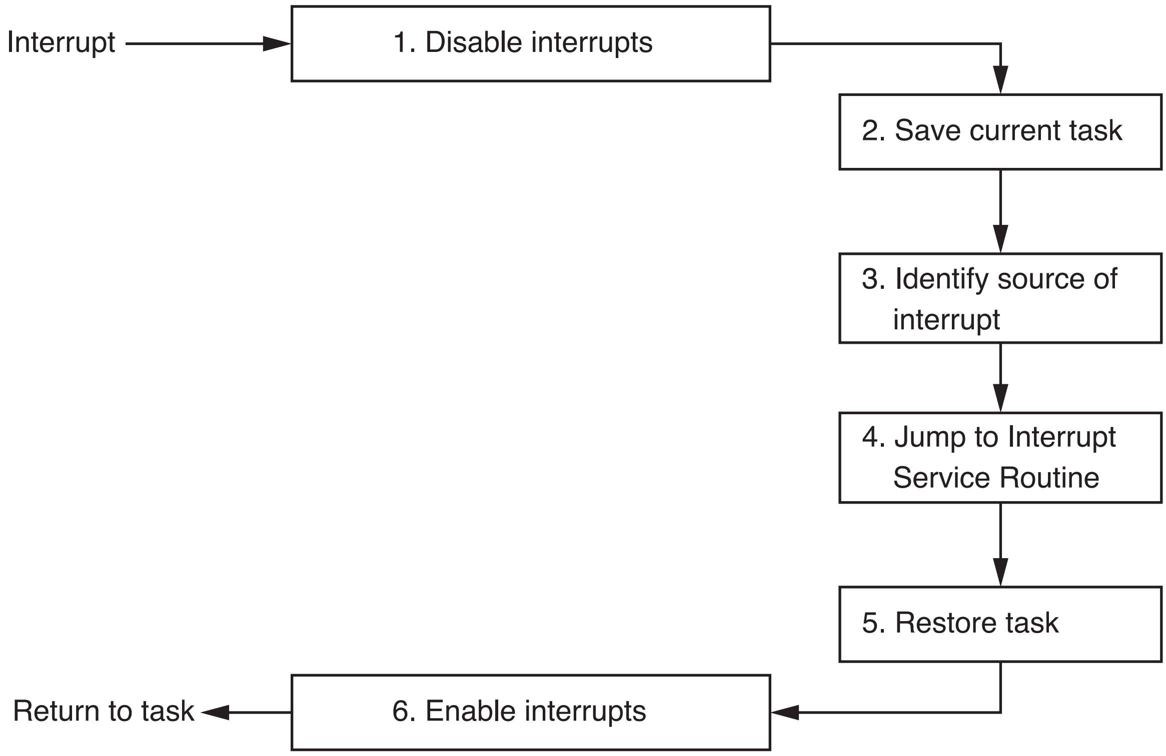
....

Drawback 2 ....

....
[2]

(c) An alternative method of reading and processing sensor data is to use interrupts. Each sensor is connected so that it can send an interrupt signal to the processor if its value changes.

On receipt of an interrupt signal, the processor carries out a number of steps as shown in the following diagram.



(i) State the purpose of step 1.

....

....

....[1]

(ii) State the purpose of step 6.

....

....

....[1]

(iii) Explain how the current task is saved in step 2.

....

....

....

....[2]

(iv) State **two** benefits of using interrupts to read and process the sensor data.

Benefit 1 ....

....

Benefit 2 ....

....
[2]

(v) The interrupt handler in step 3 has to test each bit of a 16-bit register to discover the source of the interrupt.

The contents of the 16-bit register are loaded into the 16-bit accumulator:

Accumulator

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0

An instruction is required to achieve the following:

- If bit 9 is zero, set the accumulator to zero.
- If bit 9 is one, set the accumulator to a non-zero value.

Write this instruction using an appropriate bitwise operation.

....[2]

Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER | CANDIDATE NUMBER

4904680055

COMPUTER SCIENCE **9608/41**

Paper 4 Further Problem-solving and Programming Skills **May/June 2018**

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 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.
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.

The maximum number of marks is 75.

This document consists of **17** printed pages and **3** blank pages.

DC (KN) 164472

CAMBRIDGE
International Examinations

1 A declarative language is used to represent facts and rules about flights.

```
01 direct(edinburgh, paris).
02 direct(palma, rome).
03 direct(glasgow, palma).
04 direct(glasgow, vienna).
05 direct(glasgow, salzburg).
06
07 flies(paris, fly_jet).
08 flies(mumbai, british_air).
09 flies(palma, ciebe).
10 flies(vienna, fly_jet).
11 flies(salzburg, ciebe).
12
13 can_fly(X, Y) IF direct(X, Z) AND direct(Z, Y).
```

These clauses have the following meaning:

Clause	Explanation
01	There is a direct route from Edinburgh to Paris.
07	Fly Jet operates flights to Paris.
13	It is possible to fly from X to Y if there is a direct flight from X to Z and a direct flight from Z to Y.

(a) More facts need to be included.

There is a direct flight from London to Rome and British Air flies to Rome.

14 ....

15 .... [2]

(b) Using the variable `Q`, the goal

```
flies(Q, fly_jet).
```

returns

```
Q = paris, vienna
```

Write the result returned by the goal

```
flies(K, ciebe).
```

`K` = ...[2]

(c) Use the variable `M` to write the goal to find where you can fly direct from Glasgow.

