

1(a) The following strings are stored in an array.

"rainbow"	"moon"	"sun"	"stars"	"clouds"	"tornado"
-----------	--------	-------	---------	----------	-----------

Explain how a linear search would search the array for the index that stores "clouds".

[3]

(b) State why a binary search cannot be used in this example.

[1]

- Blank lined paper for writing.

(b) A programmer needs to use a merge sort in one part of the problem to sort items in ascending order.

i. Describe how a merge sort works.

[5]

ii. Give **one** benefit and **one** drawback of the programmer using a merge sort instead of a bubble sort.

Benefit -----

Drawback -----

[2]

3(a) A tree is one example of a data structure.

i. Give **two** characteristics of a tree data structure.

1 -----

2 _____

[2]

ii. The following data is entered into a binary search tree.

22 13 5 36 55 14 8

Draw the binary search tree when the given data is entered in the order given.

[4]

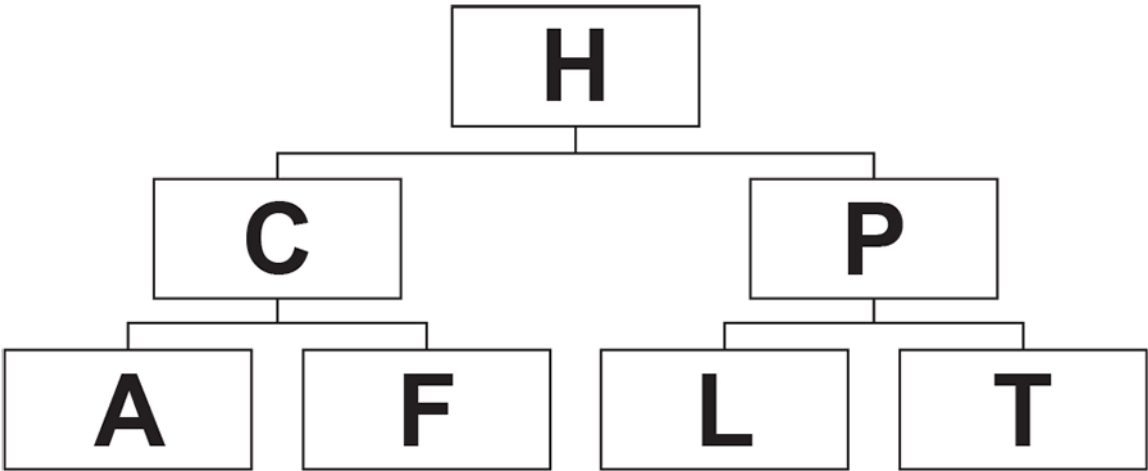
iii. Describe how a **leaf node** is deleted from a binary search tree.

[2]

iv. Describe how a binary search tree can be searched for a value.

[4]

v. Identify the order that the nodes will be visited in a **depth-first (post-order)** traversal of this binary search tree.



[4]

vi. Explain how backtracking is used in depth-first (post-order) traversals.

[2]

(b) A graph is a type of data structure.

An example graph is shown in Fig. 1.

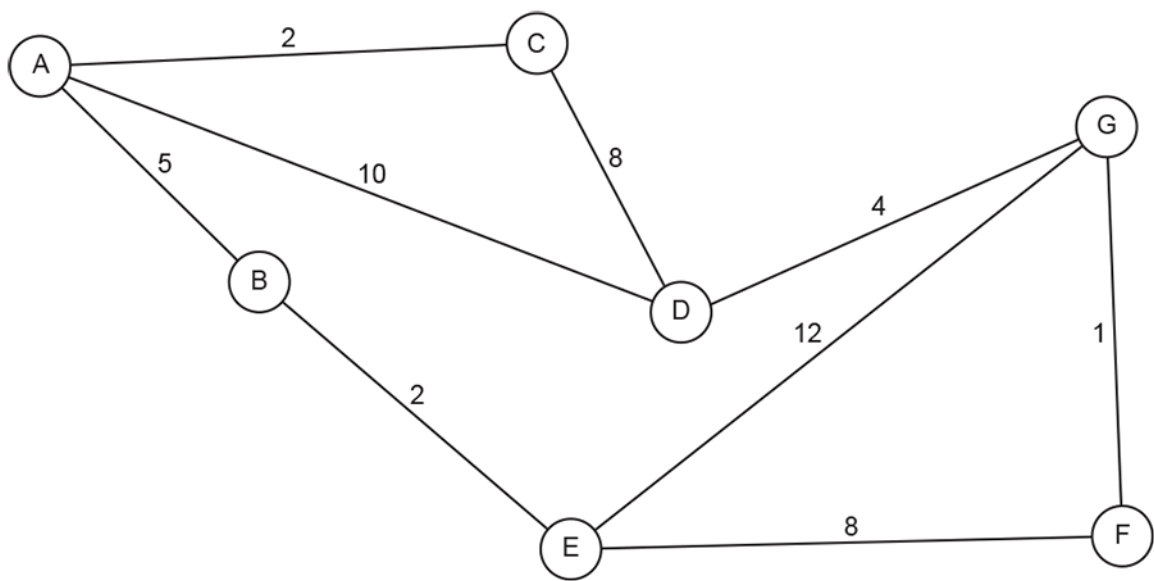


Fig. 1

Show how Dijkstra’s algorithm can be used on the graph shown in Fig. 1 to find the shortest path from start node A to end node G.

You must state the nodes on the final path and the distance of this path. Show your working.

You may use the table below to give your answer.

Node	Distance travelled	Previous node

Final path:

Distance:

[6]

4(a) A program stores data in a linked list.

The current contents of the linked list are shown in Fig. 3, along with the linked list pointers.

headPointer	1
freeListPointer	4

location	data	pointer
0	"blue"	6
1	"red"	0
2	"green"	8
3	"orange"	NULL
4		5
5		7
6	"grey"	2
7		9
8	"purple"	3
9		NULL

Fig. 3

State the purpose of headPointer and freeListPointer in the linked list shown in Fig. 3.

headPointer _____

r _____

freeListPoint _____

er _____

[2]

(b) State the meaning of the pointers with the value NULL in the linked list shown in Fig. 3.

[1]

(c) A procedure outputs the data in the linked list shown in Fig. 3 from the first item in the list, to the last item.

Give the output from the procedure.

[2]

(d) A new item needs to be added to the linked list.

Describe how a new item is added to a linked list.

[4]

- (e) The function `findNode` will search the linked list and return either the position of the node that contains the data item, or -1 if the data item is not found.

The data held in a node at location `x` can be accessed with `linkedList[x].data`. The pointer of the node at location `x` can be accessed with `linkedList[x].pointer`.

For example, using the linked list shown in **Fig. 3**:

`linkedList[2].data` returns green.

`linkedList[2].pointer` returns 8.

Complete the function, using pseudocode or program code.

```
function findNode(toFind, headPointer, linkedList)

    currentNode = .....

    while(currentNode != ..... )

        if linkedList[currentNode]. ..... == toFind then

            return currentNode

        else

            currentNode = linkedList[.....].pointer

        endif

    endwhile

    return .....

endfunction
```

[5]

5(a) A computer program is being written to store data about students.

Fig. 2 shows a binary search tree that stores data about students.

