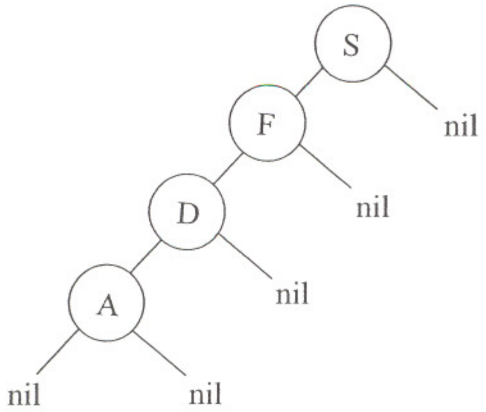
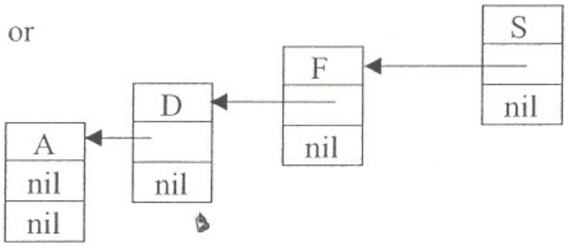
1. (a) *Award **[1 mark]** for data in the tree in the correct order, and **[1 mark]** for indicating most of the nil links (this cannot be given if the first mark is not gained).*



or



(b) (i) One solution is:

```
procedure SEARCH
   declare SEARCHITEM character
   declare CURRENT pointer->NODE

   output 'Enter item to find'
   input SEARCHITEM

   CURRENT <- ROOT

   while (CURRENT # nil) and (CURRENT->DATA # SEARCHITEM) do
      if SEARCHITEM < CURRENT->DATA then
         CURRENT <- (CURRENT->LEFT)
      else
         CURRENT <- (CURRENT->RIGHT)
      endif
   endwhile

   if CURRENT = nil then
      output 'data not in tree'
   else
      output 'data node is',CURRENT->NUMBER
   endif
endprocedure SEARCH
```

Award marks as follows:

- ***[1 mark]*** for most of the variables declared
- ***[1 mark]*** for user input of data to find
- ***[3 marks]*** for correct loop (***[1 mark]*** for any loop, ***[1 mark]*** for terminating when no data is found (*i.e.* nil) and ***[1 mark]*** for terminating when found)
- ***[3 marks]*** for correct branch (***[1 mark]*** for correct **if** statement, ***[1 mark]*** for correct left branch, ***[1 mark]*** for similar attempt at right branch)
- ***[1 mark]*** for correct test and output of error message
- ***[1 mark]*** for correct output of node number (no comment statement is required)

(ii) *Award marks as follows:*

O(logn) ***[1 mark]***
O(n) ***[1 mark]***
(Note: unless clearly labelled, do not give marks if values are in opposite order).

(c) (i) *Award marks as follows:*

`output`	STACK	CURRENT->NUMBER	CURRENT->DATA	***Marks***
	nil	1	Q	Given
Q	3T,nil			Given
		2	C	***[1 mark]***
C	4J, 3T,nil			***[1 mark]***
	3T,nil	4	J	***[2 marks]***
J				
	5M,3T,nil			***[1 mark]***
		6	E	***[1 mark]***
E				
	3T,nil	5	M	***[2 marks]***
M				
	nil	3	T	***[1 mark]***
T				
		nil	nil	

If the candidate's table gives composite results without interim steps (e.g. a push and pop in one step) do not penalise. For example the following table would get full marks:

`output`	STACK	CURRENT->NUMBER	CURRENT->DATA
	nil	1	Q
Q	3T,nil	2	C
C	4J, 3T,nil	4	J
J	5M,3T,nil	6	E
E	5M,3T,nil	5	M
M	3T,nil	3	T
T	nil	–	–

(ii) *Award **[1 mark]** for the following answer:*

Preorder

(iii) Solutions include:

```
procedure RECTRAVERSE (val CURRENT pointer -> NODE)
   output CURRENT -> DATA
   if CURRENT -> LEFT # nil then
      RECTRAVERSE (CURRENT -> LEFT)
   endif
   if CURRENT -> RIGHT # nil then
      RECTRAVERSE (CURRENT -> RIGHT)
   endif
endprocedure RECTRAVERSE
```

```
procedure RECTRAVERSE (val CURRENT pointer -> NODE)
   if CURRENT # nil then
      output CURRENT -> DATA
      RECTRAVERSE (CURRENT -> LEFT)
      RECTRAVERSE (CURRENT -> RIGHT)
   endif
endprocedure RECTRAVERSE
```

