1(a). A binary search tree, colour, stores data about colours that are entered into a computer.

A binary search tree is one example of a type of tree.

(i) State the main features of a tree.

---

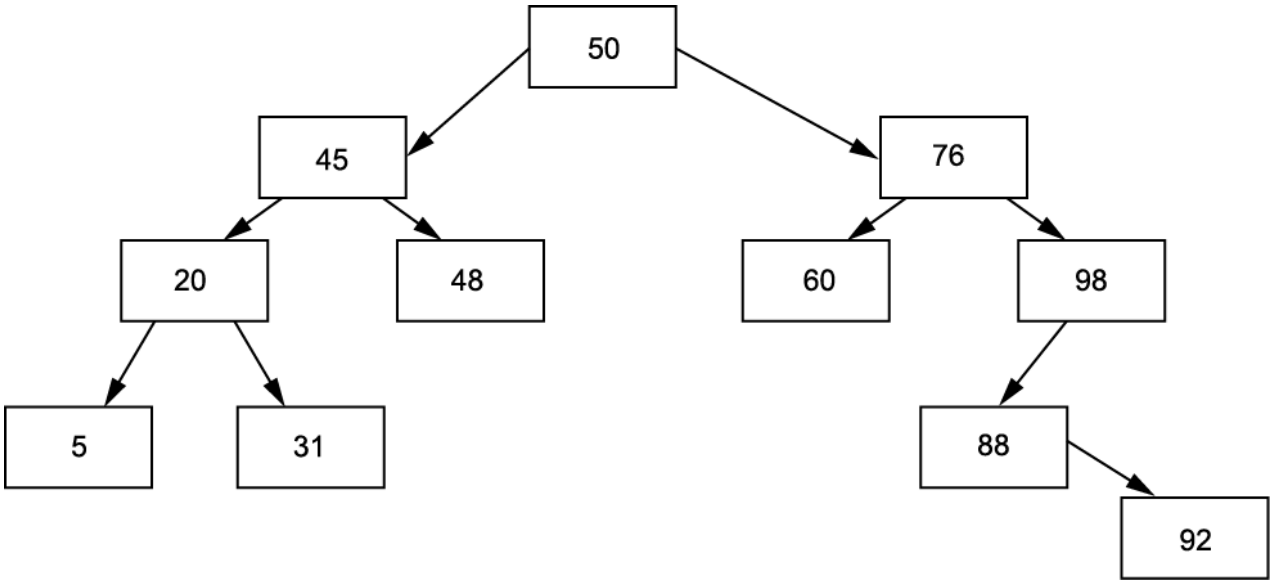
[3]

(ii) State the features that make a tree a binary search tree.

---

[1]

(b). A binary search tree, numbers, stores numbers that are entered into a computer. The contents of the tree are shown below:



(i) Explain, using the binary search tree numbers as an example, how a depth-first (post-order) traversal is performed.

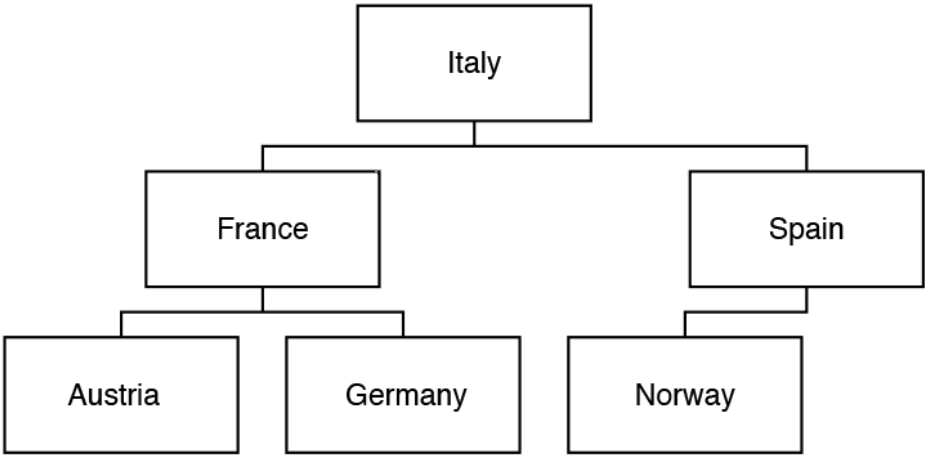
[5]

(ii) Explain, using the binary search tree numbers as an example, how a breadth-first traversal is performed.

