1.

(i) Describe how a merge sort differs from a bubble sort.

---

[4]

(ii) Name **two** sorting algorithms, other than a bubble sort and merge sort.

1

2

---

[2]

2. A 1-dimensional array stores a set of numbered cards from 0 to 7. An example of this data is shown in Fig in 4.1

2	0	1	7	4	3	5	6

Fig 4.1

* Two sorting algorithms the programmer could have used are a merge sort and bubble sort.

The worst case scenario for Merge is O(n log(n)) and for Bubble is O(n^2).

Compare the use of a merge sort and a bubble sort on this array, evaluating the performance of each sort, making reference to the worse case Big O notation.

[9]

3(a). A programmer needs to sort an array of numeric data using an insertion sort.

(i) The following, incomplete, algorithm performs an insertion sort.

Complete the algorithm.

```
procedure sortit(dataArray, lastIndex)
  for x = 1 to lastIndex
    currentData = dataArray[……]
    position = x
    while (position > 0 AND dataArray[position-1] > currentData)
      dataArray[position]  = dataArray[……]
      position = position - 1
    endwhile

    dataArray[position] = ……
  next x
endprocedure
```

[3]

(ii) Show how an insertion sort would sort the following data:

6	1	15	12	5	6	9

[6]

(b).

(i) Using Big-O notation state the best case complexity of insertion sort.

---

[1]

(ii) Explain what your answer to part **(b)(i)** means.

---

[3]

4. A 1-dimensional array stores a set of numbered cards from 0 to 7. An example of this data is shown in Fig in 4.1

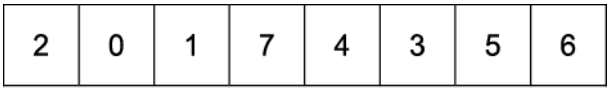

Fig 4.1

A programmer is writing a computer program to sort the cards into the correct order (0 to 7).

(i) Show how an insertion sort would sort the array in Fig 4.1 into the correct order. Draw the array after each move.

[3]

(ii) Describe how a quick sort algorithm works with the data in Fig 4.2.

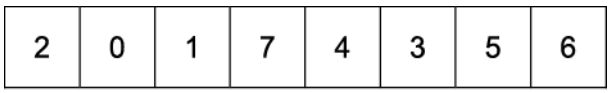

Fig 4.2

[6]

5. A programmer needs to sort an array of numeric data using an insertion sort.

The number of data items in the array is continually increasing.

Insertion sort has a worst case time complexity of $O(n^2)$ and space complexity of O(1).

An alternative sorting algorithm that could be used is bubble sort which also has a worst case time complexity of $O(n^2)$ and space complexity of O(1).

Briefly outline how the bubble sort algorithm works. Discuss the relationship between the complexities and the two sorting algorithms and justify which of the two algorithms is best suited to sorting the array. **[9]**

END OF QUESTION PAPER

Question			Answer/Indicative content	Marks	Guidance
1		i	1 mark per bullet to max 4 • Merge sort splits the data • Merge sorts the split data as it is put back together • Bubble moves through the data in a linear way • Bubble moves through the data repeatedly • Merge is more efficient with larger volumes of data to sort • Merge may require more memory space	4	Allow points by demonstration/example
		ii	1 mark per example e.g. • Insertion • Quick	2	
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
2			**Mark Band 3 – High level** **(7–9 marks)** The candidate demonstrates a thorough knowledge and understanding of merge and bubble sorts; 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. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4–6 marks)** The candidate demonstrates reasonable knoledge and understanding of merge and bubble sorts; 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. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of merge and bubble sorts 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.	9	AO1: Knowledge and Understanding Indicative content • Merge sort uses sub-lists • Bubble sort uses a temp element • Bubble sort moves through the list repeatedly • Merge sort divides the list into smaller lists • Merge is a recursive algorithm • Worst case is logarithmic, scales up well • Worst case is exponential, does not scale up well AO2: Application • Small data set • Few changes are needed • Demonstrates use of merge and / or bubble on the array • Calculations of average speed / best speed / worse speed AO3: Evaluation Candidates will need to evaluate the benefits and drawbacks of each sorting algorithm e.g. • Merge is fast on large data sets • Bubble is intuitive (easier to program) • Both are fast (or even) on smaller data sets • Bubble's average speed is worse than merge • Bubble will be easier to write for such a small data set • Accept argument for either way as long as justified

