EXAM PAPERS PRACTICE

# 4.1 Abstraction and automation part 2

Name: ______________________

Class: ______________________

Date: ______________________

| | |
|---|---|
| Time: | **361 minutes** |
| Marks: | **241 marks** |
| Comments: | |

## Q1.

(a) Dry run the following algorithm by completing the trace table.

```
x ← 5
y ← 3
Result ← 1
REPEAT
    Result ← Result * x
    y ← y - 1
UNTIL y=0
```

| **x** | **y** | **Result** |
|---|---|---|
| 5 | 3 | 1 |
| | | |
| | | |
| | | |

**(7)**

(b) What is the purpose of this algorithm?

______________________________________________________________

**(1)**

**(Total 8 marks)**

## Q2.

Processes are rated in priority according to their expected running times. Processes with the shortest running times are given top priority. A process joining the list will be placed immediately after all other processes of equal or higher priority. The name of the process indicates the order in which it joined the list. Process6 joined the list before Process7.

The table below contains for each process its name, the memory address of its process control block, its expected running time and a pointer to the position in the table of the next process to be executed.

Processes are entered in the table wherever there is a free slot.

(i) Complete the pointer column after the following processes have been placed in the table:

HeadPointer = 6

| Position | Name | Running Time | Address | Pointer |
|---|---|---|---|---|
| 1 | Process6 | 7 | 01400 | |
| 2 | | | | |

| | | | | |
|---|---|---|---|---|
| 3 | Process7 | 17 | 01700 | |
| 4 | Process2 | 17 | 02300 | |
| 5 | Process9 | 45 | 04100 | |
| 6 | Process5 | 2 | 01200 | |
| 7 | | | | |
| 8 | Process19 | 5 | 01900 | |

**(3)**

(ii) The scheduler program is written in a high level language. Name and describe a suitable data structure for this table.

**(2)**

(iii) The Computer System Manager may wish to view the current order in which the runnable processes are predicted to run. Write an algorithm that will print the process names in runnable order.

**(4)**

(iv) Name another list of processes that must be maintained by the operating system. Explain why the processes are in this list.

**(2)**

**(Total 11 marks)**

## Q3.

The following algorithm uses an array **Values** containing four numbers.

|   | Values |
|---|---|
| 1 | 24 |
| 2 | 13 |
| 3 | 57 |
| 4 | 45 |

```
Result ← 0

Index ← 0
Repeat

  Index ← Index + 1
  If Result < Values[Index]

     Then Result ← Values[Index]
  EndIf
Until Index = 4
```

(a) Dry run this algorithm by using the trace table below.

| **Result** | **Index** |
|---|---|
| 0 | 0 |
|  |  |
|  |  |
|  |  |
|  |  |

**(5)**

(b) What is the purpose of this algorithm?

___________________________________________________________________

**(1)**

**(Total 6 marks)**

## Q4.

A recursively-defined procedure X with three integer parameters is defined below.
*x* DIV *y* calculates how many times *y* divides exactly into *x*. For example 7 DIV 3 = 2.

```
Procedure X (E,L,H)
 If L > H
    Then Print 'False'

    Else M ← (L+H) DIV 2
```

```
       If E = List[M]
         Then Print ‘True’
         Else
            If E < List[M]
              Then X (E,L,M-1)
              Else X (E,M+1,H)
            Endif
       Endif
  Endif
EndProc
```

(a) What is meant by recursively-defined?

_______________________________________________________________ **(1)**

(b) (i) Using the table below, dry-run the procedure call X (6502, 1, 11) applied to the integer array *List* containing the following elements:

| **Index** | **List** |
|---|---|
| 1 | 1234 |
| 2 | 1789 |
| 3 | 3125 |
| 4 | 4789 |
| 5 | 5006 |
| 6 | 5789 |
| 7 | 6502 |
| 8 | 7411 |
| 9 | 8407 |
| 10 | 8971 |
| 11 | 9053 |

| **E** | **L** | **H** | **M** | **List[M]** | **Printed Output** |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

**(7)**

(ii) What process does procedure X describe?

______________________________________________________

**(2)**

**(Total 10 marks)**

## Q5.

(a) Name **two** different coding systems used to represent characters in a computer system.

1. ______________________________________________________

2. ______________________________________________________

**(2)**

(b) In one coding system the character digits are assigned the decimal number codes 48 to 57.

```
 The operators DIV and MOD perform integer arithmetic.
x DIV y calculates how many times y divides into x,
     for example 7 DIV 3 = 2.
x MOD y calculates the remainder that results after the division,
     for example 7 MOD 3 = 1.
```

(i) The following algorithm uses an array Result. Dry run this algorithm by completing the trace table below.

```
x ← 835
Index ← 0
Repeat
  Index ← Index + 1
  Result[Index] ← x MOD 10 + 48
  x ← x DIV 10
UNTIL x = 0
```

| x | Index | Result | | |
|---|---|---|---|---|
| | | [3] | [2] | [1] |
| 835 | 0 | – | – | – |
| 83 | 1 | – | – | 53 |
| | | | | |
| | | | | |

**(6)**

(ii) Explain the purpose of the algorithm.

______________________________________________________

**(1)**

**(Total 9 marks)**

## Q6.

A *recursively-defined* procedure B, which takes an integer as its single parameter, is defined below. The operators DIV and MOD perform integer arithmetic.
x DIV y calculates how many times y divides exactly into x. For example 7 DIV 3 = 2
x MOD y calculates the remainder that results. For example 7 MOD 3 = 1.

```
Procedure B (Number)
  If (Number = 0) OR (Number = 1)
    Then Print (Number)
    Else
     B (Number DIV 2)
     Print (Number MOD 2)
  EndIf
EndProcedure
```

(a) What is meant by recursively-defined?

______________________________________________________________________

______________________________________________________________________

**(1)**

(b) Why is a stack necessary to execute procedure B recursively?