[5]

2. A program stores entered data in a binary search tree.

The current contents of the tree are shown:



A pseudocode algorithm is written to search the tree to determine if the data item "Sweden' is in the tree.

The function `currentNode.left()`returns the node positioned to the left of `currentNode.`

The function `currentNode.right()`returns the node positioned to the right of `currentNode.`

```
function searchForData(currentNode:byVal, searchValue:byVal)
     thisNode = getData (.................................)
     if thisNode == ..................................... then
          return .............................................
     elseif thisNode < searchValue then
          if currentNode.left() != null then
               return (searchForData(currentNode.left(), searchValue))
          else
               return .................................
          endif
     else
          if ..................................... != null then
               return (searchForData(currentNode.right(), searchValue))
          else
               return false
          endif
     endif
endfunction
```

(i) Complete the algorithm.

[5]

(ii) The algorithm needs to be used in different scenarios, with a range of different trees.

Identify **two** preconditions needed of a tree for this algorithm to work.

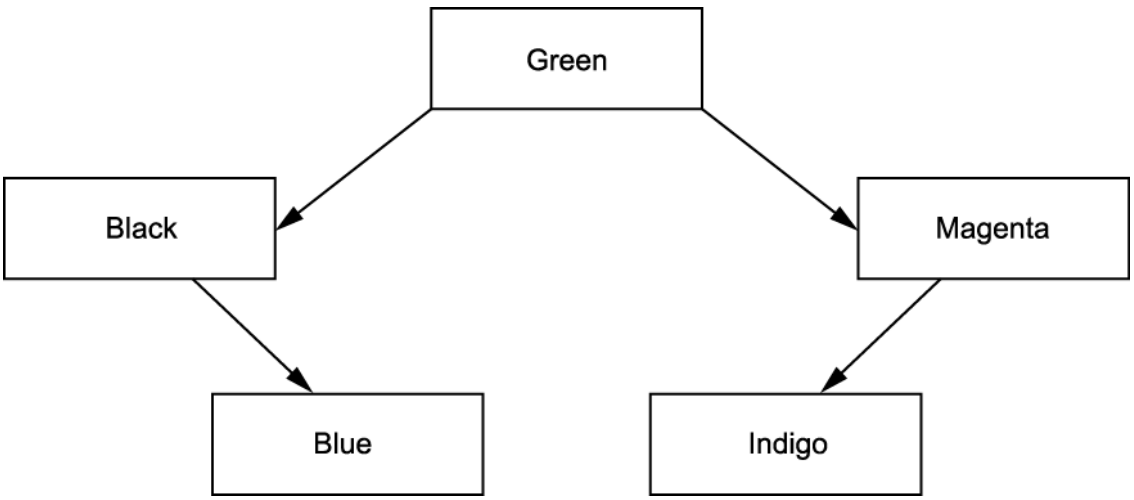
1 ----------

2 ----------

[2]

3. A binary search tree, colour, stores data about colours that are entered into a computer.

The current contents of colour are shown.



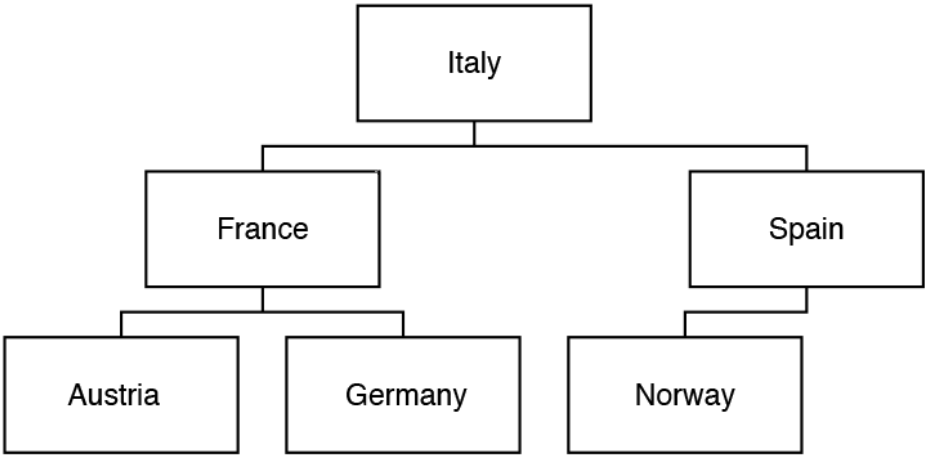
Add the following colours to the tree above in the order written:

Brown White Orange Purple

[4]

4(a). A program stores entered data in a binary search tree.