Question			Answer/Indicative content	Marks	Guidance
			Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* 0 marks No attempt to answer the question or response is not worthy of credit.		
			Total	9	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>3</td><td>a</td><td>i</td><td>1 mark for each correct item in bold

<pre>
procedure sortit(dataArray, lastIndex)
   for x = 1 to lastIndex
      currentData = dataArray[x]
      position = x
      while (position > 0 AND dataArray[x - 1] > currentData)
         dataArray[position] = dataArray[position-1]
         position = position - 1
      endwhile
      dataArray[position] = currentData

   next x
endprocedure
</pre></td><td>3
AO1.1
(3)</td><td>answers must be in the correct case as given
e.g. current<u>D</u>ata

Examiner's Comment:
Many candidates found it difficult to apply the logic required to calculate the correct solution. Stronger candidates could do so even if they did not know the algorithm for insertion sort.</td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark for contents of each row in table

<table>
<tr><td>6</td><td>1</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td></td><td></td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>6 is the sorted list
1 is the compared to sorted list
1 is put in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>15 is compared
15 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>12</td><td>15</td><td>5</td><td>6</td><td>9</td><td>12 is compared
12 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>12</td><td>15</td><td>6</td><td>9</td><td>5 is compared
5 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>12</td><td>15</td><td>9</td><td>6 is compared
6 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>9</td><td>12</td><td>15</td><td>9 is compared and put in place</td><td>1</td></tr>
</table></td><td>6
AO2.1
(6)</td><td>… each row is dependent upon the preceding row being correct

Examiner's Comment:
Some candidates confused insertion sort with other sorting algorithms, but many candidates gave good answers in diagrammatic form. Answers in diagrammatic form after each pass of the loop were often far clearer than prose descriptions. This form of answer should be encouraged.</td></tr>
<tr><td></td><td>b</td><td>i</td><td>O(n)</td><td>1
AO1.1
(1)</td><td></td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark per bullet to max 3

• The best case is for a sorted list (O(n))
• As the number of elements increases
• … the number of steps increases in a <u>linear</u> fashion</td><td>3
AO1.2
(3)</td><td>B(ii) dependent upon b(i) being correct
i.e. answers for O(n) only

Accept appropriate graph for bullet points 2 and 3

Examiner's Comment:
Whilst many candidates had some knowledge of 'Big O' notation fewer could apply it correctly within the context given.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
			Total	13	

Question			Answer/Indicative content	Marks	Guidance
4		i	1 mark for each set of 2 moves • 02174356 01274356 (1) • 01247356 01234756 (1) • 01234576 • 01234567 (1)	3	Allow follow through if one move is incorrect

<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 per bullet to max 6

• Uses divide-and-conquer (1)
• First item becomes pivot / 2 is the pivot (1)
• Compare each item to the pivot (e.g. compare 0 to 2, then 1 to 2) (1)
• Make two lists, 1 with less than the pivot… (0, 1) (1)
• … 1 with more than the pivot (7,4,3,5,6) (1)
• Quick sort the new lists (1)
• Recombine the sub-lists (1)