______________________________________________________________________

______________________________________________________________________

**(1)**

(c) Dry run the procedure call *B(53)* showing clearly the values of the parameter and the printed output for the six calls of *B*.

| Call Number | Parameter |
|---|---|
| 1 | 53 |
| 2 | 26 |
| 3 | 13 |
| 4 | |
| 5 | |
| 6 | |

Printed Output: ______________________________________________________

**(6)**

(d) What process does procedure B describe? __________________________________

**(1)**

**(Total 9 marks)**

## Q7.

The following code is part of a high level program to manipulate text:

```
Var SI: String
Var S2: String
Var Ptr: Integer
Var L: String
S1 := “PAT”
S2 := “”                        {“” denotes an empty string}
For Ptr := 1 To 3 Do
    L := Copy (S1, Ptr)
    S2 := Concat (L, S2)
EndFor
If S1 = S2
    Then Print (‘True’)
    Else Print (‘False’)
EndIf
```

(a) By copying **one** relevant line from the above code, give an example of:

(i) variable declaration____________________________________________ **(1)**

(ii) selection statement____________________________________________ **(1)**

(iii) iteration____________________________________________________ **(1)**

(b) The built-in subroutines **Copy, Concat** and **Print** have been used in the above code.

**Copy (S, n)** returns the nth character of string S
example: Copy (“ABCDE”,2) returns the character “B”.

**Concat (S1, S2)** concatenates the two strings S1 and S2 and returns a single string
example: Concat (“ABCD”,”EF”) returns the string “ABCDEF”.

**Print (S)** prints the string S as output.

Subroutines are either *functions* or *procedures*. Indicate, by ticking the correct boxes, what each of the above subroutines is.

| **subroutine** | **procedure** | **function** |
|---|---|---|
| copy | | |
| concat | | |
| print | | |

**(3)**

(c) Dry run the above code by completing the table below.

| **S1** | **Ptr** | **L** | **S2** |
|---|---|---|---|
| **“PAT”** | | | “” |
| | 1 | “P” | “P” |

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| Printed Output: | | | |

**(8)**
**(Total 14 marks)**

## Q8.

The list **Days** contains the following representation of the days of the week.

[Sun, Mon, Tue, Wed, Thu, Fri, Sat]

The table below shows some functions which take a list as their single argument and return a result which is either an element of a list, another list, or a Boolean value.

| |
|---|
| **Head(list)** - returns the element at the head of **list** (e.g. Head(Days)→ Sun) if **list** is non empty otherwise it reports an error. |
| **Tail(list)** - returns a new list containing all but the first element of the original list (e.g. Tail(Days) → [Mon, Tue, Wed, Thu, Fri, Sat]) if **list** is non-empty otherwise it reports an error. |
| **Empty(list)** - returns True if **list** is the empty list or False otherwise. The empty list is denoted by[] |

(a) What result is returned when the following function calls are made?

(i) Head (Tail(Days))__________________________________________

**(1)**

(ii) Tail ([(Head(Days)])_________________________________________

**(1)**

(iii) Empty(Tail(Tail(Tail(Days))))___________________________________

**(1)**

(b) Explain why it is faster to access these elements if the above data is stored as a one dimensional array.

____________________________________________________________

____________________________________________________________

**(2)**
**(Total 5 marks)**

## Q9.

The algorithm below re-arranges numbers stored in a one-dimensional array called **List**. **Ptr** is an integer variable used as an index (subscript) which identifies elements within **List**. **Temp** is a variable, which is used as a temporary store for numbers from **List**.

```
Ptr ← I
```

```
While Ptr < 10 Do
  If List [Ptr] > List [Ptr+ 1] Then
     Temp ← List [Ptr]
     List [Ptr] ← List [Ptr+l]
     List [Ptr+l] ← Temp
  Endif
  Ptr ← Ptr+ 1
Endwhile
```

(a) Dry-run the algorithm by completing the table below,

It is only necessary to show those numbers which change at a particular step.

| Ptr | Temp | List | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] | [10] |
| | | 43 | 25 | 37 | 81 | 18 | 70 | 64 | 96 | 52 | 4 |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |

**(7)**

(b) What will happen when **Ptr**= 10?

______________________________________________________________

**(1)**

(c) If the whole algorithm is now applied to this rearranged list, what will be the values of:

(i) List[1] ______________________________________________

(ii) List[9] ______________________________________________

(iii) List[10]? ______________________________________________

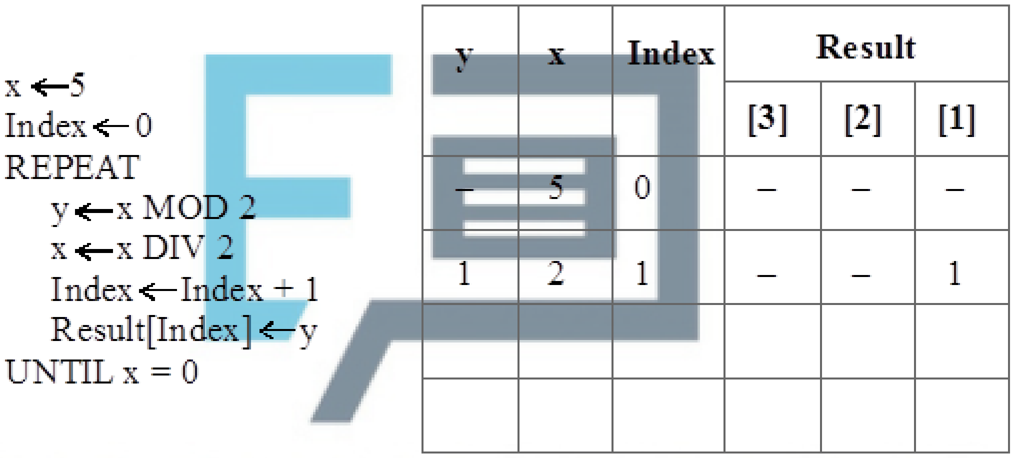
**(3)**

**(Total 11 marks)**

## Q10.

The operators DIV and MOD perform integer arithmetic.
x DIV y calculates how many times y divides into x, for example 7 DIV 3 = 2.
x MOD y calculates the remainder that results after the division, for example 7 MOD 3 = 1.

