PAPER 3

Question 1

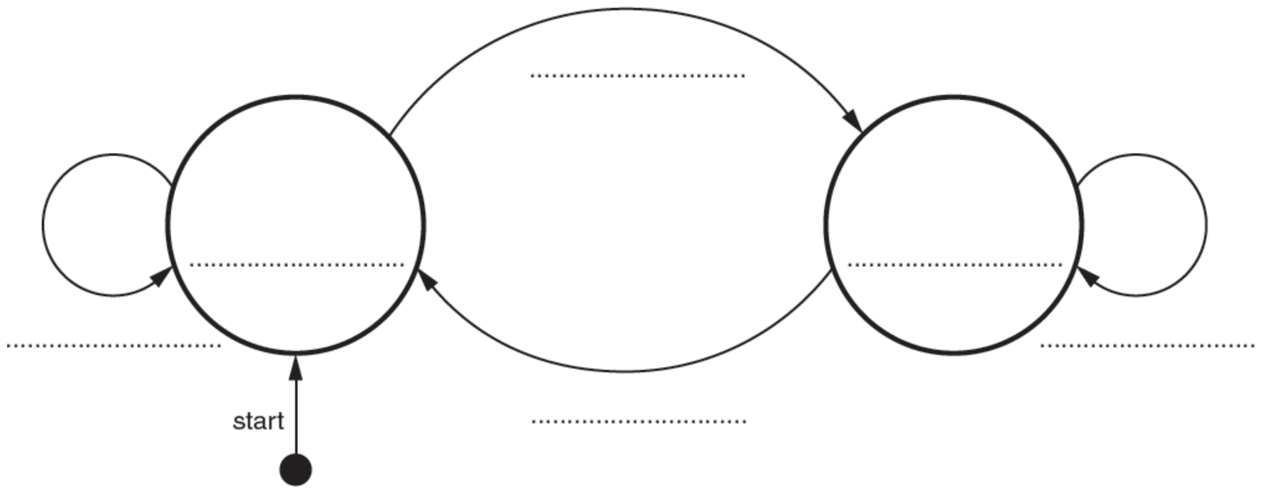
1 A turnstile is a gate which is in a locked state. To open it and pass through, a customer inserts a coin into a slot on the turnstile. The turnstile then unlocks and allows the customer to push the turnstile and pass through the gate.

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]

Question 2

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]

Question 3

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]

Question 4

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]

Question 5

6 A queue Abstract Data Type (ADT) has these associated operations:

- create queue
- add item to queue
- remove item from queue

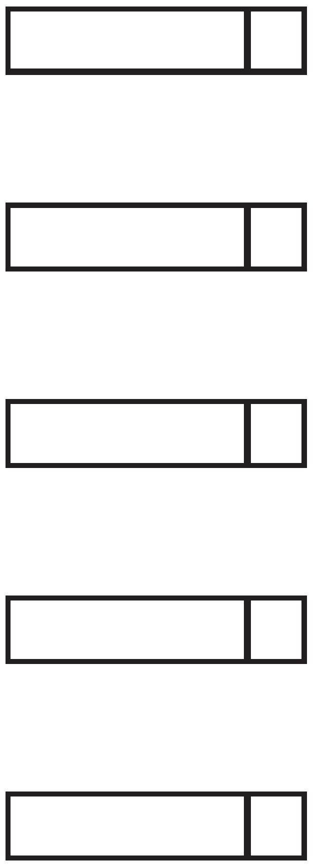
The queue ADT is to be implemented as a linked list of nodes.

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]