Award marks as follows:

- ***[1 mark]*** for a parameter deduced as **pointer** -> NODE
- ***[1 mark]*** for an correct initial **output** statement (accept **output** CURRENT)
- ***[1 mark]*** for testing correct nil condition (i.e. CURRENT->LEFT # nil (accept **if** LEFT#nil) or CURRENT if appropriate)
- ***[2 marks]*** for recursive call to left link (***[1 mark]*** for <u>any</u> attempt at recursion and ***[1 mark]*** for correct parameter in recursion call)
- ***[1 mark]*** for a similar attempt at the right call.

(a) *Award marks as follows, up to **[2 marks]** max:*

- new supercomputers would have predicted bad storm ***[1 mark]***, whereas less powerful ones didn't ***[1 mark]***.

- newer computers will be able to process equations faster ***[1 mark]***, giving forecasts earlier ***[1 mark]***.

- new computers will be able to process more complex equations (*i.e.* more than 7 variables) ***[1 mark]***, giving more accurate predictions ***[1 mark]***.

(b) *Award **[1 mark]** for defining archive data, and **[2 marks]** for a clear description of its use (**[1 mark]** for a reasonable attempt), up to **[3 marks]** max.*

- Data kept after initial use / for long-term store / not required for on-line access ***[1 mark]***.

- Used for research / tracking history ***[1 mark]*** to test for patterns ***[1 mark]***.

(c) *Award **[1 mark]** for each valid point shown, up to **[5 marks]** max:*

- people base business / leisure decisions on weather
- much of the economy is dependent on weather
- so accuracy is important

Limitations:

- only as good as the formulae
- developed on past weather
- which is not consistent
- so formulae are always changing
- because the number of variables need to be increased / can't predict 'nature'

(d) *Award marks as follows:*

(i) - Data collection site ⇒ National Weather Service ***[1 mark]***

(ii) - forecasting based on it ***[1 mark]***, so important for accuracy ***[1 mark]***

(e) *Award marks as below, up to **[4 marks]** max:*

Computers only work to set rules ***[1 mark]*** as determined by humans / based on past patterns ***[1 mark]***.

Since weather is not human-controlled / 'unpredictable' ***[1 mark]*** it is not possible to totally predict / simulate it ***[1 mark]***. Past weather patterns are not necessarily a good predictor of future weather ***[1 mark]***.

(f) *Award **[2 marks]** for a genuine fail-critical situation (**[1 mark]** for a fail-safe situation, or a fail-critical situation not clearly described), and **[2 marks]** for a clear description of how predictions can be improved (**[1 mark]** for a reasonable attempt), up to **[4 marks]** max.*

E.g. of fail-critical situations: Prediction of avalanches in skiing areas which could kill skiers if not forecast, storms at sea which could kill fisherpeople, sailors *etc.*

Better formulae / use of variables / more powerful computers ***[1 mark]*** improve forecasting ***[1 mark]***.

Research could be concentrated upon such areas ***[1 mark]*** for example using archive data ***[1 mark]***.

(g) The METAR format ***[1 mark]***. This is important so that everyone using the data can understand it ***[1 mark]*** and there are no misunderstandings / 'mistakes' ***[1 mark]***.

(h) *Award **[1 mark]** for a correct identification, and **[2 marks]** for a clear reason, for two reasons, giving a maximum of **[6 marks]**.*

New media will take less space ***[1 mark]***. As archive data continues to increase ***[1 mark]***, it will keep space used to a minimum if more compact media is used ***[1 mark]***.

Current media will become out-of-data / obsolete ***[1 mark]***. New media is always being developed ***[1 mark]*** and if data is not changed with the new media, it may not be able to be read ***[1 mark]***.

3. (a) *Award marks as follows:*

(i) An array of set length is declared ***[1 mark]***.
A variable is used for the end location ***[1 mark]***.

(ii) An array of set length is used ***[1 mark]***.
A variable is required for start location and another for end location ***[1 mark]***.

(iii) Pointers are used ***[1 mark]***.
A record / node is created (as required) ***[1 mark]***.
Pointers to start / end are used ***[1 mark]***.