Each student is represented by their ID number. The current contents of the binary search tree are:

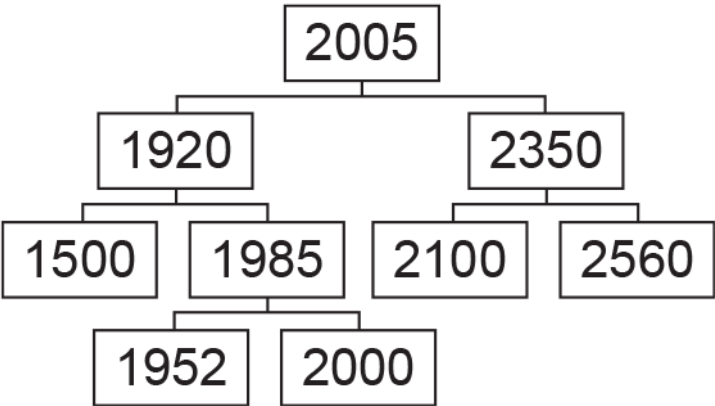


Fig. 2

Identify the root node in the binary tree shown in Fig. 2.

----- [1]

(b) Identify **two** leaf nodes in the binary tree shown in Fig. 2.

1 -----

2 -----

[2]

(c) * A programmer would like to traverse the binary search tree shown in Fig. 2.

Compare the use of a breadth-first traversal and a depth-first (post-order) traversal on the binary search tree.

You should include the following in your answer:

- how each traversal works
- the order of the return values for each traversal.

Blank lined paper for writing.

6(a) A bubble sort will sort an array of 50 integer values called numberArray.

This bubble sort algorithm is written to sort numberArray into ascending numerical order.

Complete this bubble sort algorithm.

```
arrayLength = .....
tempValue = 0
do
    flag = false
    for y = 0 to arrayLength - .....
        if numberArray[y] > numberArray[y + 1] then
            ..... = numberArray[y]
            numberArray[.....] =
numberArray[y + 1]
            numberArray[y + 1] = .....
            flag = true
        endif
    next y
until flag == false
```

[5]

2	12	1	9	3	5	15	7
---	----	---	---	---	---	----	---

Show how a merge sort will sort this section of the array numberArray into ascending numerical order.

[illegible]

(c) * Another sorting algorithm is insertion sort.

The number of values stored in the array `numberArray` has been reduced to 10.

Compare the use of bubble, merge and insertion sorts on the array `numberArray`.

You should include the following in your answer:

- how each algorithm works
- the Big O complexities for each algorithm
- the suitability of each algorithm for sorting the 10 values.

Handwriting practice lines consisting of 20 sets of three horizontal dashed lines.

[12]

- 7 **Fig. 5** shows a graph data structure representing a small section of a parcel delivery network. Each node represents an address where deliveries need to be made. The edges show the possible routes and distances between these deliveries.

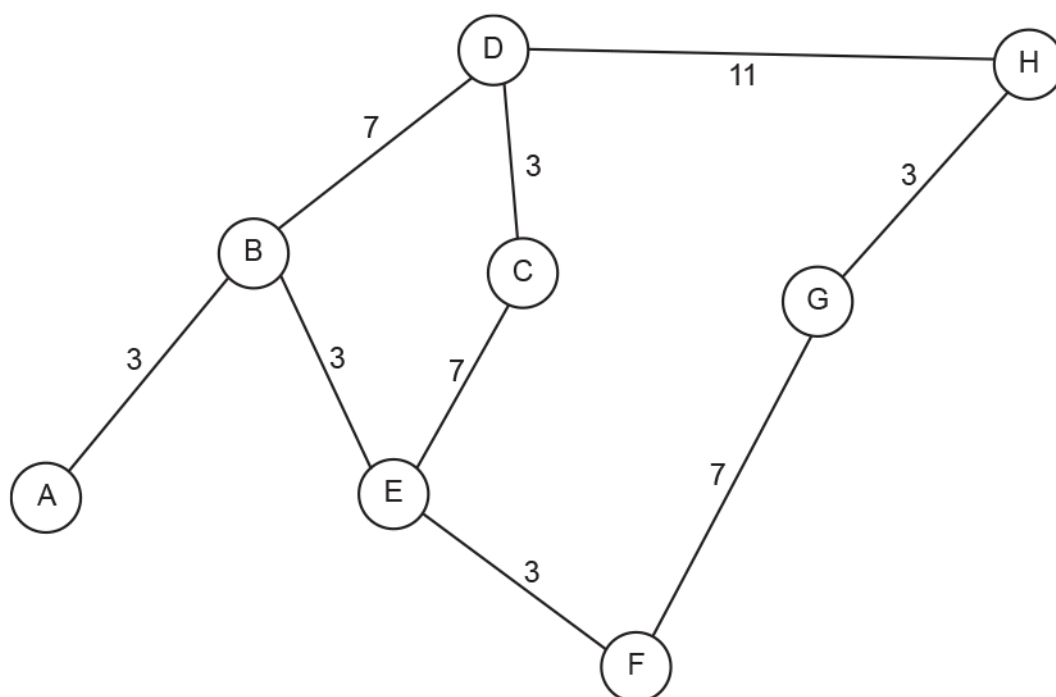


Fig. 5

- i. Show how Dijkstra's algorithm can be used on the graph shown in **Fig. 5** to find the shortest path from the

start node A and the end node H.

You should state the nodes on the final path and the overall distance. Show your working.

You may choose to use the table below to give your answer.

This image shows a blank sheet of white paper with ten horizontal dashed lines, typical of primary-ruled notebook paper. The lines are evenly spaced and extend across the width of the page. There is no handwriting or other markings on the paper.[illegible]

Final path:

Distance:

[6]

- ii. Give a similarity and difference between the performance of Dijkstra's algorithm and the performance of A* algorithm.

Similarity

.....

Difference

.....

[2]

- 8 Some of the characters in a game will move and interact independently. Taylor is going to use graphs to plan the movements that each character can take within the game.

DancerGold is one character. The graph shown in Fig. 1 shows the possible movements that DancerGold can make.

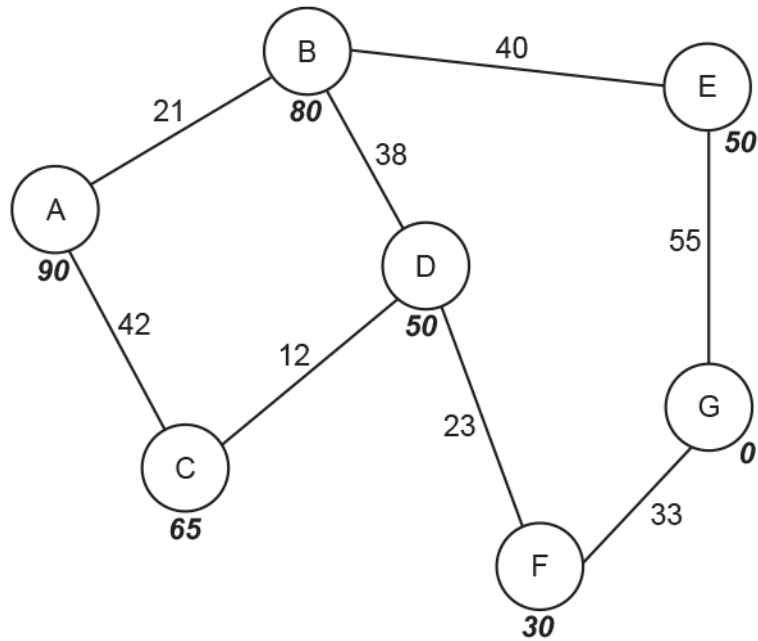


Fig. 1

DancerGold's starting state is represented by node A. DancerGold can take any of the paths to reach the end state represented by node G.

The number on each path represents the number of seconds each movement takes.

The number in bold below each node is the heuristic value from A.