....[2]

(d) If an airline flies to an airport, that airline also flies every direct route out of that airport.

Write a rule to represent this condition.

```
flies(Y, X)
```

`IF` ....

....[3]

(e) State what the following goal returns.

```
can_fly(glasgow, rome).
```

....[1]

2 The array `ItemList[1:20]` stores data. A **bubble sort** sorts these data.

(a) Complete the pseudocode algorithm for a bubble sort.

```
MaxIndex ← 20
NumberItems ← ..........................................................................................
FOR Outer ← 1 TO .......................................................................................
   FOR Inner ← 1 to NumberItems
      IF ItemList[Inner] > ..............................................................
         THEN
            Temp ← ItemList[............................................................]
            ItemList[Inner] ← ItemList[.............................................]
            ItemList[Inner + 1] ← ........................................................
      ENDIF
   ENDFOR
   NumberItems ← ..................................................................................
ENDFOR
```

[7]

(b) The algorithm in **part (a)** is inefficient.

(i) Explain why the algorithm in **part (a)** is inefficient.

....

....

....

....[2]

(ii) Explain how you would improve the efficiency of this algorithm.

....

....

....

....

....[3]

(c) An insertion sort is another sorting algorithm.

State **two** situations when an insertion sort is more efficient than a bubble sort. Give a reason for each.

Situation 1 ....

....

Reason ....

....

....

Situation 2 ....

....

Reason ....

....

....

[4]

3 An internet based music streaming service provides access to an unlimited number of songs for members to play.

The following pseudocode represents the operation of the service.

```
CALL OpenAccount()
CALL OperateAccount()
CALL CloseAccount()

PROCEDURE OperateAccount()
   WHILE RequestCloseAccount() = FALSE
      IF SubscriptionDue() = TRUE
         THEN
            CALL MakePayment()
         ELSE
             CALL PlaySong()
      ENDIF
   ENDWHILE
ENDPROCEDURE
```

(a) Complete the JSP structure diagram for this music service from the pseudocode given.

Music Service

[5]

(b) The service needs extending so that members can download songs to play offline.

- When a member selects a song, the service checks if the song has already been downloaded.
- If the member has already downloaded the song, the member has the option to delete or play it.
- If the member has not already downloaded the song they have the option to download or stream it.

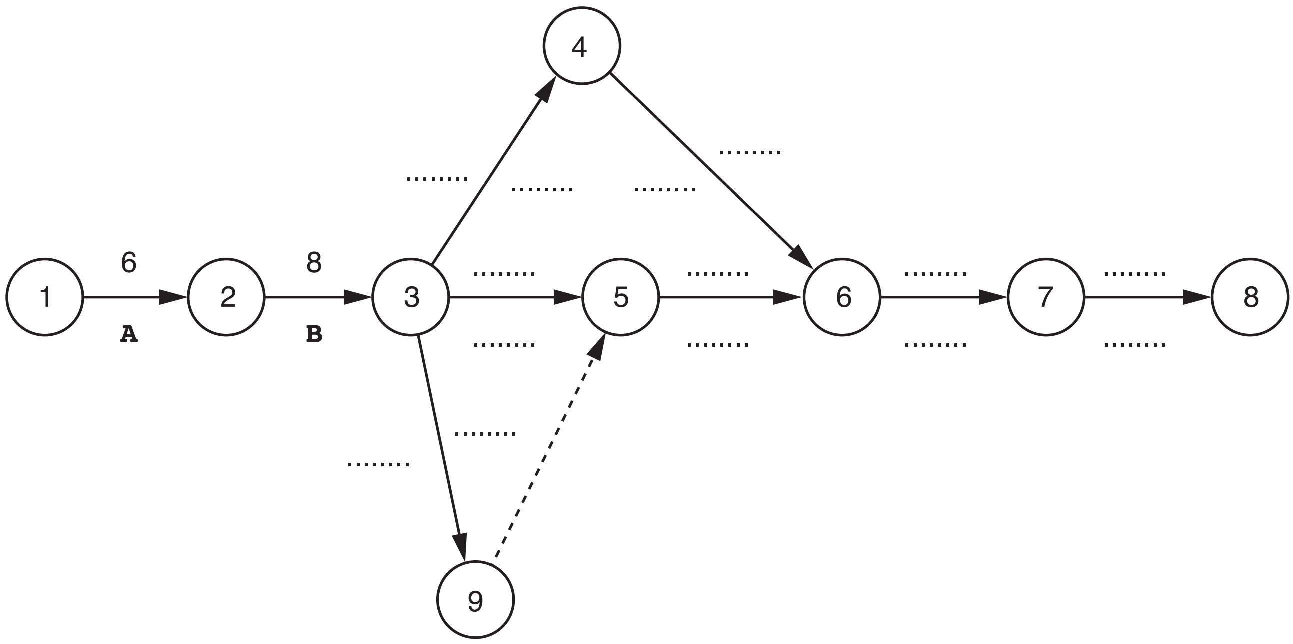
Complete the following JSP structure diagram to represent these new requirements.

[4]

4 A software company is developing a new application. The project manager has created a work breakdown structure, as shown in the following table.

Activity		Days to complete	Predecessor
A	Gather user requirements	6	
B	Design work	8	A
C	Develop server code	4	B
D	Develop application code	5	B
E	User interface development	6	B
F	Test server code	2	C
G	Test application	2	D,E
H	Test application/server integration	5	F,G
I	Roll out mobile application	3	H

(a) Use the data in the table to complete the following Program Evaluation Review Technique (PERT) chart.