(a) The following algorithm uses an array Result. Dry run this algorithm by completing the trace table below.



```
x ← 5
Index ← 0
REPEAT
    y ← x MOD 2
    x ← x DIV 2
    Index ← Index + 1
    Result[Index] ← y
UNTIL x = 0
```

| y | x | Index | Result | | |
|---|---|---|---|---|---|
| | | | [3] | [2] | [1] |
| – | 5 | 0 | – | – | – |
| 1 | 2 | 1 | – | – | 1 |
| | | | | | |
| | | | | | |

**(6)**

(b) What is the purpose of this algorithm?

______________________________________________

**(1)**

**(Total 7 marks)**

## Q11.

A procedure to process an array of numbers is defined as follows.

```
Procedure P(Number)
  Repeat

      X ← StartofArray

      Flag ← False
      Repeat
        If Number(X) > Number (X+ 1)
           Then
             Begin
```

```
                Temp ← Number(X)

                Number (X) ← Number (X+ 1)

                Number(X+I) ← Temp

                Flag ← True
              End

            X ← X+l
         Until EndofArray
       Until Flag = False
Endproc
```

The array number, containing 17, 11, 21, 9, 23, 15, is to be processed by this procedure.

(a) List the array after the outer Repeat loop has been executed once.

**(2)**

(b) What algorithm does the procedure P describe?

**(1)**

(c) What is the purpose of Flag in this procedure?

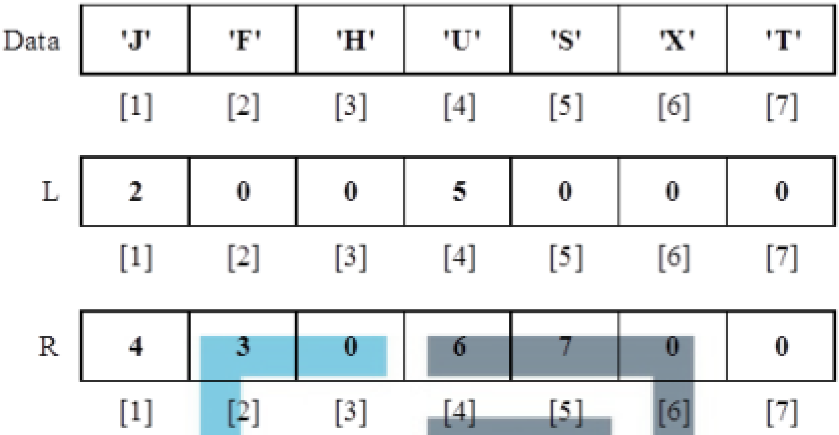
**(1)**
**(Total 4 marks)**

## Q12.

(a) The series of characters J, F, H, U, S, X, T are to be entered into a binary search tree in the order given. Draw a diagram to show how these values will be stored.

**(4)**

(b) The following data are held in arrays Data, L and R:



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Data | 'J' | 'F' | 'H' | 'U' | 'S' | 'X' | 'T' |
| | [1] | [2] | [3] | [4] | [5] | [6] | [7] |
| L | 2 | 0 | 0 | 5 | 0 | 0 | 0 |
| | [1] | [2] | [3] | [4] | [5] | [6] | [7] |
| R | 4 | 3 | 0 | 6 | 7 | 0 | 0 |
| | [1] | [2] | [3] | [4] | [5] | [6] | [7] |

Using the arrays above, dry-run the following pseudo-code by completing the trace table opposite:

```
Item ← 'T'

Ptr ← 1
WHILE Data[Ptr] < > Item DO
  PRINT Data[Ptr]
  IF Data[Ptr] > Item
     THEN Ptr ← L[Ptr]

     ELSE Ptr ← R[Ptr]
  ENDIF
ENDWHILE
PRINT Data[Ptr]
```

Trace Table:

| **Item** | **Ptr** | **Printed Output** |
|---|---|---|
| 'T' | 1 | 'J' |
| | | |
| | | |
| | | |

**(6)**
**(Total 10 marks)**

## Q13.

(a) A unique numerical code, occupying a single byte, is generated for each key pressed on a computer's keyboard.
What is meant by a byte?

___________________________________________________________________

**(1)**

(b) In one coding system, the character digits 0 to 9 are assigned the decimal number codes 48 to 57 and the letters A to Z the decimal number codes 65 to 90.
Which keys produce the following codes?

(i) 0100 0001___________________________________________________

**(1)**

(ii) 0011 1001___________________________________________________

**(1)**

(c) A number is entered at the keyboard as a sequence of character digits. This sequence is processed to convert the code representation into its decimal integer value using the following algorithm:

```
Number ← 0
While more character digits Do
     get next character digit
     and store its ASCII code in the variable Code

     Number ← Number * 10 + Code – 48
EndWhile
```

Complete the trace table for the sequence 7321.

| Code | Number |
|---|---|
| – | 0 |
| 55 | 7 |
| | |
| | |
| | |

**(6)**

**(Total 9 marks)**

## Q14.

(a) An example of an iteration in Pascal is:

FOR x : = 1 TO 10 DO writeln (‘Hello’);

In a high level programming language you are familiar with, using the correct syntax, give an example of:

(i) declaration; ______________________________________________

**(2)**

(ii) assignment; ______________________________________________

**(1)**

(iii) selection. ______________________________________________

______________________________________________

**(2)**

(b) A one-dimensional array q contains the following characters:

| q | |
|---|---|
| D | [5] |
| K | [4] |
| C | [3] |
| T | [2] |
| M | [1] |

(i) Dry run the following algorithm, recording your results in the diagram.

```
FOR pointer ← 1 to 5
     s[pointer] ← q[pointer]
END FOR
pointer1 ← 1
pointer2 ← 5
REPEAT
     q[pointer1] ←s[pointer2]
     pointer1 ← pointer1 + 1
     pointer2 ← pointer2 – 1
UNTIL pointer2 = 0
```

| q | | s | | q | |
|---|---|---|---|---|---|
| D | [5] | | [5] | | [5] |
| K | [4] | | [4] | | [4] |
| C | [3] | | [3] | | [3] |
| T | [2] | | [2] | | [2] |
| M | [1] | | [1] | | [1] |

**(10)**

(ii) What is the purpose of the above algorithm?