OR
Example of alternative quicksort method
<table><tr><td></td><td>2 →</td><td>0</td><td>1</td><td>7</td><td>4</td><td>3</td><td>5</td><td>6 ←</td></tr><tr><td></td><td>2</td><td>0 →</td><td>1</td><td>7</td><td>4</td><td>3</td><td>5</td><td>6 ←</td></tr><tr><td></td><td>2</td><td>0</td><td>1 →</td><td>7</td><td>4</td><td>3</td><td>5</td><td>6 ←</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>7 →</td><td>4</td><td>3</td><td>5</td><td>6 ←</td></tr><tr><td>Swap 7 and 6</td><td>2</td><td>0</td><td>1</td><td>6 →</td><td>4</td><td>3</td><td>5</td><td>7 ←</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>6 →</td><td>4</td><td>3</td><td>5 ←</td><td>7</td></tr><tr><td>Swap 5 and 6</td><td>2</td><td>0</td><td>1</td><td>5 →</td><td>4</td><td>3</td><td>6 ←</td><td>7</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>5</td><td>4 →</td><td>3</td><td>6 ←</td><td>7</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>5</td><td>4</td><td>3 →</td><td>6 ←</td><td>7</td></tr><tr><td></td><td>2</td><td>0</td><td>1</td><td>5</td><td>4</td><td>3</td><td>6 ← →</td><td>7</td></tr></table>marks for:

• Uses divide-and-conquer (1)
• Highlight first list element as start pointer, and last list element as end pointer
• Repeatedly compare numbers being pointed to…
• …if incorrect, swap and move end pointer
• …else move start pointer
• Split list into 2 sublists
• Quick sort each sublist
• Repeat until all sublists have only 1 number
• Combine sublists</td><td>6</td><td>If no description i.e. the candidate has just shown the quick sort, max 4 marks.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
			Total	9	

Question			Answer/Indicative content	Marks	Guidance
5			**Mark Band 3 - High level** **(7–9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of how bubble sort works and Big O complexity; 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. *There is a well-developed line of reasoning which is clear and logically structured.* *The information presented is relevant and substantiated.* **Mark Band 2 - Mid level** **(4–6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of of how bubble sort works and Big O complexity; 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. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 - Low Level** **(1–3 marks)** The candidate demonstrates a **basic** knowledge of of how bubble sort works and Big O complexity with limited understanding shown; the material is basic and contains some inaccuracies.	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1: Knowledge and Understanding** **Indicative content** • Description of bubble sort: ◊Starting at the beginning of the list Items are swapped with their neighbour if they are out of order. ◊Each pair of neighbours is checked in order. ◊When a swap is made a flag is set. ◊If at the end of the list the flag has been set the flag is unset and the algorithm starts from the beginning of the list again. ◊When the algorithm gets to the end of the list and the flag is unset the list is sorted and the algorithm finishes. • $O(n^2)$ denotes as the data size increases the time the list takes to sort increases in a quadratic manner. • O(1) denotes the space used is constant **AO2: Application** • As data set gets bigger, bubble sort's time gets larger at an increasing rate.. • Complexity doesn't denote the actual time but the order with which the time / space grows. • O(1) space complexity means no matter how big the data set becomes the amount of space (extra to the data itself) remains the same. • $O(n^2)$ time complexity means as n increases time increases by n^2 / if n doubles the time taken is squared. • Bubble sort can be tweaked with improvements (e.g. checking one less item per iteration and alternating sorting directions). • These optimisations don't change the complexity. IT will run a little quicker on smaller sets but time taken increases rapidly with data size.

Question			Answer/Indicative content	Marks	Guidance
			The candidates make 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. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* 0 marks No attempt to answer the question or response is not worthy of credit.		• When choosing an algorithm we may also want to take into account the average and best case scenarios. (in this case they are also the same for both algorithms.) AO3: Evaluation • The algorithms may have the same time complexity but this does not mean they take the same time to execute on the same data set. • Insertion sort generally performs quicker than bubble sort and is therefore preferable. (Neither scale well however.) • Both algorithms have a space complexity of O(1). This is because both algorithms are inplace (i.e. all sorting takes place within the actual data). • Both have a time complexity of $O(n^2)$ as a consequence of their nested loops. (NB last two points are only likely to appear in the very highest mark answers.) Examiner's Comment: Most candidates achieved some credit, especially for a description of the bubble sort. Fewer candidates could compare the relative merits of both bubble and insertion sort in terms of the best / average / worse case.
			Total	9	