[5]

(b) Calculate the critical path (CP). State the:

activities that form the CP ....

duration of the CP ....

[2]

(c) For activity F, state the:

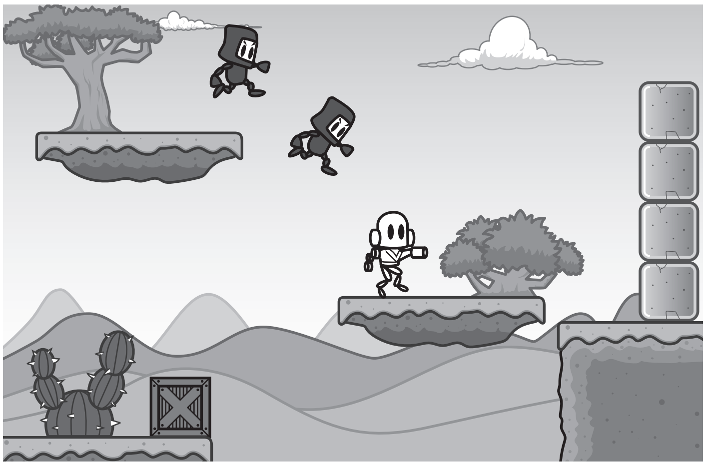
earliest start time ....

latest finish time ....

[2]

Question 5 begins on the next page.

5 A computer game is being developed using object-oriented programming. The following image is a screenshot from the game.



There are scenery elements and animated elements. The player's character is one of the animated elements.

Each game element has the attributes:

Attribute	Description	Example value
`PositionX`	The x coordinate of the game element.	`92`
`PositionY`	The y coordinate of the game element.	`106`
`Width`	The width of the game element.	`150`
`Height`	The height of the game element.	`200`
`ImageFilename`	The filename of the image file for the game element.	`GameElementFrame1.png`

Each game element has a method, `GetDetails()` that returns a string containing all the element's attributes.

The player's character is one of a number of animated elements. All animated elements have the attributes:

Attribute	Description	Example value
`AnimationFrames`	An array of `GameElement`	
`Direction`	A string giving the direction the object is travelling in.	`"Left"`
`Strength`	A value for the strength that indicates the power of the object.	`2000`
`Health`	A value for the health that indicates the health of the object.	`100`

The player's character can either move left or right, or jump.

(a) Complete the following class diagram for the game.

You do not need to include any additional get or set methods.

GameElement
PositionX: INTEGER PositionY: INTEGER Width: INTEGER Height: INTEGER ImageFilename: STRING
Constructor() GetDetails()

AnimatedElement
AnimationFrames: ARRAY OF GameElement ....
Constructor() AdjustHealth() AdjustStrength() DisplayAnimation()

Scenery
CauseDamage: BOOLEAN DamagePoints: INTEGER
Constructor() GivePoints()

Player
....

[3]

(b) Write **program code** to define the `GameElement` class.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

(c) The `Scenery()` class has two attributes, `CauseDamage` and `DamagePoints`.

If the attribute `CauseDamage` is `TRUE`, then the scenery element can cause damage.

The method `GiveDamagePoints()` checks whether the object can cause damage. If the object can cause damage, the method returns the integer value of the `DamagePoints` attribute.

Write **program code** for the `Scenery` class.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

(d) A new scenery object, `GiftBox`, is to be created.

(i) The attributes of `GiftBox` are as follows:

Attribute	Value
`PositionX`	`150`
`PositionY`	`150`
`Width`	`50`
`Height`	`75`
`ImageFilename`	`"box.png"`
`CauseDamage`	`TRUE`
`DamagePoints`	`50`

Write **program code** to create an instance of `GiftBox`.

Programming language ....

Program code

....

....

....

....

....

....

....[3]

(ii) An additional method, `GetScenery()`, returns all the attributes of the `Scenery` class.

Write **program code** for the `GetScenery()` method.

You should use the `GetDetails()` method that the `Scenery` class inherits from the `GameElement` class.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....[3]

6 An Abstract Data Type (ADT) is used to create a linked list. The linked list is created as an array of records. The records are of type `ListNode`.

An example of a record of `ListNode` is shown in the following table.

Data Field	**Value**
`Player`	`"Alvaro"`
`Pointer`	`1`

(a) **(i)** Use **pseudocode** to write a definition for the record type, `ListNode`.

....

....

....

....

....[3]

(ii) An array, `Scorers`, will hold 10 nodes of type `ListNode`. Use **pseudocode** to write an array declaration for this array. The lower bound subscript is `0`.

....[2]

(b) The linked list stores `ListNode` records in alphabetical order of player. The last node in the linked list always has a `Pointer` value of `-1`. The position of the first node in the linked list is held in the variable `ListHead`.

After some processing, the array and variables are in the state as follows:

ListHead
0

Scorers

	Player	**Pointer**
0	"Alvaro"	1
1	"Antoine"	3
2	"Dimitri"	7
3	"Cristiano"	2
4	"Gareth"	5
5	"Graziano"	6
6	"Olivier"	8
7	"Erik"	4
8	"Yaya"	9
9	"Zoto"	-1

A **recursive** function traverses the linked list to search for a player.

An example of calling the function, using pseudocode, is:

```
Position ← SearchList("Gareth", ListHead)
```

Complete the following **pseudocode** to implement the function `SearchList()`.