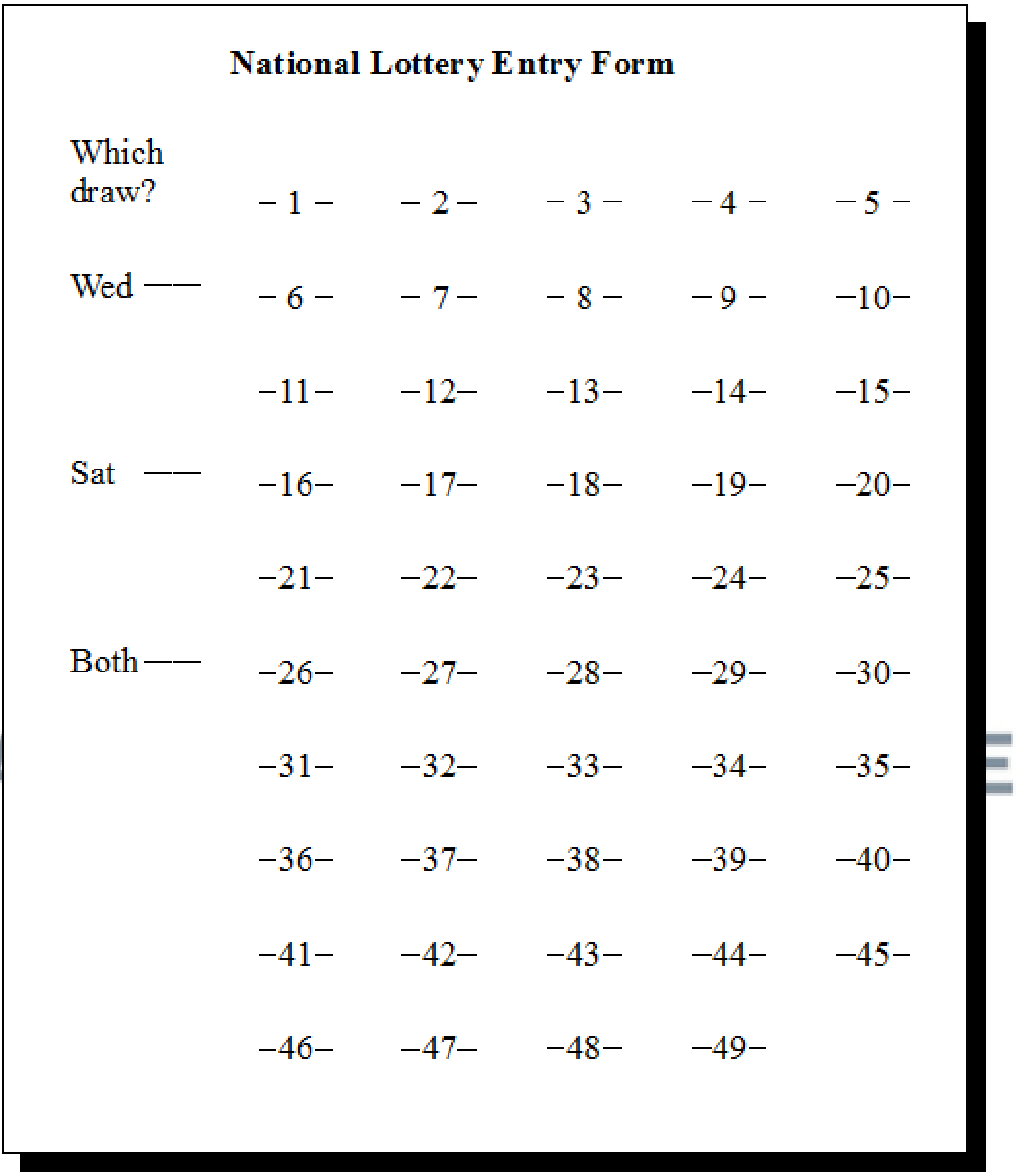
______________________________________________________________________

**(1)**

**(Total 16 marks)**

## Q15.

Players, in a national lottery, show their selection of different numbers by placing marks on an entry form similar to the one shown in **Figure 1.** The entry form is then inserted into a machine at the point of sale and the numbers are read.



**National Lottery Entry Form**

| | | | | | |
|---|---|---|---|---|---|
| Which draw? | –1– | –2– | –3– | –4– | –5– |
| Wed —— | –6– | –7– | –8– | –9– | –10– |
| | –11– | –12– | –13– | –14– | –15– |
| Sat —— | –16– | –17– | –18– | –19– | –20– |
| | –21– | –22– | –23– | –24– | –25– |
| Both —— | –26– | –27– | –28– | –29– | –30– |
| | –31– | –32– | –33– | –34– | –35– |
| | –36– | –37– | –38– | –39– | –40– |
| | –41– | –42– | –43– | –44– | –45– |
| | –46– | –47– | –48– | –49– | |