- i. Define the term heuristic in relation to the A* algorithm.

[2]

- ii. Perform an A* algorithm on the graph shown in Fig. 1 to find the shortest path from the starting node to the end node. Show your working, the nodes visited and the distance. You may choose to use the table below to give your answer.

Node	Distance travelled	Heuristic	Distance travelled + Heuristic	Previous node

Final path: -----

Distance: -----

9 Kira is creating a computer game where the user can play against the computer.

In each turn, each character can make one move from a selection of possible moves.

Kira uses a tree data structure shown in Fig. 1 to identify the range of possible moves the computer can make from starting position A. Each connection is a move, with each node representing the result of the move.

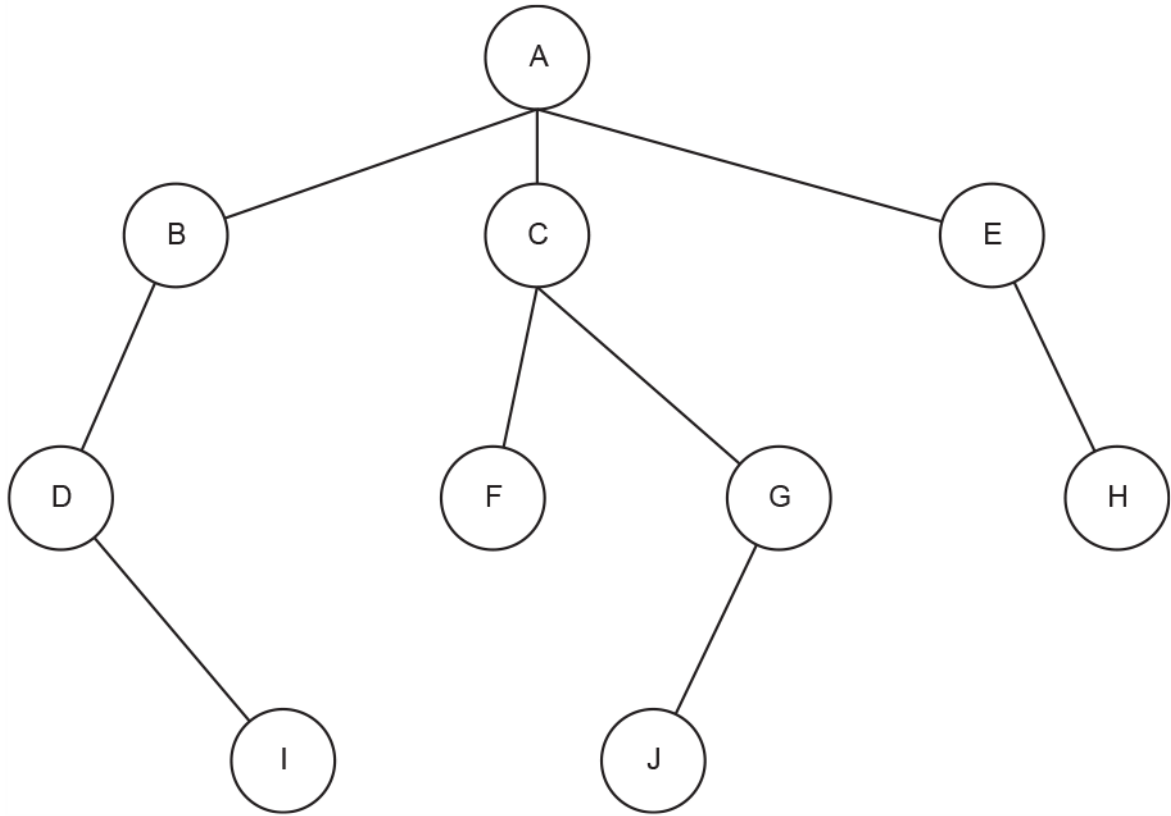


Fig. 1

Kira wants to make some changes to the data that is stored in the tree structure shown in Fig. 1.

- i. The move represented by node 'E' needs to be deleted.

Describe the steps an algorithm will follow to delete node 'E' from the tree.

- ii. The move represented by the node 'K' needs to be added. Node 'K' needs to be joined to node 'G.'

Describe the steps the algorithm will follow to add node 'K' to the right of node 'G'.

10(a) The following pseudocode procedure performs an insertion sort on the array parameter.

```
01 procedure insertionSort(dataArray:byRef)
02   for i = 1 to dataArray.Length - 1
03     temp = dataArray[i]
04     tempPos = i - 1
05     exit = false
06     while tempPos >= 0 and exit == false
07       if dataArray[tempPos] < temp then
08         dataArray[tempPos + 1] = dataArray[tempPos]
09         tempPos = tempPos - 1
10       else
11         exit = true
12       endif
13     endwhile
14     dataArray[tempPos + 1] = temp
15   next i
16 endprocedure
```

* Two sorting algorithms are merge sort and quick sort.

Compare the use of merge sort, quick sort and insertion sort on an array with a small number of elements, and on an array with a very large number of elements.

You should make reference to the time complexities of each algorithm using the Big O notation in your answer.

Blank lined paper for writing.

[6]

12 An insertion sort can be used to sort the array instead of a merge sort.

[2]

Created in ExamBuilder

- 13 The temperatures of an ocean are input into a computer system. They are recorded, and will be accessed, in the order in which they arrive. The data for one week is shown:

5, 5.5, 5, 6, 7, 6.5, 6

The data is to be stored in a data structure. The programmer stores the data in a queue.

Explain why a queue is used instead of a stack.

[2]

END OF QUESTION PAPER

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
1	a		1 mark each • Compare the first element (rainbow) to search item / clouds • If it is equal to the search item return index / found • If it is not equal move to the next element • Repeat until either search item / clouds is equal // or the end of the list has been reached	3	Allow answers by example from the given dataset <u>Examiner's Comments</u> Most candidates scored the majority of the marks available and demonstrated a clear understanding of a linear search. Many candidates answered by example with values from the given list.
	b		1 mark for: the data is not in order/sorted	1	<u>Examiner's Comments</u> Most candidates correctly identified the requirement for data to be sorted/ordered for a binary search to work. 'Organised' was too vague and was not accepted.

Mark Scheme

Question		Answer/Indicative content						Marks	Guidance									
	c		rainbow	moon	sun	stars	clouds	tornado	Marking Guidance	Marking Guidance	5	If candidate has given <u>descending</u> order, max 4. MP1, MP3 and MP4 are lines that show a change of values during a pass. MP2 and MP5 do not have to be explicitly given in full if there is a comment to identify no change occur during the pass. Award no marks if not an insertion sort. Examiner's Comments Nearly half the candidates achieved full marks and clearly demonstrated the steps involved in an insertion sort for the given data. Some candidates confused insertion sort with either bubble, merge or selection sort, and so scored no marks for not answering the question. Exemplar 3 (c) Show how an insertion sort will sort the given data into <u>ascending</u> alphabetical order. ABCDEFGHIJKLMNOPQRSTUVWXYZ <table><tr><td>"rainbow"</td><td>"moon"</td><td>"sun"</td><td>"stars"</td><td>"clouds"</td><td>"tornado"</td></tr></table> rainbow moon, sun, stars, clouds, tornado moon, rainbow sun, stars, clouds, tornado moon, rainbow, sun stars, clouds, tornado moon, rainbow, stars, sun clouds, tornado clouds, moon, rainbow, stars, sun tornado clouds, moon, rainbow, stars, sun, tornado * A sorted list is made with an element in the list. The element in the old list (right) are sorted into the new list (left) in ascending order.	"rainbow"	"moon"	"sun"	"stars"	"clouds"	"tornado"
"rainbow"	"moon"	"sun"	"stars"	"clouds"	"tornado"													
		Total						9										