The function will return a value of `99` when a player is not found.

```
FUNCTION SearchList(Find : STRING, Position : INTEGER) RETURNS INTEGER

   IF Scorer[Position].Player = ............................................................

      THEN

         RETURN ...................................................................................

      ELSE

         IF Scorer[Position].Pointer <> -1

            THEN

               Position ← SearchList(Find, ........................................)

               RETURN ...........................................................................

            ELSE

               RETURN ...........................................................................

         ENDIF

   ENDIF

ENDFUNCTION
```

[5]

BLANK PAGE

BLANK PAGE

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

4911392142

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **October/November 2022**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **12** pages. Any blank pages are indicated.

DC (LK/CGW) 302515/4

1 Real numbers are stored in a computer system using floating-point representation with:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both mantissa and exponent.

(a) Write the normalised floating-point representation of +202 in this system.
Show your working.

Mantissa **Exponent**

Working ....

....

....

....

....

.... [3]

(b) Write the normalised floating-point representation of –202 in this system.
Show your working.

Mantissa **Exponent**

Working ....

....

....

....

....

.... [3]

(c) A binary number is stored in the computer system.

Mantissa

0	0	0	1	1	1	1	0

Exponent

0	0	0	1	1	0	0	0

(i) State why the number is **not** normalised.

....

.... [1]

(ii) Write the normalised floating-point representation of the number.

Mantissa

Exponent

[2]

2 Outline the functions of the Transport and Internet layers of the TCP/IP protocol suite.

Transport layer ....

....

....

....

Internet layer ....

....

....

....
[5]

3 **(a)** State what is meant by the term **enumerated data type**.

....

.... [1]

(b) State what is meant by the term **pointer data type**.

....

.... [1]

(c) The months of the year are: January, February, March, April, May, June, July, August, September, October, November and December.

Write the **pseudocode** statement to define the data type `Quarter1`, to hold the names of the first three months of a year.

....

....

....

.... [2]

(d) The composite data type `Pet` is used to store data about the various pets of a group of students. It uses these fields:

Field name	Data type
`PetName`	String
`AnimalType`	String
`PetAge`	Integer
`PetGender`	Char
`OwnerName`	String

(i) Write the **pseudocode** statement to set up a variable for one record of the composite data type `Pet`.

....

.... [1]

(ii) Write **pseudocode** to store the details of the following pet, in the variable you set up in **part (d)(i)**.

PetName	AnimalType	PetAge	PetGender	OwnerName
Tibbles	Cat	8	M	Jasmine Smith

....

....

....

....

....

.... [3]

4 Draw **one** line to connect each stage of compilation to its **most appropriate** description.

Stage of compilation	**Description**
	minimising a program's execution time and memory requirement
Lexical analysis	converting an intermediate representation of source code into an executable form
Syntax analysis	converting a sequence of characters into a sequence of tokens
Code generation	directly executing instructions written in a scripting language
Optimisation	using parsing algorithms to interpret the meaning of a sequence of tokens

[4]

5 **(a)** Write the infix expression in Reverse Polish Notation (RPN).

```
a * b + b - d + 15
```

....

.... [1]

(b) **(i)** Write the RPN expression in infix form.

```
a b - c d + * a /
```

....

.... [1]

(ii) Evaluate your infix expression from **part (b)(i)** when a = 5, b = 10, c = 27 and d = 12.

....

.... [1]

6 A message is encrypted using a private key and sent to an individual using asymmetric encryption.

(a) State what is meant by a **private key**.

....

....

....

.... [2]

(b) Describe the process of asymmetric encryption.

....

....

....

.... [2]

(c) Explain how a digital signature is used to verify a message when it is received.

....

....

....

....

....

....

....

.... [4]

7 **(a)** Complete the Karnaugh map (K-map) for the Boolean expression.

$$\mathbf{Z = \overline{A}.\ B.\ \overline{C}.\ \overline{D} + \overline{A}.\ B.\ \overline{C}.\ D + A.\ B.\ \overline{C}.\ \overline{D} + A.\ B.\ \overline{C}.\ D + A.\ \overline{B}.\ \overline{C}.\ \overline{D} + A.\ \overline{B}.\ \overline{C}.\ D}$$

CD \ AB	00	01	11	10
00				
01				
11				
10				

[2]

(b) Draw loop(s) around appropriate group(s) in the K-map to produce an optimal sum-of-products. [2]

(c) Write the Boolean expression from your answer to **part (b)** as a simplified sum-of-products. Use Boolean algebra to give your answer in its simplest form.

Simplified sum-of-products

Z = ...

Simplest form

Z = ...

[3]

8 Virtual memory, paging and segmentation are used in memory management.

(a) Explain what is meant by **virtual memory**.

....

....

....

....

....

.... [3]

(b) State **one** difference between paging and segmentation in the way memory is divided.

....

.... [1]

9 Deep learning is used in Artificial Intelligence (AI).

(a) Describe what is meant by **deep learning**.

....

....

....

.... [2]

(b) Outline the reasons for using deep learning.

....

....

....

.... [2]

10 Reduced Instruction Set Computers (RISC) and Complex Instruction Set Computers (CISC) are two types of processor.

(a) Tick (✓) **one** box in each row to show if the statement applies to RISC or CISC processors.

Statement	RISC	CISC
uses a smaller instruction set		
uses single-cycle instructions and limited addressing modes		
uses fewer general-purpose registers		
uses both hardwired and micro-coded control unit		
uses a system where cache is split between data and instructions		

[2]

(b) Describe the process of pipelining during the fetch-execute cycle in RISC processors.