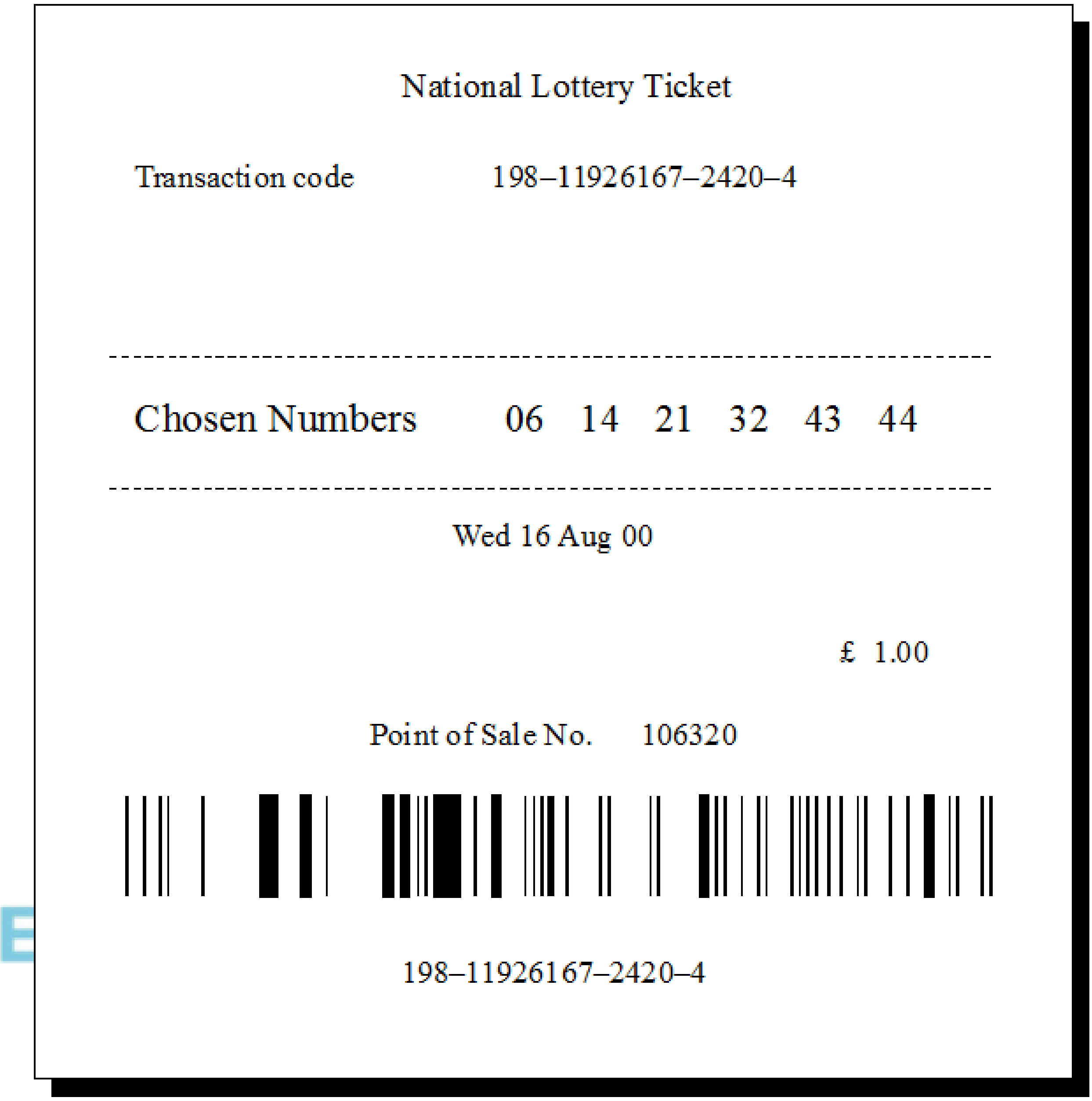
**Figure 1**

(a) Name the method being used to read the data.

______________________________________________________________________

**(1)**

The data are transmitted to a central computer which allocates a unique transaction code. This code is relayed back to the point of sale where a machine prints the chosen numbers and a transaction code onto the ticket similar to the one shown in **Figure 2.**



National Lottery Ticket

Transaction code 198–11926167–2420–4

Chosen Numbers 06 14 21 32 43 44

Wed 16 Aug 00

£ 1.00

Point of Sale No. 106320

198–11926167–2420–4

**Figure 2**

(b) Each transaction code includes a check digit. What is a check digit and why is it used?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(2)**

(c) Each transaction is recorded in a separate record. All transaction records for a particular lottery draw are stored in a single transaction file.

The transaction record includes the following fields:

Date of Purchase
Date of Draw
Point of Sale Identification Code
Transaction Code
Chosen Numbers

(i) What is meant by primary key?

______________________________________________________________

______________________________________________________________

**(1)**

(ii) Which of the above fields should be chosen as the primary key?

______________________________________________________________

**(1)**

(iii) What would be a suitable file organisation for the transaction file if it is required that the ticket(s) with the winning numbers is to be found? Justify your choice.

______________________________________________________________

______________________________________________________________

______________________________________________________________

______________________________________________________________

**(2)**

(iv) If individual records need to be accessed quickly what file organisation should be used? Justify your choice.

______________________________________________________________

______________________________________________________________

______________________________________________________________

**(2)**

(d) After a draw, some lottery prize-winners can check their tickets at any lottery point of sale machine. State the processing steps required by the lottery's computer system to check if the ticket is a winning ticket.

______________________________________________________________

______________________________________________________________

______________________________________________________________

**(4)**

**(Total 13 marks)**

## Q16.

The memory of a computer holds an array of records, each of which includes name, address and other information.

(a) What condition is necessary for the binary search (binary chop) process to work correctly?

**(2)**

(b) Describe this process to find the position in the array of the record containing a given name.