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
2	a	Mark Band 3 – High level (7–8 marks) The candidate demonstrates a thorough knowledge and understanding of Big O; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. <i>There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.</i> Mark Band 2 – Mid level (4–6 marks) The candidate demonstrates reasonable knowledge and understanding of Big O; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. <i>There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.</i> Mark Band 1 – Low Level (1–3 marks) The candidate demonstrates a basic knowledge of Big O with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. <i>The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be</i>	9	AO1: Knowledge and Understanding Indicative content • Big O measures the number of steps and memory usage change according to the data as the amount of data being processed increases • Linear - grows in proportion to amount of data • Exponential – the rate of increase is at the rate kn as n increases • Constant - it does not change • Logarithmic – means the rate of increase gets smaller as the amount of data increases time / time increases at a rate of $\log kn$ as n increases. AO2: Application • Algorithm 1 – The time taken increases as the data set grows. The space taken also significantly increases. This algorithm is not memory efficient. • Algorithm 2 – The time increases significantly and therefore this algorithm is not time efficient. The space will never change which means the amount of memory will not change as the data set grows. • Algorithm 3 – The time will grow less fast as the data set grows relative to the other algorithms. The space required will also increase, but not insurmountably. This is therefore an efficient algorithm with large data sets compared to algorithm 1 and 2 overall. AO3: Evaluation • Number of elements is unknown. Exponential is least appropriate because this could increase significantly and be unmanageable. • Constant is the most ideal as the time will not increase. • Algorithm 3 is more suitable because it has a logarithmic time complexity, so it increases less quickly than the other algorithms. It will be reasonable with a small amount (2 items) of data, but then when very large amounts (2 billion items) are needed it will not be significantly more.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			<i>clear.</i> 0 mark No attempt to answer the question or response is not worthy of credit.		<u>Examiner's Comments</u> Several candidates confused logarithmic and exponential growth rates, and a significant number of candidates thought that exponential complexity was the polynomial function $O(n^2)$ rather than $O(2^n)$. Many candidates also defined terms such as exponential complexity as 'where it grows exponentially', such circular definitions were not given marks. Level 1 responses identified some characteristics of Big O for constant, logarithmic, linear and exponential complexity with some occasional errors or inconsistencies. Level 2 responses gave reasonable descriptions of both time and space complexity alongside accurate definitions for the complexity terms with an identification of algorithm 3 as being the best overall choice. Level 3 responses had far more evaluative AO3 analysis but were few and far between. In this scenario candidates identified that it depended on the value of n, as if n was very small, exponential time growth was not an issue, so algorithm 2 might be preferential, but as n could grow to 2 billion this would be intractable, so algorithm 3 was chosen as the most practical. The data set in question had 2 billion items at most so $\log_2 2,000,000,000 = 30$ meant space overheads in algorithm 3 were manageable with modern storage capacities. Exemplar 2

Mark Scheme

Question	Answer/Indicative content	Marks	Guidance
			Order is used to evaluate the complexity of an algorithm, in respect to how its performance changes with a change to the size of input. This may be measuring how much longer it takes to complete, or how much additional load is placed on components such as memory. Linear complexity means that the time taken is proportional to the size of the input, and will grow at a steady rate as input size increases. Constant complexity means that no matter the size of input, the time / load will always be the same, regardless of the data being processed. Exponential complexity grows very quickly as input increases, whilst logarithmic increases, though at a Algorithm 1 takes pretty poorly - though linear time is acceptable for small inputs, it quickly becomes unreasonable for larger data sets, and the exponential space complexity will mean the data quickly requires much more processing power for just a small increase in input volume. Though Algorithm 2's constant space complexity is ideal, the exponential time complexity means many of the larger possible data inputs will take exceptionally long to complete, and so is impractical and shouldn't be used. Since algorithm 3 has both time and space complexity as logarithmic it will scale well, as input volume can increase greatly with well, as small decline in, and therefore must gradually be noticeable, possibly as soon as This response clearly demonstrates that a high scoring Level 3 response is possible within the space provided. The second page of the response shows clear AO3 evaluation between the merits of the three different algorithms and the scenario.