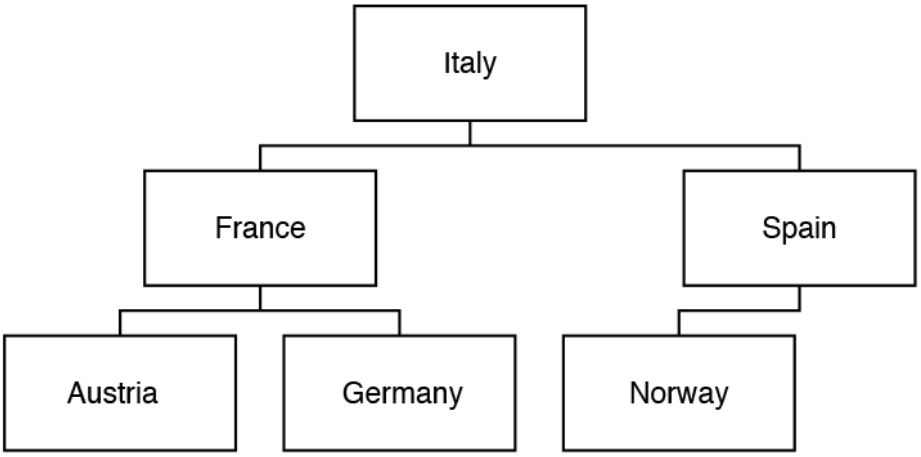
The current contents of the tree are shown:



Complete the diagram to show the contents of the tree after the following data is added:

England, Scotland, Wales, Australia **[3]**

(b). Show the order of the nodes visited in a breadth first traversal on the following tree.



[3]

5(a). A data structure is shown below in Fig. 4.1.