....

....

....

....

....

....

....

.... [4]

11 (a) Define these Object-Oriented Programming (OOP) terms:

Instance ....

....

Inheritance ....

....

Polymorphism ....

....

[3]

(b) In OOP, a class contains attributes and methods.

Complete the pseudocode for the class `Car` to enable objects to be created. The class needs to include:

- string attributes to store the make, model, body type and fuel type
- an integer attribute to store the number of cars of that type built.

The attributes must be available only through the methods of the class.

```
CLASS ....................................

    PRIVATE Make : STRING

    PRIVATE ...............................................................

    .......................................................................

    .......................................................................

    .......................................................................

    PUBLIC PROCEDURE New(CarMake : STRING, .................................................. ,

                        .....................................................)

        Make ← .....................................

        Model ← .....................................

        BodyType ← CarBodyType

        Fuel ← ""

        NumberBuilt ← 0

    ENDPROCEDURE

    GetFuel()

    GetNumberBuilt()

....................................
```

[5]

12 (a) The array `Names[0:99]` is in alphabetical order.

Complete this pseudocode binary search algorithm.

```
Lower ← 0

..........................................................................................
Mid ← 0
Exit ← FALSE
OUTPUT "Enter the name to be found "
INPUT Target
REPEAT

  .......................................................................................... THEN

     OUTPUT Target, " does not exist"
     Exit ← TRUE
  ENDIF
  Mid ← Lower + (Upper - Lower + 1) DIV 2
  IF Names[Mid] < Target THEN

     Lower ← ..........................................................................................
  ENDIF
  IF Names[Mid] > Target THEN

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

  .......................................................................................... THEN
     OUTPUT Target, " was found at location ", Mid
     Exit ← TRUE
  ENDIF

..........................................................................................
```

[6]

(b) Big O notation is used to classify efficiency of algorithms.

The Big O notation for time complexity in a binary search is O(log n).

(i) State the Big O notation for time complexity of a linear search.

.... [1]

(ii) Describe the meaning of O(log n) as it applies to a binary search algorithm.

....

....

....

.... [2]

BLANK PAGE

Cambridge Assessment International Education

Cambridge International AS & A Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

1612799207

COMPUTER SCIENCE **9618/31**

Paper 3 Advanced Theory **October/November 2023**

1 hour 30 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- You may use an HB pencil for any diagrams, graphs or rough working.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

DC (PQ/CB) 318339/3

1 Real numbers are stored in a computer using floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and exponent.

(a) Write the normalised floating-point representation of +65.25 in this system.

Show your working.

Mantissa **Exponent**

Working ....

....

....

....

....

.... [3]

(b) Explain the problem that will occur in storing the normalised floating-point representation of +65.20 in this system.

....

....

....

.... [2]

2 **(a)** Draw **one** line to connect **each** protocol to its most appropriate use.

Protocol	**Use**
HTTP	to provide peer-to-peer file sharing
BitTorrent	when retrieving email messages from a mail server over a TCP/IP connection
SMTP	when transmitting hypertext documents
IMAP	to map MAC addresses onto IP addresses
	when sending email messages towards the intended destination

[4]

(b) Outline the purpose of the **Link layer** in the TCP / IP protocol suite.

....

....

....

.... [2]

3 Describe what is meant by **enumerated** and **pointer** data types.

Enumerated ....

....

....

....

Pointer ....

....

....

....

[4]

4 **(a)** Describe sequential and random methods of file organisation.

Sequential file organisation ....

....

....

....

....

Random file organisation ....

....

....

....

....

[4]

(b) Outline the process of sequential access for serial and sequential files.

....

....

....

.... [2]

5 Describe the features of SISD and MIMD computer architectures.

SISD ....

....

....

....

MIMD ....

....

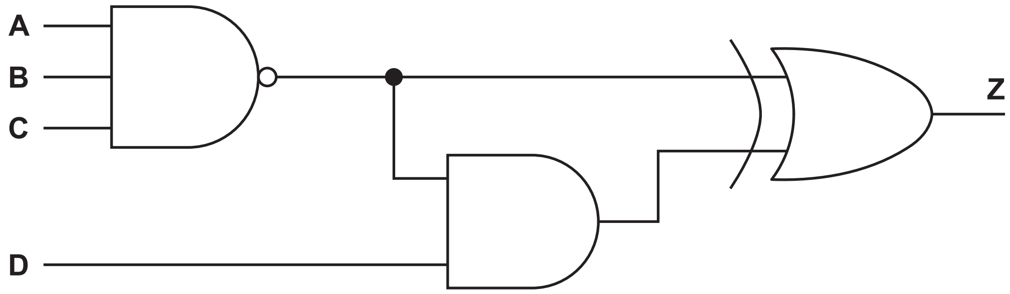
....

....

[4]

BLANK PAGE

6 This diagram represents a logic circuit.



(a) Complete the truth table for the given logic circuit.

A	B	C	D	Working space	Z
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

[3]

(b) Simplify the given Boolean expression using Boolean algebra.
Show your working.

$$\mathbf{Y = \overline{A}.\overline{B}.\overline{C}.\overline{D} + \overline{A}.\overline{B}.C.\overline{D} + \overline{A}.B.\overline{C}.\overline{D} + \overline{A}.B.C.\overline{D}}$$

....

....

....

....

....

.... [3]

7 **(a)** A student buys a new computer.

State **one** benefit to the student of a user interface **and** give an example.