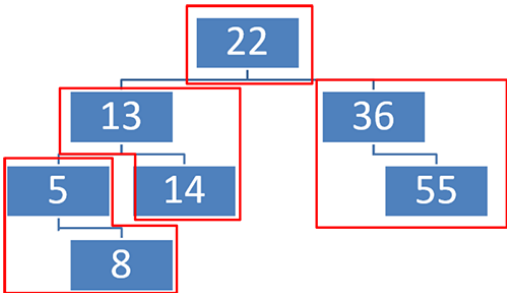
Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
	b	i	1 mark each to max 5 • The data list is split into two lists • These sublists continue to be (recursively) split... • ...until each sublist is one item • The first element in two different sublists is compared... • ...the smaller item is then selected... • ...and written to a new list • ...until both sublists fully merged • Repeated until all sorted sublists are recombined	5	Allow array/list as equivalent <u>Examiner's Comments</u> Many candidates were not precise in describing how the initial data set was repeatedly halved and did not explain how the data set was actually broken down into individual items. Very few candidates could accurately describe how the merge phase of a merge sort combines two separate ordered lists together, and frequently this was omitted altogether. A number of candidates continued to demonstrate the misconception that data is sorted within sub-lists, rather than by the actual merging of two separate ordered lists together.  Misconception Many candidates continue to think that a merge sort performs sorting of data within lists. Merge sort performs the sorting part of the operation when it merges together two separate sub-lists that are already in a sorted order. This is done with the use of a pointer to each of the two sub-lists that acts as a placeholder. The two positions in the separate lists are compared, and the smaller item is added to the new sorted list and the pointer is incremented. The merging process then repeats until the sub-lists have been completely merged.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
		ii	1 mark for benefit e.g. • More efficient time complexity (for large data sets) // takes fewer steps to sort the data • Time complexity $O(n \log n)$, rather than $O(n^2)$ • Uses divide and conquer • Can apply concurrent processing to reduce sorting time 1 mark for drawback e.g. • More difficult to implement // needs more complex code • Less efficient space complexity // uses more memory with more data items • Space complexity of $O(n)$/linear, rather than $O(1)$ / constant • Merge sort is always $O(n \log 2n)$ whereas the best case for bubble sort is $O(n)$	2	<u>Examiner's Comments</u> Most candidates recognised that merge sort is typically quicker than bubble sort, but there were some unqualified answers. Clear responses were phrased in terms of Big O complexities or the numbers of data items to be sorted. Some responses were phrased in terms of Big O complexities or the numbers of
			Total	16	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
3	a	i	1 mark each to max 2 • It is a hierarchical structure / not directed • Data is stored in nodes • Nodes are linked by branches/edges • It has a root node • Each node has zero or more nodes 'beneath' it // nodes can link to child nodes • It has leaf nodes / nodes without any lower nodes are leaf nodes • It has no cycles/loops (distinguishing it from a graph)	2	<u>Examiner's Comments</u> This question was generally well answered by most candidates with the most common answers including different types of nodes such as root, child and leaf nodes. Some candidates used vague terminology or used terminology more commonly associated with graphs such as vertices instead of nodes, or edges instead of branches. Imprecise language such as undirected/non-directed was used whereas it would have been clearer to have indicated that trees are hierarchical structures that are rooted. Some candidates erroneously confused binary trees with general trees stating that nodes could only have a maximum of two child nodes, as the tree in this question was not specifically limited to the special case of a binary tree.
		ii	1 mark each • Root node 22 at the start • 13 and 14 in correct order • 5 and 8 in correct order • 36 and 55 in correct order 	4	Do not allow nodes to be drawn downwards. <u>Examiner's Comments</u> Most candidates gained some marks, with many gaining full marks. Occasionally the left/right ordering of child nodes was unclear, so marks could not be given in such instances.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
		iii	1 mark each • Search/traverse tree until the required node is found • Set the parent node pointer to the leaf node to null • Add the deleted node to the free storage list // leave for garbage clear up	2	<u>Examiner's Comments</u> This question was generally less successfully answered by many candidates. There was considerable vagueness in some responses such as 'finding the end node and removing it'. This left questions such as 'Which end node?', 'Removed how?' for the examiner to infer, so were too vague to gain marks. When candidates identified the need to locate the leaf node, most omitted the need for the parent node pointer to be set to null to break the link. Very few candidates indicated that a deleted node would then be added to a free list or that the memory freed could be retrieved via garbage collection.
		iv	1 mark each to max 4 • Check if the root node is equal to search value and if so.... • ...return/output/report found • If value is less than root node take left subtree • If value is greater than root node take right subtree • Repeat process with the subtree... • ...until search value is found • ...until no more branches can be travelled.	4	<u>Examiner's Comments</u> This question was generally well answered by many candidates who appreciated that a Binary Search Tree (BST) could be efficiently searched because it is ordered, rather than traversed. Weaker candidate often confused the search of a BST structure with traversal algorithms such as breadth first or depth first traversals.
		v	1 mark each • Visiting A first... • ...Then visiting F, C... • ...Then visiting L, T, P... • ...Visiting H last Solution: A, F,C, L,T,P, H	4	<u>Examiner's Comments</u> Some candidates confused Depth First Search with other traversal algorithms, most commonly Breadth First Search, but this question was generally answered well by most candidates. Another common mistake was to output the nodes in the order in which they were first encountered rather than the order in which they would be visited/output.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance																													
		vi	1 mark each to max 2 • When a leaf node is reached... • ...the traversal backtracks to the leaf's parent node • ...backtracks to last node with unvisited children	2	Candidates may use an example from the tree in 1a(v) to illustrate their response. If an answer gives implementational detail of how a stack is used, map to the bullet points given. Examiner's Comments Some candidates described the term 'backtracking' in general, so did not answer the question, which required a response in the context of a depth first tree traversal. Many candidates thought that backtracking always began at the leftmost node, which is not true, it starts when a leaf node is reached. However, a pleasing number of candidates did identify the first example of backtracking in the given tree, from leaf A to parent node C.																													
	b		1 mark for final path A, D, G 1 mark for final distance 14 1 mark for each SECTION or equivalent working shown. <table><tr><th>N o d e</th><th>Distance travelled</th><th>Previous node</th><th>Marking Guidance</th></tr><tr><td>A</td><td>0</td><td>- / N/A / blank / None</td><td rowspan="2">1 Mark</td></tr><tr><td>B</td><td>5</td><td>A</td></tr><tr><td>C</td><td>2</td><td>A</td><td rowspan="2">1 Mark</td></tr><tr><td>D</td><td>10</td><td>A</td></tr><tr><td>E</td><td>7</td><td>B</td><td rowspan="2">1 Mark</td></tr><tr><td>F</td><td>15</td><td>E</td></tr><tr><td>G</td><td>~~19~~ 14</td><td>~~E~~ D</td><td>1 Mark</td></tr></table>	N o d e	Distance travelled	Previous node	Marking Guidance	A	0	- / N/A / blank / None	1 Mark	B	5	A	C	2	A	1 Mark	D	10	A	E	7	B	1 Mark	F	15	E	G	~~19~~ 14	~~E~~ D	1 Mark	6	Nodes should appear in the alphabetical order given if candidates add them as the algorithm progresses but allow other orderings of the nodes. For the last mark in the table there must be a clear indication that G 19 from E is overwritten by G 14 from D. Allow equivalent discrete maths approach or textual description. Check diagram for annotations / solution. Examiner's Comments Candidates answering 'by inspection' could gain 2 marks for the final path and distance, and many less successful responses thus scored 1 or 2 marks by doing so, demonstrating little real understanding of how Dijkstra's algorithm operates. A common error was the incorrect solution ACDG that showed ignorance of not continuing the search from the node with the least distance travelled that has not already been marked as visited. For full marks there had to be an indication that the first path to G (19 from E) was overwritten by G (14 from D) for the last mark in the table. Relatively few candidates demonstrated this.
N o d e	Distance travelled	Previous node	Marking Guidance																															
A	0	- / N/A / blank / None	1 Mark																															
B	5	A																																
C	2	A	1 Mark																															
D	10	A																																
E	7	B	1 Mark																															
F	15	E																																
G	~~19~~ 14	~~E~~ D	1 Mark																															

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			Total	24	

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
4	a	1 mark each • <code>headPointer</code>: To indicate the first element in the list • <code>freeListPointer</code>: To indicate the next index to store data in (the <code>freeList</code>)	2	<u>Examiner's Comments</u> While many candidates did gain full marks, a noticeable number of candidates struggled with the use of technical language in this question. Where a candidate's intention or meaning was clear, for example, '<code>headPointer</code> is the first item' or '<code>freeListPointer</code> is the first free space', marks were given. Some candidates did not show an understanding that these pointers represented the start of two separate linked lists within the static array structure.
	b	It doesn't point to another node Indicates the end of the linked list	1	<u>Examiner's Comments</u> This question was generally well answered, with many candidates gaining marks. A generic 'no assigned value' was not given marks, as the response had to be answered in the context of the linked list, so end of list/not pointing to another node was required, rather than stating pointing to nothing.
	c	• first output red... • ...remainder of list correct e.g. red blue grey green purple orange	2	<u>Examiner's Comments</u> Again, this question was generally well answered by most candidates. The most common erroneous answer was 'Blue' as it was the first item in Figure 3 given at index 0. This was invariably given as a response when candidates did not know what the function of the <code>headPointer</code> was.