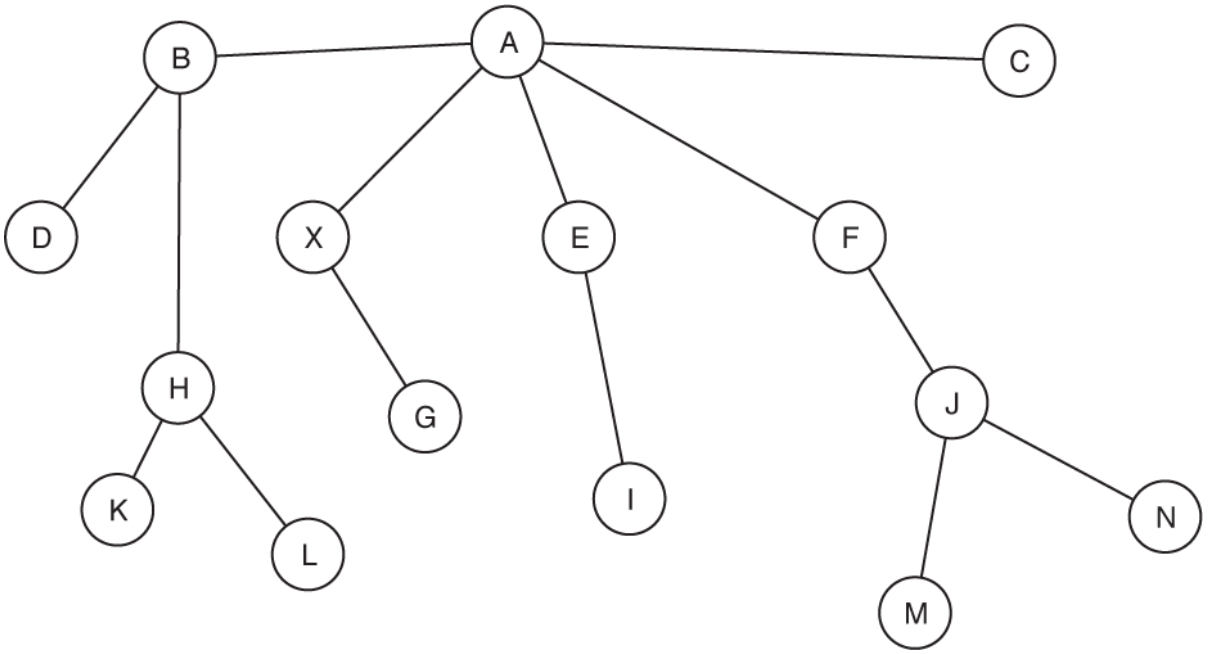


Fig. 4.1

Identify the data structure shown in Fig. 4.1.

---

[1]

(b). The programmer is considering using a depth-first (post-order) traversal, or a breadth-first traversal to find the path between node A and node X.

(i) Explain the difference between a depth-first (post-order) and breadth-first traversal.

---

[4]

(ii) Show how a depth-first (post-order) traversal would find the path between node A and node X for the structure shown in Fig. 4.1.

---

[6]

(iii) Explain how you used backtracking in your answer to part **(b)(ii)**.

[3]

6(a). A programmer is developing an ordering system for a fast food restaurant. When a member of staff inputs an order, it is added to a linked list for completion by the chefs.

Each element in a linked list has:

- a pointer, `nodeNo`, which gives the number of that node
- the order number, `orderNo`
- a pointer, `next`, that points to the next node in the list

Fig. 2.1 shows the current contents of the linked list, `orders`.

nodeNo	orderNo	next
0	154	1
1	157	2
2	155	3
3	156	∅

Fig. 2.1

∅ represents a null pointer.

(i) Order 158 has been made, and needs adding to the end of the linked list.

Add the order, 158, to the linked list as shown in Fig. 2.1. Show the contents of the linked list in the following table.

nodeNo	orderNo	next

[2]

(ii) Order 159 has been made. This order has a high priority and needs to be the second order in the linked list. Add the order, 159, to the original linked list as shown in Fig. 2.1. Show the contents of the linked list in the following table.

nodeNo	orderNo	next

[3]

(b). The linked list is implemented using a 2D array, `theOrders`:

- Row 0 stores `orderNo`
- Row 1 stores `next`

The data now stored in `theOrders` is shown in Fig. 2.2.

184	186	185	187
1	2	3	

Fig. 2.2

`theOrders [1,0]` would return `1`

The following algorithm is written:

```
procedure x()
    finished = false
    count = 0
    while NOT(finished)
        if theOrders[1,count] == null then
            finished = true
        else
            output = theOrders[0,count]
            print(output)
            count = theOrders[1,count]
        endif
    endwhile
    output = theOrders[0,count]
    print(output)

endprocedure
```

(i) Outline why nodeNo does not need to be stored in the array.

---

[1]

(ii) Complete the trace table for procedure x, for the data shown in Fig. 2.2.

finished	count	output

[3]

(iii) Describe the purpose of procedure x.

[2]

(iv) A new order, 190, is to be added to `theOrders`. It needs to be the third element in the list.
The current contents of the array are repeated here for reference:

184	186	185	187		
1	2	3			

Describe how the new order, 190, can be added to the array, so the linked list is read in the correct order, without rearranging the array elements.

[4]

(c). Explain why a linked list is being used for the ordering system.

---

[2]

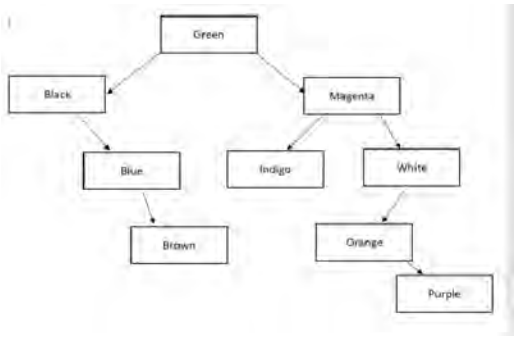
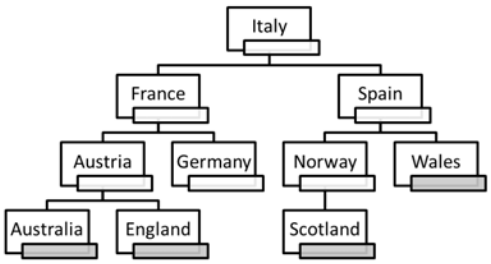
END OF QUESTION PAPER