Question 6

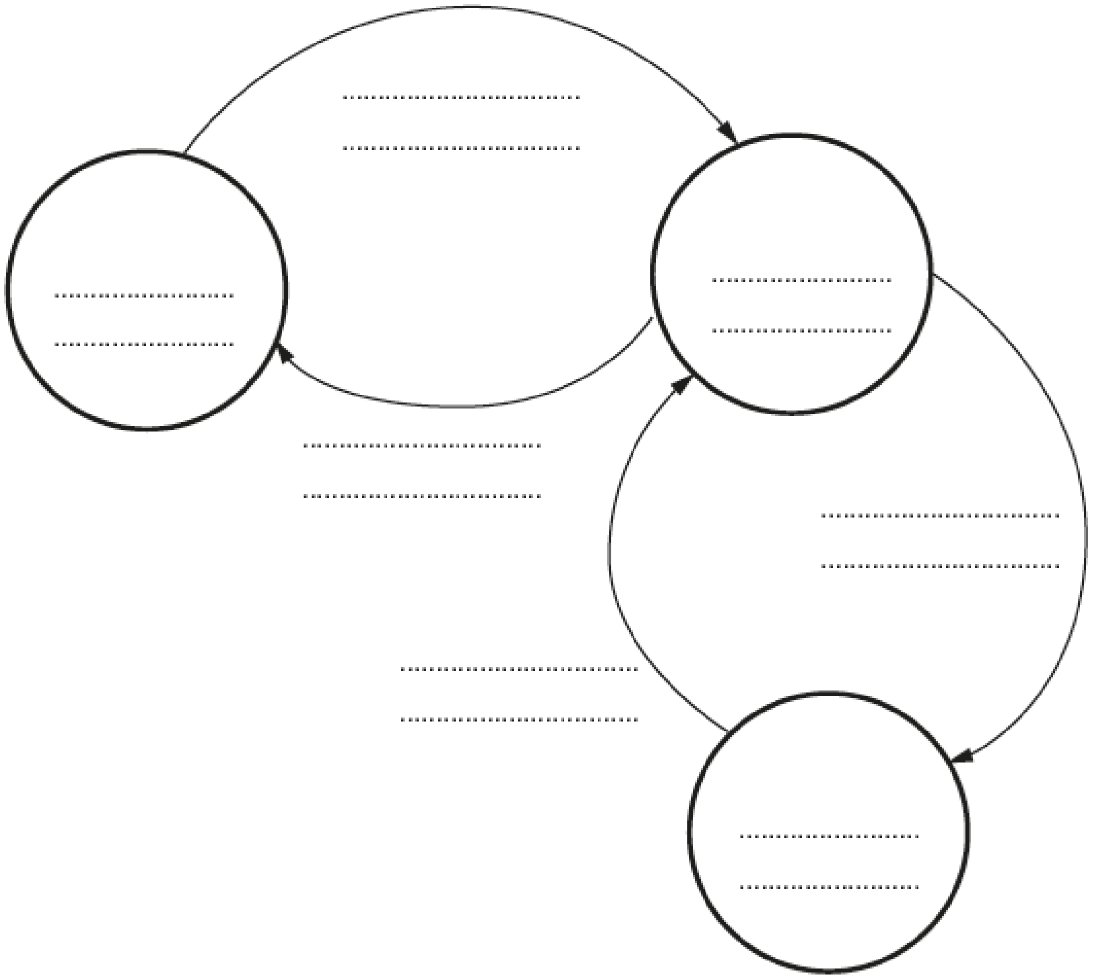
1 A greenhouse has a window that automatically opens and closes depending on the internal temperature.

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]

Question 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.

....

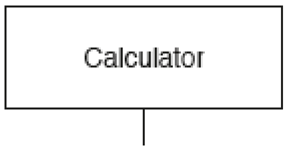
....[1]

(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.



Question 8

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]

Question 9

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).

(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		

Question 10

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]

Question 11

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]