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
	d	1 mark each to max 4 Check space available in the free list • Check to make sure <code>freeListPointer</code> is not Null Add new data item to first free space in free list • Insert new data item at index <code>freeListPointer</code> (index 4) Append e.g. • Traverse to / locate the end of the list (index 3 'orange') • Set the pointer of the last item in the linked list to <code>freeListPointer</code> (pointer at index 3 'orange' changes from Null to 4)... • ... update <code>freeListPointer</code> to the location that new data item pointer is pointing to at present. (<code>freeListPointer</code> changes from 4 to 5) • ... update pointer from new data item to Null (index 4 pointer changes from 5 to Null) Prepend e.g. • Update <code>freeListPointer</code> to point to the location that the pointer from the first item in the free list is pointing to (<code>freeListPointer</code> changes from 4 to 5) • ... Update pointer from new data item to <code>headPointer</code> (index 4 pointer changes from 5 to 1) • ... Update <code>headPointer</code> to the index of new data item (<code>headPointer</code> changes from 1 to 4)	4	Note descriptions could be for either appending an item to the end of the current list or prepending it to the start. There are different ways to achieve this. Allow answers that illustrate solutions by example from the table in Fig 3 at the start of the question. Responses must refer to the relevant pointers or give clear exemplifications. <u>Examiner's Comments</u> More successful responses gave good clear examples to illustrate a potential sequence of operations to describe the process, and many therefore achieved 3 or 4 marks. For example 'new item added to location 4 indicated by <code>FreeListPointer</code> and its pointer set to Null; Existing list searched from <code>headPointer</code> to its end at <i>Orange</i> at index 3, and its pointer updated to the new last item in the list at index 4.' Some candidates lacked appreciation that this was a static record structure in this scenario, so locations from the free list had to be used. Many candidates were unclear as to how the end of the current linked list was found, and just gave answers from that point onward without saying how it was found by traversing to the end of the existing list. There were relatively few prepend type solutions, but they were accepted as valid where seen.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
	e		1 mark for each statement <pre> function findNode(toFind, headPointer, linkedList) currentNode = headPointer while(currentNode != NULL) if linkedList[currentNode].data == toFind then return currentNode else currentNode = linkedList[currentNode].pointer endif endwhile return -1 endfunction </pre>	5	Ignore case of identifiers in pseudocode Only penalise excessive spaces within identifier names if obvious. <u>Examiner's Comments</u> This question required exact answers only. Many candidates gained some marks, and there was a good distribution of marks. More successful programmers tended to get most of the marks available.
			Total	14	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
5	a		2005	1	Examiner's Comments Most candidates correctly identified the root node as 2005.
	b		1 mark for each to max 2 • 1500 • 1952 • 2000 • 2100 • 2560	2	Examiner's Comments Most candidates correctly identified valid leaf nodes, but a significant number erroneously gave 1920 and 2350 as the child nodes of the root instead of identifying leaf nodes.
	c		Mark Band 3 – High level (7-9 marks) The candidate demonstrates a thorough knowledge and understanding of search traversals; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. <i>There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.</i> Mark Band 2 – Mid level (4-6 marks) The candidate demonstrates reasonable knowledge and understanding of search traversals; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. <i>There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.</i> Mark Band 1 – Low Level (1-3 marks) The candidate demonstrates a basic knowledge of search traversals with limited understanding shown; the material	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	AO1: Knowledge and Understanding Indicative content • Breadth first takes first value then visits all nodes connected to it. It then takes all nodes connected to those nodes. • Depth first goes to the left node, this becomes a new tree. It continues going to the left until a leaf. It then returns this, then goes right and repeats from the start. Follow left, follow right, take root. AO2: Application • Breadth will return 2005 1920 2350 1500 1985 2100 2560 (1420) 1952 2000 (2050) (2780) (2600) • Post-order / Depth will return (1420) 1500 1952 2000 1985 1920 (2050) 2100 (2600) (2780) 2560 2350 2005 AO3: Evaluation Evaluations may vary and include one or more of the following points: • Breadth is more efficient when the data searched for is closer to the root. • Depth is more efficient when data to be search for is further down. • Depth memory requirement is linear • Depth can be written recursively to aid understanding. • Breadth in general is better time complexity • In large trees depth may never return a value Candidates are not expected to know the complexities for the search traversals, however credit should be awarded if candidates choose to include these.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. <i>The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.</i> 0 marks No attempt to answer the question or response is not worthy of credit.		Limit to band 2 if there is no evaluation of BFS/DFS <u>Examiner's Comments</u> Many candidates achieved some marks, but a few did confuse Breadth-first search (BFS) and Depth-first search (DFS), getting them the wrong way round. Most had a much clearer understanding of BFS than DFS and were able to score marks in Level 1 for showing how a BFS would be carried out. Fewer managed to accurately show how a DFS would work. While there was an understanding that DFS would traverse leftward to the lowest leftmost node, descriptions of back tracking were not given as many marks. A common mistake made with DFS was to output the nodes in the order they were first visited rather than the order in which they are output. Few could describe the mechanics of the algorithms with BFS using a queue and DFS using a stack. Those who did achieved the top of Level 2. Very few candidates could give exemplar uses of BFS or DFS (e.g. deleting nodes in a tree) or compare cases where they might be used (e.g. distance of target node(s) from root), with very little evaluation. This meant there were very few Level 3 responses seen.
			Total	12	

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
6	a	1 mark for each completed statement <pre> arrayLength = 50 // numberArray.length tempValue = 0 do flag = false for y = 0 to arrayLength - 2 if numberArray[y] > numberArray[y + 1] then tempValue = numberArray[y] numberArray[y] = numberArray[y + 1] numberArray[y + 1] = tempValue flag = true endif next y until flag == false </pre>	5	Note if numberArray - 1 // 49 used, then for loop for y will need to be 0 to arrayLength - 1 Allow other suitable valid identifier in place of tempValue e.g. temp Examiner's Comments Many candidates scored at least 1 mark for correctly initialising the arrayLength, although some erroneous length.arrayLength rather than arrayLength.length answers were seen. If candidates are assuming the existence of inbuilt methods they should reference them in the correct OOP way. Very few candidates achieved the second marking point for the loop. A very common off-by-one error was seen with the value 1 given. If arrayLength was set to 50 this would cause a run time error for an out-of-bounds reference when the loop ran. Some candidates tried to swap the array values directly in a Pythonic style that was not suitable for a pseudocode solution in context, and some used incorrectly formatted variable identifiers. Variable identifiers Valid identifiers must be single words. In a number of instances, it was clear that tempValue was given as temp Value. Where candidates give an answer that clearly has spaces within an intended identifier name no marks will be given.
	b	1 mark for each stage shown - Splitting into individual items 2 12 1 9 3 5 15 7 - Combining in pairs 2 12 1 9 3 5 7 15 - Merge pairs 1 2 9 12 3 5 7 15 Merge for final 1 2 3 5 7 9 12 15	4	Do not award a mark for the final stage, unless candidate has shown the previous sorting stage(s). Examiner's Comments Many candidates presented very clear solutions that used the values in numberArray to construct clearly annotated diagrams. Textual prose responses tended to either not refer at all to the given data set, or mentioned it only partially, so lost marks. There was occasional confusion with other sorting algorithms, but this was seen relatively

Mark Scheme

Question	Answer/Indicative content	Marks	Guidance
			infrequently, Where errors were made candidates did not split the data to the atomised level. Many did not fully understand how a merge sort works by merging two separate lists of ordered values together but performed in-place sorts in sub-lists. Misconception The merge phase in a merge sort takes two separate lists that are already ordered. Values in those two separate lists are taken and merged in sequence to form a new list. E.g. List 1 [1, 9] and List 2 [2, 10] are taken and merged 1 from List 1, 2 from List 2, 9 from List 1 and then 10 from List 2, into a new List. Values in lists [1, 9] and [2, 10] are not placed in a list [1, 9, 2, 10] and then sorted in-situ. A significant number of candidates thought that a merge sort split lists up until values were in pairs and showed [15, 7] going directly to [7, 15] (i.e. an in-situ sort) without first being atomised into two separate lists of [15] and [7] and then being merged to give [7, 15]. Exemplar 2 Candidate responses to questions that require 'showing' how data sets are ordered by sorting algorithms are best tackled by using clearly annotated diagrams such as this exemplar response.