Question			Answer/Indicative content	Marks	Guidance
1	a	i	1 mark per bullet to max 3 • A data structure • Consists of nodes • That have sub nodes (children) • First node is called the root • Lines that join nodes are called branches	3	
		ii	1 mark per bullet to Max 1 • In a binary search tree, each node only has max. 2 sub nodes • If a child node is less than a parent node, it goes to the left of the parent. • If a child node is greater than a parent node, it goes to the right of the parent.	1	
	b	i	2 marks for correct order • 5, 31, 20, 48, 45, 60 • 92, 88, 98, 76, 50 1 mark per bullet to max 3 for explanation • Visit all nodes to the left of the root node • Visit right • Visit root node • Repeat three points for each node visited	5	
		ii	2 marks for correct order • 50, 45, 76 • 20, 48, 60, 98, 5, 31, 88, 92 1 mark per bullet to max 3 for explanation • Visit root node • Visit all direct subnodes (children) • Visit all subnodes of first subnode • Repeat three points for each subnode visited	5	

Question			Answer/Indicative content	Marks	Guidance
			Total	14	

2		i	1 mark per bullet to max 5
			
			`function searchForData(currentNode:byVal, searchValue:byVal)`
			`  thisNode = getData(currentNode)`
			`  if thisNode == searchValue then`
			`  return true`
			`  elseif thisNode < searchValue then`
			`  if currentNode.left () != null then`
			`  return (searchForData(currentNode.left (), searchValue))`
			`  else`
			`  return false`
			`  endif`
			`  else`
			`  if currentNode.right() != null then`
			`  return (searchForData(currentNode.right (), searchValue))`
			`  else`
			`  return false`
			`  endif`
			`  endif`
			`endfunction`
				5
				AO2.2
				(2)
				AO3.2
				(3)
					The line **`elseif thisNode < searchValue then`** should have read **`elseif thisNode > searchValue then`**
					
					If candidates attempt to correct the code and their answers are consistent with, and work with their amendment, such answers should be credited.
					
					Examiner's Comments
					
					Some candidates found the first two marks more difficult to access than the last three, because they did not fully understand that the parameters to the function were to be used.

		ii	• It's a binary tree • It's ordered / sorted	2 AO2.2 (2)	Examiner's Comments Many candidates recognised that a binary search tree was required, but some lost marks when they specified that 2 children were always required for each parent node, when it is a maximum of two children that are allowed, so that there can be 0, 1 or 2 children.
			Total	7	

Question			Answer/Indicative content	Marks	Guidance
3			1 mark for each node as a correct sub-node 	4	Allow follow through e.g. if white is incorrect, but orange follows through in a logical position.
			Total	**4**	
4	a		 1 mark for each of: - Scotland in correct place - Wales in correct place - Australia and England both in correct place	3 AO2.2 (3)	Examiner's Comments The majority of candidates had little difficulty with this question, but a surprising number of errors were made because candidates could not sort alphabetically past the first letter of the data values given to be inserted into the tree.
	b		1 mark per bullet to max • Italy • France, Spain • Austria, Germany, Norway	3 AO1.1 (1) AO2.1 (1) AO2.2 (1)	Examiner's Comments Most candidates found this to be straight-forward. Where there was confusion, candidates often performed a depth-first traversal, instead of a breadth-first traversal.
			Total	**6**	

Question			Answer/Indicative content	Marks	Guidance
5	a		Tree // Graph (undirected)	1 AO1.2 (1)	Do not accept binary tree
	b	i	1 mark per bullet to max 4 • Depth-first goes to left child node when it can… • If there is no left child it goes to the right child • when there are no child nodes the algorithm 'visits' it' and backtracks to the parent node. • Breadth-first visits all nodes connected directly to start node… • Then visits all nodes directly connected to each of those nodes (and then all nodes directly connected to those nodes and so on…) • Depth-first uses a stack • Breadth-first uses a queue	4 AO1.2 (4)	
		ii	1 mark per node in correct order D→K→L→H→B→G (X)	6 AO2.1 (6)	