Benefit ....

....

....

Example ....

....
[2]

(b) Two of the process states are the running state and the ready state.

Identify **one other** process state.

.... [1]

(c) Outline conditions under which a process could change from the running state to the ready state.

....

....

....

.... [2]

8 **(a)** A pseudocode algorithm finds a customer account record in a random file and outputs it. The records are stored using the user-defined data type `TAccount`.

```
TYPE TAccount
    DECLARE AccountNumber : INTEGER
    DECLARE LastName : STRING
    DECLARE FirstName : STRING
    DECLARE Address : STRING
    DECLARE ContactNumber : STRING
ENDTYPE
```

Complete the file handling pseudocode.

The function `Hash()` takes the customer account number as a parameter, calculates and returns the hash value.

```
DECLARE Customer : TAccount
DECLARE Location : INTEGER
DECLARE AccountFile : STRING

.............................................................................................. ← "AccountRecords.dat"

............................................................ AccountFile ........................................................
OUTPUT "Please enter an account number"
INPUT Customer.AccountNumber

Location ← Hash(..............................................................................................)

SEEK ........................................................................................................... , Location

........................................................... AccountFile, ........................................................
OUTPUT Customer              // output customer record
CLOSEFILE AccountFile
```

[5]

(b) Define the term **exception handling**.

....

.... [1]

(c) State **two** possible causes of an exception.

....

.... [2]

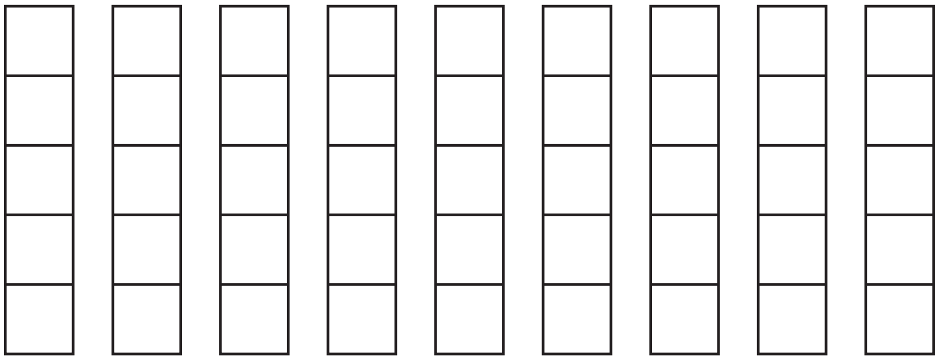
9 **(a)** **(i)** Write the infix expression for this Reverse Polish Notation (RPN) expression:

```
5 2 − 5 4 + * 9 /
```

....

.... [2]

(ii) Show how the contents of the following stack will change as the RPN expression in part **(a)(i)** is evaluated.



[4]

(b) Explain how a stack can be used to evaluate RPN expressions.

....

....

....

....

....

.... [3]

10 A stack is to be set up using the information in the table.

Identifier	Data type	Description
`BasePointer`	`INTEGER`	points to the bottom of the stack
`TopPointer`	`INTEGER`	points to the top of the stack
`Stack`	`REAL`	1D array to implement the stack

A constant, with identifier `Capacity`, limits the size of the stack to 25 items.

(a) Write the **pseudocode** for the required declarations.

....

....

....

....

....

.... [3]

(b) Complete the pseudocode function `Pop()` to pop an item from `Stack`.

```
// popping an item from the stack

FUNCTION Pop() ..........................................................................
   DECLARE Item : REAL
   Item ← 0

   .......................................................................... BasePointer THEN

      Item ← ..........................................................................

      TopPointer ← ..........................................................................
   ELSE
      OUTPUT "The stack is empty - error"
   ENDIF

   ..........................................................................
ENDFUNCTION
```

[5]

(c) Compare and contrast the queue and stack Abstract Data Types (ADT).

....

....

....

....

....

.... [2]

11 A **declarative** programming language is used to represent subjects that students can choose to study.

Students must choose two subjects.

```
01 subject(mathematics).
02 subject(physics).
03 subject(chemistry).
04 subject(computer_science).
05 subject(geography).
06 subject(history).
07 subject(english).
08 subject(biology).
09 student(tomaz).
10 student(josephine).
11 student(elspeth).
12 student(nico).
13 student(teresa).
14 student(pietre).
15 choice1(tomaz, mathematics).
16 choice1(teresa, chemistry).
17 choice1(pietre, mathematics).
18 choice1(nico, mathematics).
19 choice1(elspeth, chemistry).
20 choice2(tomaz, computer_science).
21 choice2(nico, geography).
```

These clauses have the meanings:

Clause	Meaning
01	Mathematics is a subject.
09	Tomaz is a student.
15	Tomaz has chosen mathematics as his first choice.
20	Tomaz has chosen computer science as his second choice.

(a) Anthony is a student who would like to study history and geography.

Write additional clauses to represent this information.

22 ....

23 ....

24 ....
[3]

(b) Using the variable `X`, the goal:

```
choice1(X, chemistry)
```

returns

```
X = teresa, elspeth
```

Write the result returned by the goal:

```
choice1(X, mathematics)
```

X = ... [1]

(c) Students must choose two different subjects such that:

N may choose S, if N is a student and S is a subject and N has not chosen S as the first choice.

Write this as a rule.

`may_choose_subject(N, S)`

IF ....

....

....