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
	c	Mark Band 3 – High level (9-12 marks) The candidate demonstrates a thorough knowledge and understanding of sorting algorithms; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. <i>There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.</i> Mark Band 2 – Mid level (5-8 marks) The candidate demonstrates reasonable knowledge and understanding of sorting algorithms; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. <i>There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.</i> Mark Band 1 – Low Level (1-4 marks) The candidate demonstrates a basic knowledge of sorting algorithms with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. <i>The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be</i>	12 AO1.1 (3) AO1.2 (3) AO2.1 (3) AO3.3 (3)	AO1: Knowledge and Understanding Indicative content • Merge sort splits data into individual lists and merges • Insertion makes first value sorted list, then inserts each item into the sorted list • Bubble sort looks through each item in turn, number of items times AO2: Application • Merge uses more memory as new lists are needed. Insertion and Bubble need constant memory. • Bubble and Insertion have the best best-times, both $O(n)$ because they run through the data once. merge sort requires a minimum number of stages so best case is longer ($O(n \log(n))$) • Merge average is the same as best. Insertion and Bubble has average $O(n^2)$. • Worst time merge has same as best and average because same number of stages are needed. Bubble sort and insertion all have worse $O(n^2)$ AO3: Evaluation • There are a small number of elements (10) therefore a bubble sort of insertion would be better space wise because no further space is needed. • Merge would not need excessive amounts of more memory as there are only a small number of elements. • Time complexity, there is a small number of elements therefore Bubble and Insertion may be preferable. Differences are unlikely to be significant, so either would be more appropriate. <u>Examiner's Comments</u> Most candidates could describe the basic elements of each of the bubble, merge and insertion sort, although some had difficulty remembering insertion sort and confused it with quick sort.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			<i>clear.</i> 0 marks No attempt to answer the question or response is not worthy of credit.		Many candidates struggled to show accurate knowledge of the Big O time complexity values for best/average/worst case for the three algorithms, which is a learnt response. Some did give very good examples of where bubble and insertion sort would give best/worst case times. Very few candidates could cite space complexity, and where they did a number thought bubble/insertion were space complexity $O(n)$ because there are n elements rather than $O(1)$ which means constant with no extra memory overhead. There was some evaluation of the number of items being sorted in most cases. Overall, many responses were clustered in Level 2, but a pleasing number of candidates achieved Level 3 with clear and detailed descriptions, accurate Big O values, and an evaluation of the size of the data set being sorted.
			Total	21	

Mark Scheme

Question			Answer/Indicative content				Marks	Guidance																														
7		i	<table><tr><th>Node</th><th>Distance travelled</th><th>Previous node</th><th>Marking Guidance</th></tr><tr><td>A</td><td>0 / -</td><td>N/A / -</td><td>1 Mark</td></tr><tr><td>B</td><td>3</td><td>A</td><td rowspan="2">1 Mark</td></tr><tr><td>C</td><td>13</td><td>E</td></tr><tr><td>D</td><td>10</td><td>B</td><td rowspan="2">1 Mark</td></tr><tr><td>E</td><td>6</td><td>B</td></tr><tr><td>F</td><td>9</td><td>E</td><td rowspan="2">1 Mark</td></tr><tr><td>G</td><td>16</td><td>F</td></tr><tr><td>H</td><td>24 19</td><td>⊘ G</td><td>1 Mark</td></tr></table>	Node	Distance travelled	Previous node	Marking Guidance	A	0 / -	N/A / -	1 Mark	B	3	A	1 Mark	C	13	E	D	10	B	1 Mark	E	6	B	F	9	E	1 Mark	G	16	F	H	24 19	⊘ G	1 Mark	6	Order of previous nodes visited must be clear Note that nodes in the table do not have to be given in alphabetical order by candidates <u>Examiner's Comments</u> Most candidates gave the final path and the total distance correctly by inspection if nothing else. All nodes are initially set to infinity, so A is updated to 0 and has a distance 0 from A as the start node and many candidates missed this. Those who gave an answer by inspection wrote ABEFGH without knowing Dijkstra's algorithm but gained some marks by giving the distances to BEFGH along the way, but distances to nodes C and D were omitted. Few candidates clearly showed that the initial calculation for the path distance to H (from D, distance 21) was later updated and overwritten with the more optimal path length from G with distance 19.
			Node	Distance travelled	Previous node	Marking Guidance																																
			A	0 / -	N/A / -	1 Mark																																
			B	3	A	1 Mark																																
			C	13	E																																	
			D	10	B	1 Mark																																
			E	6	B																																	
			F	9	E	1 Mark																																
			G	16	F																																	
			H	24 19	⊘ G	1 Mark																																
Final Path = A,B,E,F,G,H, Distance = 19 (1 Mark)																																						
ii		1 mark per bullet	2	Must contain a similarity and a difference for both marks. <u>Examiner's Comments</u> Those candidates who scored well in c(i) frequently gained full marks for describing the similarities and differences between Dijkstra's algorithm and the A* algorithm. The most common responses were that both give the shortest path and that A* uses heuristics.																																		
Similarity:																																						
- Both always find the shortest route - Both are pathfinding algorithms																																						
Differences:																																						
		- A* is (usually) more efficient // dijkstra's is (usually) slower - A* uses heuristics to find a solution faster // Dijkstra's does not use heuristics																																				
Total			8																																			

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance																																																
8		i	1 mark per bullet to max 2 e.g: • A rule of thumb / estimate / guess • That estimates the distance / cost from each node to the destination node • To speed up the process of finding a solution • ...by identify which paths to follow first	2 AO1.1 (1) AO1.2 (1)																																																	
		ii	<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>Node</th><th>Distance travelled</th><th>Heuristic</th><th>Distance travelled + Heuristic</th><th>Previous node</th><th>MARKING GUIDANCE</th></tr> </thead> <tbody> <tr> <td>A (✓)</td><td>0</td><td>90</td><td>90</td><td></td><td>1 MARK</td></tr> <tr> <td>B (✓)</td><td>≤ 21</td><td>80</td><td>101</td><td>A</td><td>1 MARK</td></tr> <tr> <td>C (✓)</td><td>≤ 42</td><td>65</td><td>107</td><td>A</td><td>1 MARK</td></tr> <tr> <td>D (✓)</td><td>∞ 42+12=54</td><td>50</td><td>104</td><td>C</td><td>1 MARK</td></tr> <tr> <td>E</td><td>∞ 21+40=61</td><td>50</td><td>111</td><td>B</td><td>1 MARK</td></tr> <tr> <td>F (✓)</td><td>∞ 42+12+23=77</td><td>30</td><td>107</td><td>D</td><td>1 MARK</td></tr> <tr> <td>G</td><td>∞ 42+12+23+33=110</td><td>0</td><td>110</td><td>F</td><td>1 MARK</td></tr> </tbody> </table> Final path = A,C,D,F,G and Distance = 110 (1 Mark)	Node	Distance travelled	Heuristic	Distance travelled + Heuristic	Previous node	MARKING GUIDANCE	A (✓)	0	90	90		1 MARK	B (✓)	≤ 21	80	101	A	1 MARK	C (✓)	≤ 42	65	107	A	1 MARK	D (✓)	∞ 42+12=54	50	104	C	1 MARK	E	∞ 21+40=61	50	111	B	1 MARK	F (✓)	∞ 42+12+23=77	30	107	D	1 MARK	G	∞ 42+12+23+33=110	0	110	F	1 MARK	8 AO1.2 (3) AO2.1 (3) AO2.2 (2)	
Node	Distance travelled	Heuristic	Distance travelled + Heuristic	Previous node	MARKING GUIDANCE																																																
A (✓)	0	90	90		1 MARK																																																
B (✓)	≤ 21	80	101	A	1 MARK																																																
C (✓)	≤ 42	65	107	A	1 MARK																																																
D (✓)	∞ 42+12=54	50	104	C	1 MARK																																																
E	∞ 21+40=61	50	111	B	1 MARK																																																
F (✓)	∞ 42+12+23=77	30	107	D	1 MARK																																																
G	∞ 42+12+23+33=110	0	110	F	1 MARK																																																
			Total	10																																																	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
9		i	1 mark per bullet to max 3 • Search the tree to find the location of Node E // by example of search • Replace the content of node E with blank/null/equivalent • Make node A point to the node H • Add node E to the empty node list	3 AO1.2 (3)	
		ii	1 mark per bullet to max 3 • Search the tree to find the location of node G // by example of search • Create a new node with value K • Add a pointer from node G to the new node • Make node K point to null/equivalent	3 AO1.2 (3)	
			Total	6	

