4.1 Decision Maths: Algorithms, Sorting and Bin Packing: 1-hour Pack

Total Marks

EDEXCEL 9FM0 · Calculator allowed · about 60 minutes

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 The list 40, 1, 24, 34, 50, 55, 44, 28 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 1 is 7 marks)

---

2 The list 48, 38, 40, 43, 22, 12, 49, 52 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 2 is 7 marks)

© Revision Library | revisionlibrary.co.uk

3 The list 56, 10, 39, 2, 27, 11, 14 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 3 is 7 marks)

---

4 The list 14, 2, 7, 15, 28, 51, 54, 16 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 4 is 7 marks)

© Revision Library | revisionlibrary.co.uk

5 The list 3, 23, 46, 36, 54, 27, 43, 58, 55 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 5 is 7 marks)

---

6 The list 7, 56, 49, 51, 31, 59, 28, 55, 9 is to be sorted into ascending order using a bubble sort. In each pass, adjacent items are compared from the left and swapped if they are in the wrong order.

(a) Show the list after the first pass and state the number of swaps made in the first pass. **(3)**

(b) Show the list after the second pass. **(2)**

(c) State the number of passes needed to sort the list completely, not counting a final pass in which no swaps are made, and the total number of swaps. **(2)**

(Total for Question 6 is 7 marks)

© Revision Library | revisionlibrary.co.uk

7 This question is about determining the order of an algorithm from timed test results.

(a) Ravi times how long Algorithm 1 takes to run on lists of different sizes, n. His results are shown below. **(3)**

n = 10, time = 8 ms
n = 20, time = 32 ms
n = 40, time = 128 ms
n = 80, time = 512 ms

Use a doubling ratio test (comparing the run time each time n doubles) to estimate the order of Algorithm 1, giving your answer in the form $O(n^k)$.

n	Time (ms)
10	8
20	32
40	128
80	512

(b) Ravi also tests Algorithm 2, obtaining the results below. **(2)**

n = 5, time = 4 ms
n = 10, time = 32 ms
n = 20, time = 256 ms

Use the same method to estimate the order of Algorithm 2.

n	Time (ms)
5	4
10	32
20	256

(c) A third algorithm has order O(n). It takes 15 ms to process a list of 200 items. Estimate how long it would take this algorithm to process a list of 800 items, showing your reasoning. **(2)**

(Total for Question 7 is 7 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 4.1 Decision Maths: Algorithms, Sorting and Bin Packing: 1-hour Pack

Question 1

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 1, 24, 34, 40, 50, 44, 28, 55	**A1**
(a) 5 swaps	**B1**
(a) Answer: 1, 24, 34, 40, 50, 44, 28, 55; 5 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 1, 24, 34, 40, 44, 28, 50, 55	**A1**
(b) Answer: 1, 24, 34, 40, 44, 28, 50, 55	
(c) 5 passes	**B1**
(c) 10 swaps in total	**B1**
(c) Answer: 5 passes, 10 swaps	

Question 2

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 38, 40, 43, 22, 12, 48, 49, 52	**A1**
(a) 5 swaps	**B1**
(a) Answer: 38, 40, 43, 22, 12, 48, 49, 52; 5 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 38, 40, 22, 12, 43, 48, 49, 52	**A1**
(b) Answer: 38, 40, 22, 12, 43, 48, 49, 52	
(c) 5 passes	**B1**
(c) 12 swaps in total	**B1**
(c) Answer: 5 passes, 12 swaps	

Question 3

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 10, 39, 2, 27, 11, 14, 56	**A1**
(a) 6 swaps	**B1**
(a) Answer: 10, 39, 2, 27, 11, 14, 56; 6 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 10, 2, 27, 11, 14, 39, 56	**A1**
(b) Answer: 10, 2, 27, 11, 14, 39, 56	
(c) 3 passes	**B1**
(c) 13 swaps in total	**B1**
(c) Answer: 3 passes, 13 swaps	

Question 4

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 2, 7, 14, 15, 28, 51, 16, 54	**A1**
(a) 3 swaps	**B1**
(a) Answer: 2, 7, 14, 15, 28, 51, 16, 54; 3 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 2, 7, 14, 15, 28, 16, 51, 54	**A1**
(b) Answer: 2, 7, 14, 15, 28, 16, 51, 54	
(c) 3 passes	**B1**
(c) 5 swaps in total	**B1**
(c) Answer: 3 passes, 5 swaps	

Question 5

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 3, 23, 36, 46, 27, 43, 54, 55, 58	**A1**
(a) 4 swaps	**B1**
(a) Answer: 3, 23, 36, 46, 27, 43, 54, 55, 58; 4 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 3, 23, 36, 27, 43, 46, 54, 55, 58	**A1**
(b) Answer: 3, 23, 36, 27, 43, 46, 54, 55, 58	
(c) 3 passes	**B1**
(c) 7 swaps in total	**B1**
(c) Answer: 3 passes, 7 swaps	

Question 6

(a) compares adjacent pairs from the left, swapping when the left item is larger	**M1**
(a) list after pass 1: 7, 49, 51, 31, 56, 28, 55, 9, 59	**A1**
(a) 6 swaps	**B1**
(a) Answer: 7, 49, 51, 31, 56, 28, 55, 9, 59; 6 swaps	
(b) repeats the process, ignoring the final item (already in place)	**M1**
(b) 7, 49, 31, 51, 28, 55, 9, 56, 59	**A1**
(b) Answer: 7, 49, 31, 51, 28, 55, 9, 56, 59	
(c) 7 passes	**B1**
(c) 19 swaps in total	**B1**
(c) Answer: 7 passes, 19 swaps	

© Revision Library | revisionlibrary.co.uk

Question 7

(a) correctly calculates the ratio of successive run times: 32/8 = 4, 128/32 = 4, 512/128 = 4 **M1**

(a) identifies that the ratio is consistently 4 each time n doubles **A1**

(a) correctly concludes the order is $O(n^2)$, since $4 = 2^2$ oe **A1**

(a) Answer: Algorithm 1 has order $O(n^2)$.

(b) correctly calculates the ratio of successive run times: 32/4 = 8, 256/32 = 8 **M1**

(b) correctly concludes the order is $O(n^3)$, since $8 = 2^3$ oe **A1**

(b) Answer: Algorithm 2 has order $O(n^3)$.

(c) recognises that 800 is 4 times 200, and that for an O(n) algorithm run time scales by the same factor as n **M1**

(c) 60 ms cao **A1**

(c) Answer: 60 ms.

© Revision Library | revisionlibrary.co.uk