(b) *Award marks as follows up to **[4 marks]** max:*

In a static array the head will always be in position 1/0 ***[1 mark]***.
Additions are placed at the first free location ***[1 mark]***.
Removing data 'shuffling up' all the data behind it ***[1 mark]***.

In a circular array data is added at the location given by a variable ***[1 mark]***.
As data is removed, the head variable (pointer) simply increases by one ***[1 mark]***.
so much less data movement is required so it is faster ***[1 mark]***.

Allow any of the marks to be gained by giving a clear diagram.

(c) *Award **[2 marks]** for **one** well-explained advantage of **each** method (**[1 mark]** for a partially correct/incomplete answer). (**[4 marks]** maximum.)*

Advantages of circular array over dynamic linked list:

dealing with array locations of an array is faster than pointer ***[1 mark]*** since new nodes do not need creating / direct access of memory locations is faster than pointer manipulation ***[1 mark]***.

Array implementation cannot run out of memory ***[1 mark]*** since space is allocated at start of program ***[1 mark]***. (Of course, queue may get full, but this is covered below).

Advantage of DLL over CA:

A node can be allocated as required (during the program) ***[1 mark]***, thus size of queue is only restricted by size of available RAM ***[1 mark]***.

4. (a) *Award marks as follows up to **[2 marks]** max:*

tracks above each other ***[1 mark]*** where data is stored ***[1 mark]*** to minimise head movement ***[1 mark]***.

(b) *Award marks as follows up to **[2 marks]** max:*

to compensate for speed difference ***[1 mark]*** between peripheral and processor ***[1 mark]***.

(c) *Award marks as follows, up to **[3 marks]** max:*

if there wasn't a second buffer, the head could not start reading as sector start passes under head because the buffer wouldn't be empty ***[1 mark]***, part way through rotation the buffer would be empty ***[1 mark]*** so would have to wait (wasted time) until sector passed back under head ***[1 mark]***. Switching to the second buffer stops this delay ***[1 mark]*** and so data transfer is faster ***[1 mark]***.
(Do not award marks for answers like "one is used for reading, and one is used for writing" etc.)

(d) *Award marks as follows, up to **[6 marks]** max:*

data transfer is slow compared to processor speed ***[1 mark]***
if processor was 'in charge' of transfer, it could not do other tasks ***[1 mark]***
this would slow down overall processing (*e.g.* user programs) ***[1 mark]***
the DMA transfers data independent of processor ***[1 mark]***
so processor can continue with other tasks ***[1 mark]***
DMA receives data from the disk drive ***[1 mark]***
and stores it directly into RAM ***[1 mark]***

(e) *Award marks as follows up to **[2 marks]** max:*

fully-indexed file has every record key in its index ***[1 mark]***
partially-indexed file has data in sorted order ***[1 mark]***

5. (a) *Award marks as follows, up to **[3 marks]** max:*

Design solution ***[1 mark]*** using top-down approach ***[1 mark]*** of the overall problem ***[1 mark]*** so that all the teams' parts are mapped / linked together ***[1 mark]***.

(b) *Award marks as follows, up to **[3 marks]** max:*

(i) To test final code ***[1 mark]*** because results are independent of programming ***[1 mark]*** they are problem-based, not code-based ***[1 mark]***. Not allow knowledge of program dictate test data ***[1 mark]***.

(ii) *Award **[1 mark]** for any feasible person, e.g.:*

Systems analyst / weather forecast manager
(Do not accept programmer)

(c) *Award marks as follows, up to **[2 marks]** max for (i) and (ii), **[4 marks]** max for (iii):*

Disruption must be minimised during installation ***[1 mark]*** so that forecasting can continue ***[1 mark]*** because it is an important activity ***[1 mark]***.

The training must be done so that staff are confident in the new system ***[1 mark]*** before it is to be used on its own ***[1 mark]*** so that errors are not made ***[1 mark]***

Parallel running could be used to compare results ***[1 mark]***, but since large systems / fast time is required, may not be feasible ***[1 mark]*** so probably a direct changeover would be used ***[1 mark]*** but if errors are found, there is no old system to revert to ***[1 mark]***. A staggered changeover could be used ***[1 mark]*** where some tasks are carried out on the new system and others on the old ***[1 mark]*** until it is certain that it works ***[1 mark]*** then a new task is transferred until the new system is in full use ***[1 mark]***.