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
10	a	Mark Band 3 – High level (7-9 marks) The candidate demonstrates a thorough knowledge and understanding of big O and sorting algorithms; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up the use of the sorting algorithms which results in a supported and realistic judgment as to whether it is possible to use them in this context. <i>There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.</i> Mark Band 2 – Mid level (4-6 marks) The candidate demonstrates reasonable knowledge and understanding of big O and sorting algorithms; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether it is possible to use the sorting algorithms in this context. <i>There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence</i> Mark Band 1 – Low Level (1-3 marks) The candidate demonstrates a basic knowledge of big O and sorting algorithms with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. <i>The information is basic and communicated</i>	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	AO1: Knowledge and Understanding Indicative content O(1) - Constant space, does not change O(n) - Linear - Same as number of elements - As number of elements increases so does the time/space O(n²) - polynomial - As number of elements increases, time/space increases by *n O(n log(n)) - Linearithmic AO2: Application - Space: Merge sort will require more memory usage as the number of elements increases. Insertion will not require any more space than original. Quick will increase but not as much as merge. - Best time: Insertion increases at the same rate as the number of elements. Quick and merge increase at greater rate - Worst time: insertion and quick increase significantly by n for each additional item. Merge sort increases less per element. - Log more appropriate for large number of elements AO3: Evaluation e.g. - Small array – space is not important. Few number of elements. Look for consistency. - Large array therefore memory important – could remove merge as inappropriate. Logarithmic more efficient.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			<i>in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.</i> 0 marks No attempt to answer the question or response is not worthy of credit.		
	b		1 mark per bullet for description to max 6 • Compare each pair of adjacent elements • If they are not in the correct order then swap the elements • If they are in the correct order then do no swap elements • When you read the end of the array return to the start • Repeat n elements time • Set a flag to be false whenever a swap is made • ...repeat the loop if the flag is not false	6 AO1.1 (2) AO1.2 (4)	
			Total	15	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance																																																																																	
11			1 mark for final solution, max 5 for showing the stages • Mark A as the current node initially • Record B = 1, C = 2 (mark A as visited) • Record E = 5 (and mark B as visited) • (Record D = 3, F = 5 (and mark B as visited) • Change E to 4 (overriding previous value, and mark D as visited) • Record G = 6 (and mark E as visited) • ...Do not change G as greater than current (mark F as visited) • (G as visited) H = 10 (Mark G as visited) • Solution: A-C-D-E-G-H path length 10 <table><tr><th>Node</th><th>Visited</th><th>From A</th><th>Previous Node</th><th></th></tr><tr><td>A</td><td>✓</td><td>0</td><td>-</td><td>1 Mark</td></tr><tr><td>B</td><td>✓</td><td>1</td><td>A</td><td>1 Mark</td></tr><tr><td>C</td><td>✓</td><td>2</td><td>A</td><td></td></tr><tr><td>D</td><td>✓</td><td>3</td><td>C</td><td>1 Mark</td></tr><tr><td>F</td><td>✓</td><td>5</td><td>C</td><td></td></tr><tr><td>E</td><td>✓</td><td>4</td><td>B D</td><td>2 Marks Initial visit, plus override values</td></tr><tr><td>G</td><td>✓</td><td>6</td><td>E</td><td>1 Mark</td></tr><tr><td>H</td><td></td><td>10</td><td>G</td><td>1 Mark</td></tr></table>	Node	Visited	From A	Previous Node		A	✓	0	-	1 Mark	B	✓	1	A	1 Mark	C	✓	2	A		D	✓	3	C	1 Mark	F	✓	5	C		E	✓	4	B D	2 Marks Initial visit, plus override values	G	✓	6	E	1 Mark	H		10	G	1 Mark	6 AO1.2 (1) AO2.2 (3) AO2.2 (2)	Guidance – 1 mark only for stating the solution of A-C-D-E-G-H length 10 <u>Examiner's Comments</u> Exemplar 2 Show how Dijkstra's algorithm would find the shortest path from A to H. <table><tr><th>Node</th><th>Visited</th><th>Shortest distance from start</th><th>Previous node</th></tr><tr><td>A</td><td>✓</td><td>0</td><td>-</td></tr><tr><td>B</td><td>✓</td><td>1</td><td>A</td></tr><tr><td>C</td><td>✓</td><td>2</td><td>A</td></tr><tr><td>D</td><td>✓</td><td>3</td><td>C</td></tr><tr><td>E</td><td>✓</td><td>4</td><td>D</td></tr><tr><td>F</td><td>✓</td><td>5</td><td>C</td></tr><tr><td>G</td><td>✓</td><td>6</td><td>E</td></tr><tr><td>H</td><td>✓</td><td>10</td><td>G</td></tr></table> Shortest path is A-C-D-E-G-H and takes a total of 10 moves. [6] Dijkstra's shortest path algorithm has appeared in previous sessions and candidates generally had a good grasp of the principles of the algorithm. Verbose text often made some responses difficult to follow. The clearest responses (as exemplified) tabulated the steps in the algorithm.	Node	Visited	Shortest distance from start	Previous node	A	✓	0	-	B	✓	1	A	C	✓	2	A	D	✓	3	C	E	✓	4	D	F	✓	5	C	G	✓	6	E	H	✓	10	G
Node	Visited	From A	Previous Node																																																																																			
A	✓	0	-	1 Mark																																																																																		
B	✓	1	A	1 Mark																																																																																		
C	✓	2	A																																																																																			
D	✓	3	C	1 Mark																																																																																		
F	✓	5	C																																																																																			
E	✓	4	B D	2 Marks Initial visit, plus override values																																																																																		
G	✓	6	E	1 Mark																																																																																		
H		10	G	1 Mark																																																																																		
Node	Visited	Shortest distance from start	Previous node																																																																																			
A	✓	0	-																																																																																			
B	✓	1	A																																																																																			
C	✓	2	A																																																																																			
D	✓	3	C																																																																																			
E	✓	4	D																																																																																			
F	✓	5	C																																																																																			
G	✓	6	E																																																																																			
H	✓	10	G																																																																																			
			Total	6																																																																																		
12			1 mark per bullet • Merge sort might create a new array each time it splits and merges / often implemented recursively which places additional data on the stack • Insertion sort does not use any additional arrays//Insertion sort is an in-place algorithm.	2 AO1.2 (2)	<u>Examiner's Comments</u> Many candidates recognised that merge sort could generate an additional array each time a list was split. Fewer could explain that insertion sort worked in-situ and has a space complexity O(1).																																																																																	
			Total	2																																																																																		

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
13			1 mark per bullet • Queue outputs data in a First In First Out fashion • It will retrieve the temperature values in the order they were recorded or • Stack outputs the data in a Last In First Out fashion • It will retrieve the temperature values in the reverse of the order they were recorded	2 AO1.2 (1) AO2.2 (1)	Mark Point 1 is the definition Mark Point 2 is for context of the temperature values <u>Examiner's Comments</u> The majority of candidates were able to define a queue as a FIFO structure. Most of those who did so were able to successfully go on to relate this to the context of the question, which was to be able to retrieve the temperature values in the same order as they had originally been recorded. A number of candidates answered equally well by explaining why a stack would not have been appropriate.
			Total	2	