.... [4]

12 Artificial neural networks have played a significant role in the development of machine learning.

Explain what is meant by the term **artificial neural network**.

....

....

....

....

....

....

....

.... [4]

BLANK PAGE

BLANK PAGE

Cambridge International Examinations
Cambridge International Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

7098878869

COMPUTER SCIENCE **9608/32**

Paper 3 Advanced Theory **October/November 2016**

1 hour 30 minutes

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 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.
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.

The maximum number of marks is 75.

This document consists of **12** printed pages.

DC (ST/JG) 116627/3

CAMBRIDGE
International Examinations

1 In a particular computer system, real numbers are stored using floating-point representation with:

- 8 bits for the mantissa
- 8 bits for the exponent
- two's complement form for both mantissa and exponent

(a) Calculate the floating point representation of +3.5 in this system. Show your working.

Mantissa

•							

Exponent

[3]

(b) Calculate the floating-point representation of –3.5 in this system. Show your working.

Mantissa

•							

Exponent

[3]

(c) Find the denary value for the following binary floating-point number. Show your working.

Mantissa

0 ●	1	1	1	0	0	0	0

Exponent

0	0	0	0	0	1	0	0

....

....

....

....

....

.... [3]

(d) (i) State whether the floating-point number given in **part (c)** is normalised or not normalised.

.... [1]

(ii) Justify your answer given in **part (d)(i)**.

....

.... [1]

(e) Give the binary two's complement pattern for the negative number with the largest magnitude.

Mantissa

●							

Exponent

[2]

2 There are four stages in the compilation of a program written in a high-level language.

(a) Four statements and four compilation stages are shown below.

Draw a line to link each statement to the correct compilation stage.

Statement	**Compilation stage**
This stage can improve the time taken to execute the statement: `x = y + 0`	Lexical analysis
This stage produces object code.	Syntax analysis
This stage makes use of tree data structures.	Code generation
This stage enters symbols in the symbol table.	Optimisation

[4]

(b) Write the Reverse Polish Notation (RPN) for the following expression.

`P + Q − R / S`

.... [2]

(c) An interpreter is executing a program. The program uses the variables `a`, `b`, `c` and `d`.

The program contains an expression written in infix form. The interpreter converts the infix expression to RPN. The RPN expression is:

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

The interpreter evaluates this RPN expression using a stack.

The current values of the variables are:

```
a = 2   b = 2   c = 1   d = 3
```

(i) Show the changing contents of the stack as the interpreter evaluates the expression.

The first entry on the stack has been done for you.

2								

[4]

(ii) Convert back to its original infix form, the RPN expression:

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

....

.... [2]

(iii) One advantage of using RPN is that the evaluation of an expression does not require rules of precedence.

Explain this statement.

....

....

....

.... [2]

3 A computer operating system (OS) uses paging for memory management.

In paging:

- main memory is divided into equal-size blocks, called page frames
- each process that is executed is divided into blocks of the same size, called pages
- each process has a page table that is used to manage the pages of this process

The following table is the incomplete page table for a process, Y.

Page	Presence flag	Page frame address	Additional data
1	1	221	
2	1	222	
3	0	0	
4	0	0	
5	1	542	
6	0	0	
249	0	0	

(a) State **two** facts about Page 5.

1 ....

....

2 ....

.... [2]

(b) Process Y executes the last instruction in Page 5. This instruction is not a branch instruction.

(i) Explain the problem that now arises in the continued execution of process Y.

....

....

....

....

....

.... [2]

(ii) Explain how interrupts help to solve the problem that you explained in **part (b)(i)**.

....

....

....

....

....

.... [3]

(c) When the next instruction is not present in main memory, the OS must load its page into a page frame. If all page frames are currently in use, the OS overwrites the contents of a page frame with the required page.

The page that is to be replaced is determined by a page replacement algorithm.

One possible algorithm is to replace the page which has been in memory the shortest amount of time.

(i) Give the additional data that would need to be stored in the page table.

....

.... [1]

(ii) Complete the table entry below to show what happens when Page 6 is swapped into main memory. Include the data you have identified in **part (c)(i)** in the final column. Assume that Page 1 is the one to be replaced.

In the final column, give an example of the data you have identified in **part (c)(i)**.

Page	Presence flag	Page frame address	Additional data
6	…………	………………	………………………………

[3]

Process Y contains instructions that result in the execution of a loop, a very large number of times. All instructions within the loop are in Page 1.

The loop contains a call to a procedure whose instructions are all in Page 3.

All page frames are currently in use. Page 1 is the page that has been in memory for the shortest time.

(iii) Explain what happens to Page 1 and Page 3, each time the loop is executed.

....

....

....

....

....

.... [3]

(iv) Name the condition described in **part (c)(iii)**.

.... [1]

4 Both clients and servers use the Secure Socket Layer (SSL) protocol and its successor, the Transport Layer Security (TLS) protocol.

(a) **(i)** What is a protocol?

....

....

....

.... [2]

(ii) Name the client application used in this context.

.... [1]

(iii) Name the server used in this context.

.... [1]

(iv) Identify **two** problems that the SSL and TLS protocols can help to overcome.

1 ....

2 .... [2]

(b) Before any application data is transferred between the client and the server, a handshake process takes place. Part of this process is to agree the security parameters to be used.

Describe **two** of these security parameters.

1 ....

....

....

....

2 ....

....

....

.... [4]

(c) Name **two** applications of computer systems where it would be appropriate to use the SSL or TLS protocol. These applications should be different from the ones you named in **part (a)(ii)** and **part (a)(iii)**.