Question			Answer/Indicative content	Marks	Guidance
		iii	Max 3 e.g. • When a node does not have any node to visit e.g. D • The algorithm goes back to the previous visited node e.g. B • To check for further nodes to visit e.g. H • This repeats until a new node can be visited, or all nodes have been visited	3 AO1.2 (2) AO2.1 (1)	Examiner's Comment: A number of candidates incorrectly identified the data structure as a binary tree which indicated that this was the only type of tree that they were familiar with. Descriptions of depth and breadth first traversals were often very vague, and precision in terms of the algorithmic steps involved would have produced stronger answers. A pleasing number of candidates produced the correct traversal of the tree, but of those, a number did not appreciate that the node was only output when it was popped from the stack, and hence missed location G before X was actually output.
			Total	14	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>6</td><td>a</td><td>i</td><td>1 mark per bullet

• <code>nodeNo</code> and <code>next</code> columns are both correct

• <code>orderNo</code> column is correct

<code>nodeNo</code> <code>orderNo</code> <code>next</code>
<table><tr><td>Ø</td><td>154</td><td>1</td></tr><tr><td>1</td><td>157</td><td>2</td></tr><tr><td>2</td><td>155</td><td>3</td></tr><tr><td>3</td><td>156</td><td>4</td></tr><tr><td>4</td><td>158</td><td>Ø</td></tr></table></td><td>2
AO1.2
(2)</td><td></td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark per correct column

<code>nodeNo</code> <code>orderNo</code> <code>next</code>
<table><tr><td>Ø</td><td>154</td><td>4</td></tr><tr><td>1</td><td>157</td><td>2</td></tr><tr><td>2</td><td>155</td><td>3</td></tr><tr><td>3</td><td>156</td><td>Ø</td></tr><tr><td>4</td><td>159</td><td>1</td></tr></table></td><td>3
AO1.2
(3)</td><td>Examiner's Comment:
Most candidates scored well in part (i), but fewer understood how the pointers in a linked list could be updated in part (ii) to allow the insertion of the new item in the next free space.</td></tr>
<tr><td></td><td>b</td><td>i</td><td>• The index / subscript of the array acts as the nodeNo</td><td>1
AO1.2
(1)</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark for each correctly completed column

<table><tr><th>Finished</th><th>Count</th><th>output</th></tr><tr><td>False</td><td>0</td><td>184</td></tr><tr><td>(False)</td><td>1</td><td>186</td></tr><tr><td>(False)</td><td>2</td><td>185</td></tr><tr><td>True</td><td>3</td><td>187</td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table></td><td>3
AO1.2
(1)
AO2.2
(2)</td><td></td></tr>
<tr><td></td><td></td><td>iii</td><td>1 mark per bullet to max 2

• Output the order numbers …

• … in the order they are in the linked list</td><td>2
AO2.2
(2)</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td></td><td></td><td>iv</td><td>1 mark per bullet to max

• Order 190 is added to the end

• Pointers are updated

• 186 will point to 4

• 190 will point to 2

OR
<table><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>Data</td><td>184</td><td>186</td><td>185</td><td>187</td><td>190</td></tr><tr><td>Pointer</td><td>1</td><td>4</td><td>3</td><td></td><td>2</td></tr></table></td><td>4
AO1.2
(2)
AO2.1
(2)</td><td>If a diagram is given then the mark for updating the pointers is implicit

Examiner’s Comment:
Few candidates could give a clear answer in part (i) using the correct technical vocabulary that the array index / subscript could be used as the node number. Many candidates could work through the logic required in the trace table in part (ii), but fewer could actually explain what it was doing in (iii) within the context of the scenario. Part (iv) was often best answered by those candidates who used the diagram to give the solution. Candidates should be encouraged to use diagrams where they can be used to good effect rather than lengthy or vague prose descriptions.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
	c		1 mark per bullet, to max 2, e.g. • Orders can be processed in the order they are in the queue • Orders can be inserted at any place in the list e.g. high priority item inserted earlier in the list • Orders can be deleted from any position in the list once they are complete • List is dynamic… • … to allow orders to be added / deleted	2 AO1.2 (1) AO2.1 (1)	Examiner's Comment: Many candidates struggled to apply the context given to computer science concepts and hence answer with the relevant properties of a linked list that would be relevant in context.
			Total	17	