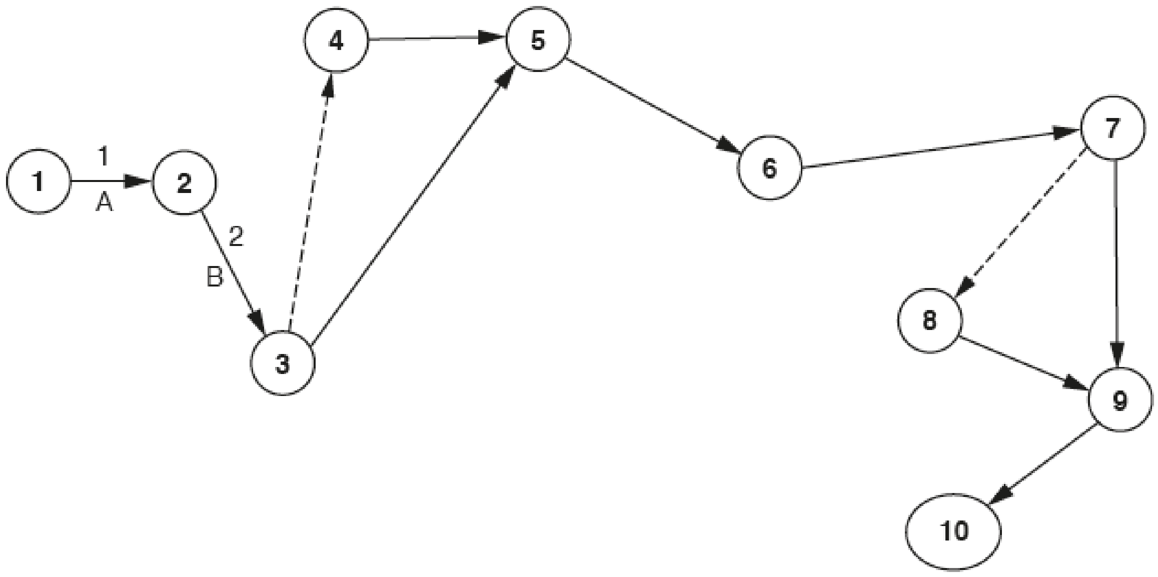
Question 12

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.

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]

Question 13

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:

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]

Question 14

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]

Question 15

(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		

(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]

Question 16

(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]

Question 17

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.

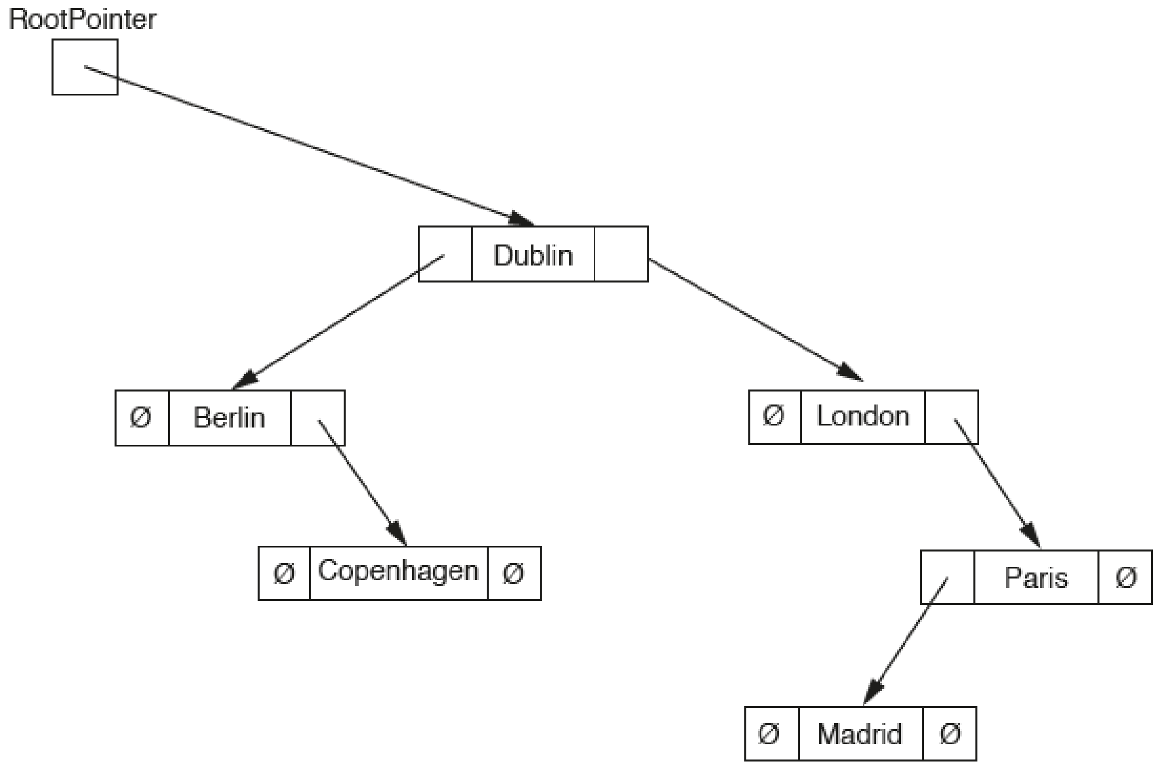
Explain the meaning of the term **null pointer**.

....