1 ....

....

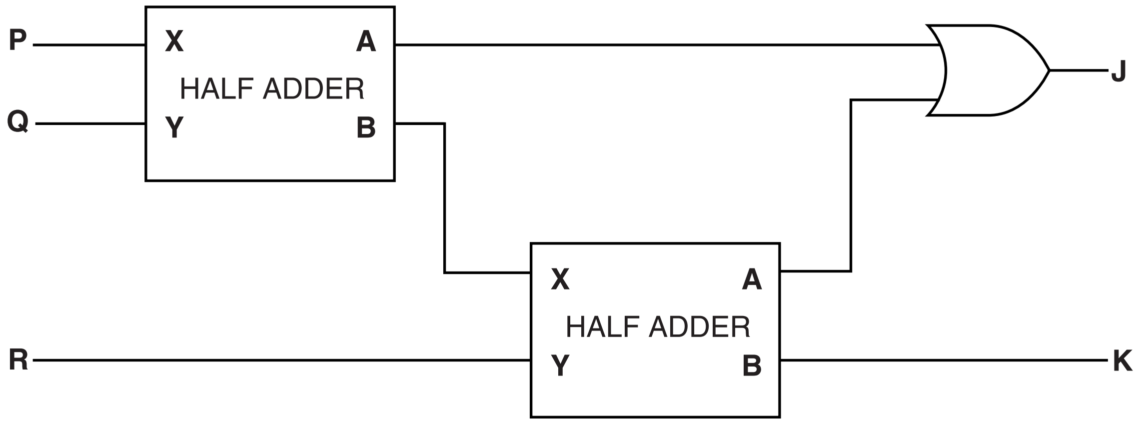
2 ....

.... [2]

5 **(a)** **(i)** A half adder is a logic circuit with the following truth table.

Input		**Output**	
X	**Y**	**A**	**B**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

The following logic circuit is constructed.



Complete the following truth table for this logic circuit.

Input			**Working space**	**Output**	
P	**Q**	**R**		**J**	**K**
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

[2]

(ii) State the name given to this logic circuit.

.... [1]

(iii) Name the labels usually given to **J** and **K**.

Label **J** ....

Label **K** ....

Explain why your answers are appropriate labels for these outputs.

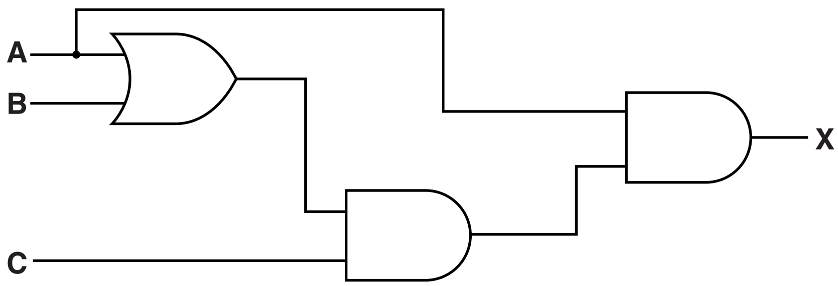
....

....

....

.... [4]

(b) (i) Write down the Boolean expression corresponding to the following logic circuit:



.... [2]

(ii) Use Boolean algebra to simplify the expression given in **part (b)(i)**.

Show your working.

....

....

....

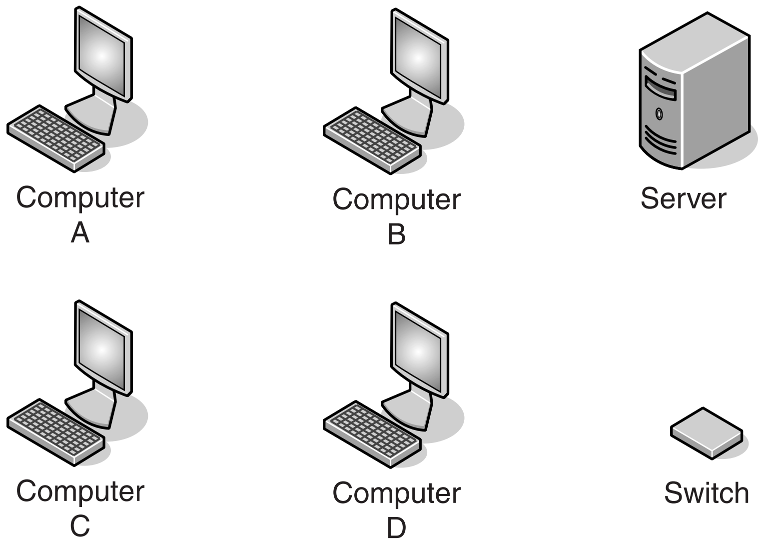
....

....

.... [4]

6 A Local Area Network (LAN) consists of four computers, one server and a switch. The LAN uses a star topology.

(a) Complete the diagram below to show how to connect the devices.



[2]

(b) The LAN uses packets to transfer data between devices.

Three statements are given below.

Tick (✓) to show whether each statement is true or false.

Statement	True	False
All packets must be routed via the server.		
Computer B can read a copy of the packet sent from the Server to Computer A.		
No collisions are possible.		

[3]

(c) In the same building as this star network, there is another star network.

(i) Name the device needed to connect the two networks together.

.... [1]

(ii) Explain how the device in **part (c)(i)** decides whether to transfer a packet from one network to the other.

....

....

....

.... [2]