**(5)**

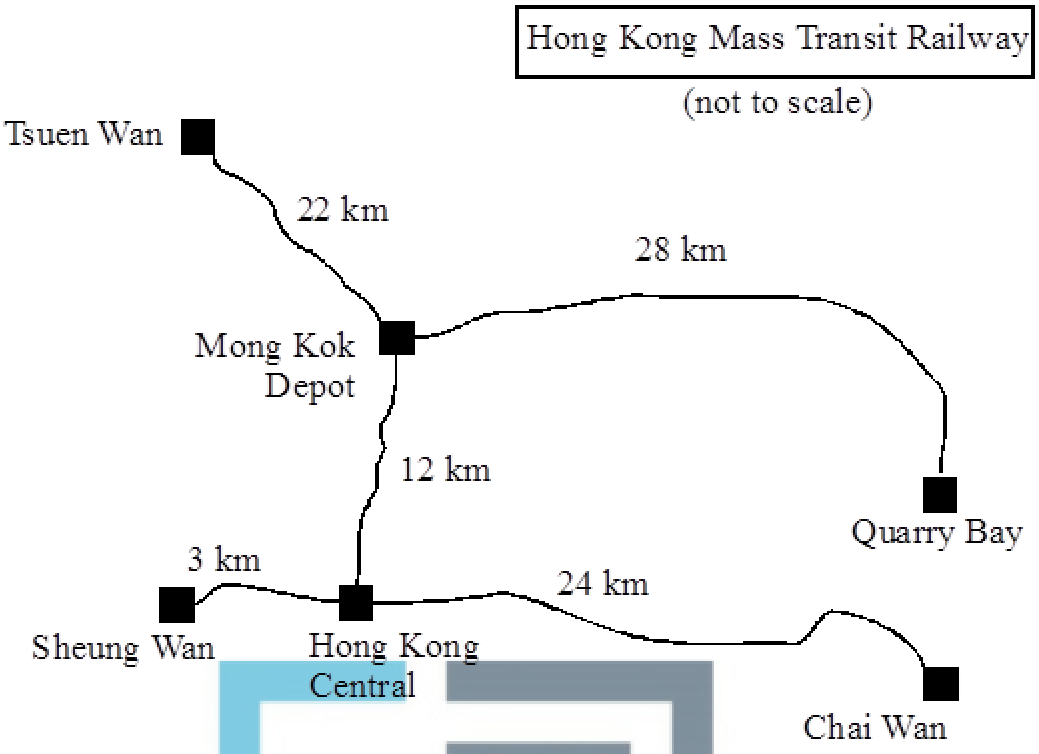
(c) Why is this search method normally faster than a linear search?

**(2)**

**(Total 9 marks)**

## Q17.

The plan below shows the layout of the Mass Transit Railway (MTR) in Hong Kong. The maintenance depot is at Mong Kok.



All the trains operate the same cycle (sequence) of journeys, given by the algorithm below. The algorithm is intended to ensure that:

1. trains are serviced as soon as possible after covering 135 km, and

2. each train will have travelled in both directions along each track at least once in the cycle.

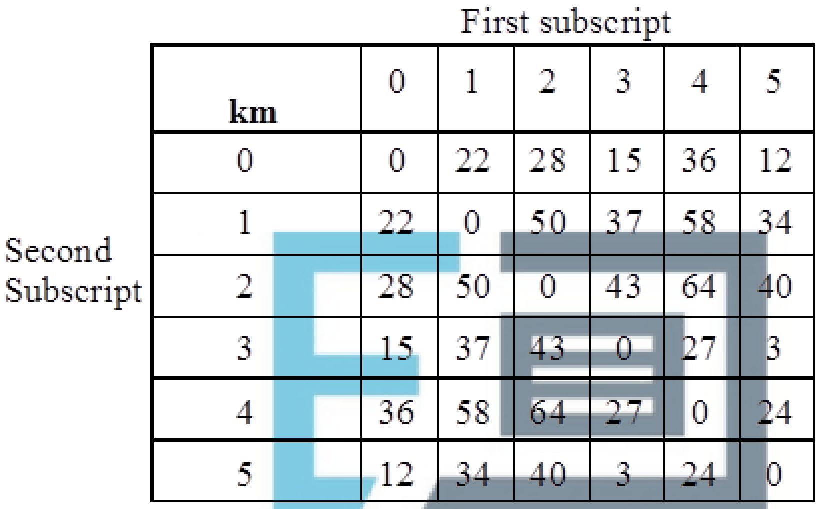
The algorithm relates to three arrays called *station*, *journey* and *km*. The contents of these arrays are shown below.

| Subscript | **Station** |
|---|---|
| 0 | Mong Kok depot (MK) |
| 1 | Tsuen Wan (TW) |
| 2 | Quarry Bay (QB) |
| 3 | Sheung Wan (SW) |
| 4 | Chai Wan (CW) |
| 5 | Hong Kong Central (HK) |

| Subscript | **Journey** |
|---|---|
| 0 | 3 |

| | |
|---|---|
| 1 | 4 |
| 2 | 3 |
| 3 | 1 |
| 4 | 5 |
| 5 | 2 |
| 6 | 3 |

The 6 × 6 two-dimensional array **km**, representing the distance between stations (in kilometres), contains



| | | First subscript | | | | | |
|---|---|---|---|---|---|---|---|
| | **km** | 0 | 1 | 2 | 3 | 4 | 5 |
| Second Subscript | 0 | 0 | 22 | 28 | 15 | 36 | 12 |
| | 1 | 22 | 0 | 50 | 37 | 58 | 34 |
| | 2 | 28 | 50 | 0 | 43 | 64 | 40 |
| | 3 | 15 | 37 | 43 | 0 | 27 | 3 |
| | 4 | 36 | 58 | 64 | 27 | 0 | 24 |
| | 5 | 12 | 34 | 40 | 3 | 24 | 0 |

The proposed algorithm is:

```
org:=0
last := 1
dest:= 3
maintain := FALSE
start := station[org]
finish := station[dest]
totalkm := km [org, dest]
org := dest
while(TRUE)
     n := 0
     repeat

          n := n + 1
          if (maintain = TRUE) then

              n := last
              totalkm := 0
              maintain := FALSE

          endif

          dest := journey [n]
          if (totalkm > 135) then

              dest := 0
```

```
               last := n
               maintain := TRUE

          endif

          start := station[org]
          finish := station[dest]
          totalkm := totalkm + km[org, dest]
          org := dest

     until n >= 6
endwhile
```

(a) What is the effect of the instructions **while**(TRUE) and **endwhile**?