....[1]

(b) The following diagram shows an ordered binary tree after the following data have been added:

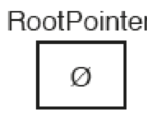
Dublin, London, Berlin, Paris, Madrid, Copenhagen



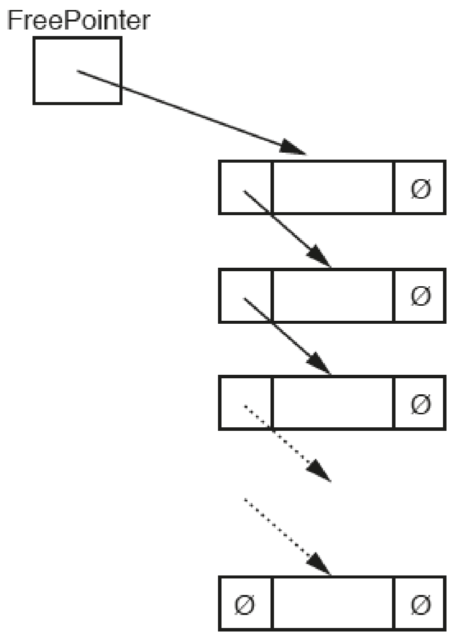
Another data item to be added is Athens.

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]

Question 18

(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]

Question 19

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]