________________________________________________________________________

________________________________________________________________________

**(1)**

(b) For each of the variables *maintain* and *n*, state with a reason what data type it should be.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

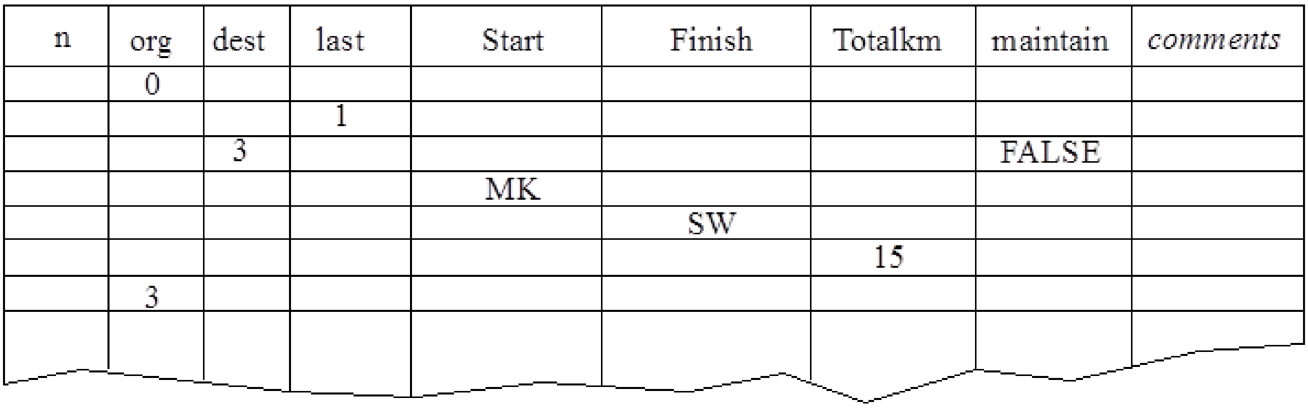
________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

**(4)**

(c) Copy and complete the trace table below, for one iteration of the outer (**while** .... **endwhile**) loop.



| n | org | dest | last | Start | Finish | Totalkm | maintain | *comments* |
|---|---|---|---|---|---|---|---|---|
| | 0 | | | | | | | |
| | | | 1 | | | | | |
| | | 3 | | | | | FALSE | |
| | | | | MK | | | | |
| | | | | | SW | | | |
| | | | | | | 15 | | |
| | 3 | | | | | | | |
| | | | | | | | | |

**(8)**

(d) An objective of the algorithm is that each train has travelled in both directions along every track at least once in the cycle. Using your trace table, state, with reasons, whether this objective has been achieved.

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

**(2)**
**(Total 15 marks)**

## Q18.

Data is held in a linked list. The array animals contain records with the content shown.

| Subscript | Data | Pointer |
|---|---|---|
| 1 | Elephant | |
| 2 | Deer | |
| 3 | Bear | |
| 4 | Rabbit | |
| 5 | Cow | |
| 6 | | |

(a) Give the values that would be needed in the pointer field of each non-empty record to produce a list in alphabetical order. A pointer value of zero indicates the end of the list.

___________________________________________________________________

___________________________________________________________________

**(2)**

(b) The variables *Start* and *Freestorage* are used to point to the start of the list and the next free space, respectively. What values should they contain?

___________________________________________________________________

___________________________________________________________________

**(2)**

(c) Describe the steps needed to add “Monkey” to the list.

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

___________________________________________________________________

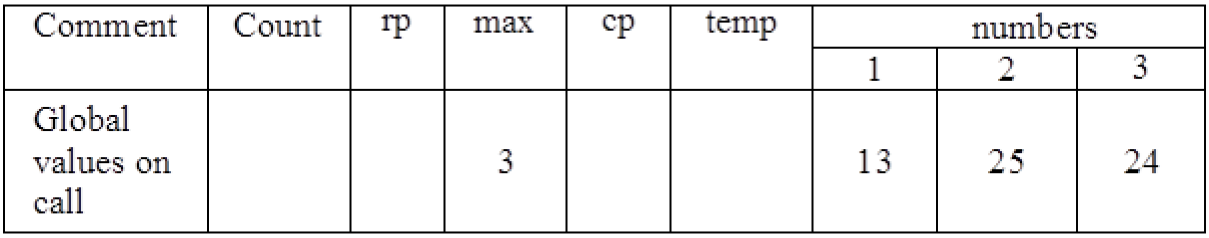
**(5)**
**(Total 9 marks)**

## Q19.

The algorithm below shows a procedure called sort.

```
//numbers is a global array of integers
// max is a global integer holding the number of values to be sorted

procedure sort

     integer: cp, rp, temp, count
     rp := 1

     repeat

          rp := rp+1
          cp := 1
          while rp > cp do
               if numbers[rp] > numbers[cp] then
                    temp := numbers[rp]
                    for count := rp to cp + 1 step - 1
                         numbers[count] := numbers[count - 1]
                    endfor
                    numbers[cp] := temp
               endif
          cp := cp+1
          endwhile
     until rp = max

endproc
```

(a) Using the column headings shown below, trace the algorithm for the procedure *sort* when the array *numbers* contains the values 13, 25, 24 and max = 3.



| Comment | Count | rp | max | cp | temp | numbers | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | 1 | 2 | 3 |
| Global values on call | | | 3 | | | 13 | 25 | 24 |

**(10)**

(b) Name the sort method used in the algorithm above.