Question 20

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]
```

Question 21

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]

Question 22

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]

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 23

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]

Question 24

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]

(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]

Question 25

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]

Question 26

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]

Question 27

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]

Question 28

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]

Question 29

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:

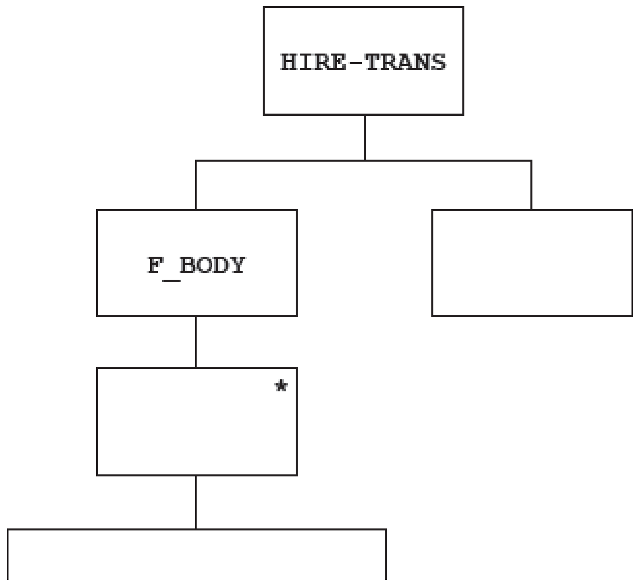
- car registration
- hire start date
- number of days hired

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.

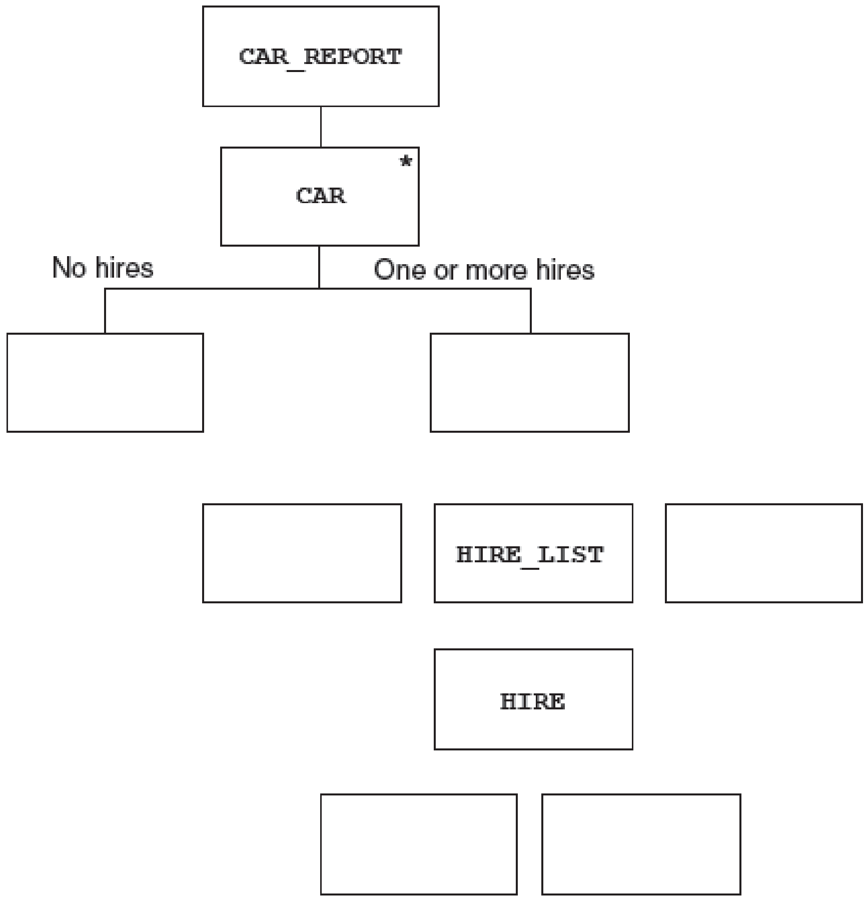
One report is `CAR_REPORT`. This displays all hire data for all cars.

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.



Question 30

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]

Question 31

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`.

....

....

....

....

....

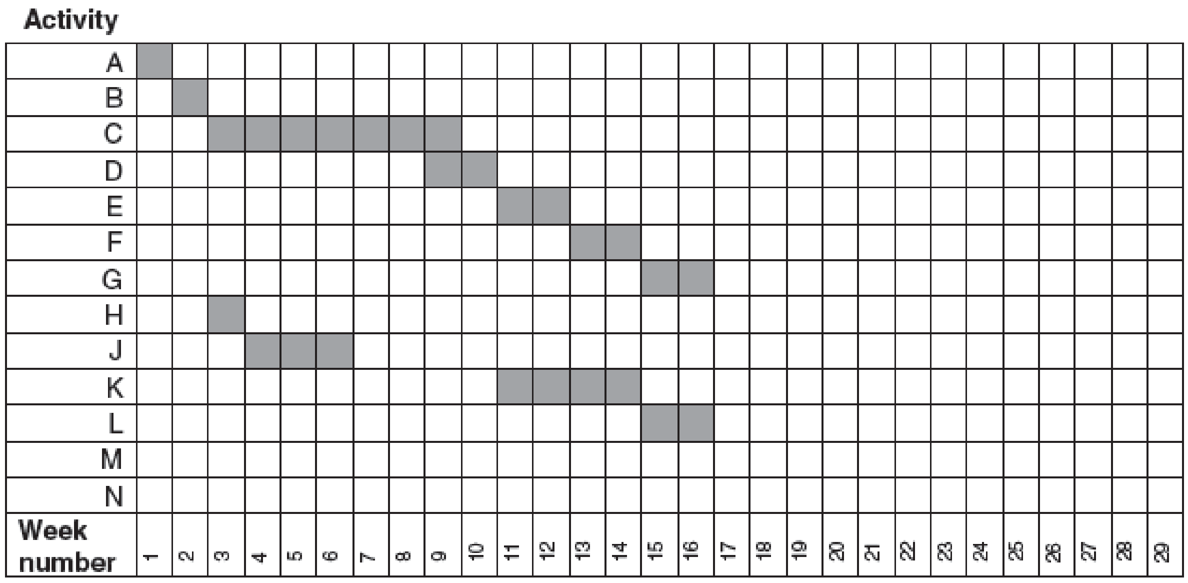
....[3]

Question 32

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																													
A	▨																												
B		▨																											
C			▨	▨	▨	▨	▨	▨	▨																				
D									▨	▨																			
E											▨	▨																	
F																													
G																													
H			▨																										
J																													
K																													
L																													
M																													
N																													
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	26	27	28	29

[9]

(ii) State the new estimated completion date.

Week number .... [1]

Question 33

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]

Question 34

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 35

(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]

Question 36

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]

Question 37

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 38

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:

```
10 // generate record address

20 NewAddress ← .................................................................................................

30 // move pointer to the disk address for the record

40 SEEK ........................................................................................................

50 PUTRECORD "MembershipFile", ..........................................................................
```

[4]

(ii) Before records can be saved to the file `MembershipFile`, the file needs to be opened.

Complete the pseudocode.

```
01 TRY

02    OPENFILE ......................................................................... FOR RANDOM

03 EXCEPT

04    ..........................................................................................................

05 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]