______________________________________________________________________

**(1)**

(c) Why would this method be inefficient if the array *numbers* contained 500 values?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(2)**

**(Total 13 marks)**

## Q20.

The operating system of a computer network includes the following functions and procedures:

OpenScreen( ComputerName, Channel )

where ComputerName is a character string identifying a computer on the network, and Channel is an integer identifying a communication channel.

This function opens a communication channel to the screen of the computer specified,and returns an integer, which is 0 if the function is successful, otherwise it returns one of various error codes.

SendCharacter( Char, Channel, x, y )

where Char is a character, Channel identifies the communication channel, and the integer variables x and y are screen coordinates. This procedure sends a character to the screen of the other computer using the communication channel. It does not return a value.

CloseScreen( Channel ) closes the specified communication channel. It does not return a value.

InputText( Buffer ) accepts a string of characters from the keyboard, terminated by a carriage return (character code 13), and stores it in Buffer. It does not return a value.

A computer on the network is running a program, designed to enable the user to send messages to another computer user. Part of the program uses the following algorithm:

```
Array of characters: Msg[50]
```

```
Integer: Count, Err, Col, Row
Character: Ch

InputText ( Msg )        // uses carriage return, code 13,  as terminator
Count := 0
Err := OpenScreen( “Admin_Computer”, 10 )
if ( Err = 0 ) then
       Col := 1
       Row := 12
       Ch := “A”
       while ( Ch does not contain the code 13 ) do
               Ch := Msg[ Count ]
               SendCharacter( Ch, 10, Col, Row )
               Count := Count + 1
               Col := Col + 1
       endwhile
       CloseScreen( 10 )
else
       case ( Err ) of
               when 1: print( “Specified computer is offline
               or does not exist” )
               when 2: print( “Cannot output - network interface problem”
)
               when 3: print( “The network is down” )
       endcase
endif
```

(a) What is meant by the term parameters? Illustrate your answer by using examples of the use of parameters from the algorithm above.

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(3)**

(b) What is the benefit to the programmer of using parameters?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(2)**

(c) How would the array Msg be stored?

______________________________________________________________________

______________________________________________________________________

**(1)**

(d) Describe in detail the operation of the **while** ... **endwhile** section of the algorithm.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

**(4)**

(e) What is the effect of the **case** … **of** … **endcase** section of the algorithm?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

**(2)**

(f) The algorithm does not impose any limit on the length of the string the user inputs. What might happen if a string 60 characters long were entered?

________________________________________________________________________

________________________________________________________________________

**(1)**

**(Total 13 marks)**

## Q21.

(a) Draw a diagram to show the structure of a queue.

**(3)**

(b) Write an algorithm to show how a data item is added to a queue. Take into account the possibility that the queue is full.

**(5)**
**(Total 8 marks)**

## Q22.

A routine for manipulating text strings uses the following pre-defined functions:

Len(q) returns the number of characters in the text string q;

Right$(q,p) returns a string consisting of the last (rightmost) p characters of the string q;

Left$(q,p) returns a string consisting of the first (leftmost) p characters of the string q.

The algorithm for this routine is shown below.

string: message, newstring

```
// main program

     input message
     newstring : = “”
     output message
     docharacter(message,newstring)
     output newstring
// end of program
```

**procedure** docharacter(a,outstring) // *both parameters passed by reference* string: piece integer: x

```
     x : = Len(a)
     piece : = Right$(a,1)
     outstring : = outstring + piece
     x : = x - 1
     if x > 0 then
          a : = Left$(a,x)
          docharacter(a,outstring)
     endif
endproc
```

Trace the algorithm and show what is output if the word CAT is input.

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(Total 9 marks)**

## Q23.

A security system uses the control port shown below.

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| Alarm | Security light | not used | Window contact | Door contact | Internal movement sensor | External movement sensor | System activated |

The purpose of each bit is shown in the diagram.
Bits 0 to 4 are inputs, bit 5 is not used and should always be zero, bits 6 and 7 are output bits.
The bit pattern shown occurs when the system is first activated.

Detection of movement results in the corresponding bit being set to 1.
Breaking of a contact results in the corresponding bit being set to 0.
Bits 6 and 7 will turn on the security light and alarm respectively when set to 1.

The system, if activated, must turn on the security light if external movement is detected. The alarm must be turned on if either or both contacts are broken or if internal movement is detected.

(a) Give the masks and the logical operations needed for **each** of the following. In each case all other bits must remain unchanged.

(i) Testing the state of the external movement sensor.

______________________________________________________________

______________________________________________________________

______________________________________________________________

______________________________________________________________

(ii) Turning on the alarm.

______________________________________________________________

______________________________________________________________

**(4)**

(b) Write an algorithm for the procedure required to check the sensors and contacts and to activate the alarm or security light when necessary.

**(7)**
**(Total 11 marks)**

## Q24.

The following section of pseudo-code processes a one-dimensional integer array called *List.* The numbers in *List* are stored in ascending order, and x, *Low, High, Middle* are all integer variables. (The function Int returns the whole number part of its parameter.)

```
Proc Process(Low, High, x)
         Found ← False
        Repeat
                 Middle ← Int((Low + High)/2)
                 If List(Middle) = x
                 Then Found ← True
                 Else If List(Middle) > x
                      Then High ← Middle -1
                      Else Low ← Middle +1 {List(Middle) <x}
        Until Found = True
```

(a) Complete the following dry-run table for Process (1, 10, 19), given that the integers in the list are:

2,4, 6, 7, 11, 13, 19, 21, 27, 29

| Low | High | Middle | Found |
|---|---|---|---|
| 1 | 10 | | |

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| | | | |

**(7)**

(b) What type of routine does this pseudo-code define?

______________________________________________________________________

**(1)**
**(Total 8 marks)**

## Q25.

Explain how the elements in a non-empty queue may be reversed with the aid of a stack.

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

**(Total 4 marks)**