inkMaths

ANSWER BOOK

Algorithms, sorting and bin packing

Decision Mathematics 1 · Year FM

6 questions · 24 marks

NAVY EDITION

SECTION A · Warm-up

A1 A finite sequence of precise instructions that solves a problem and always terminates. Its order says how the number of operations grows with the size n of the problem. An order n^2 algorithm takes four times as long on a list twice as long. B1 for the definition, B1 for the order. A complete definition names precision and termination; a vague 'a set of steps' gives neither. [2]

A2 Pass 1 compares 5 with 12, then 12 with 3, 9 and 1, so the 12 is carried to the end: **5,** 3, 9, 1, 12. Pass 2: **3,** 5, 1, 9, 12. Pass 3: **3,** 1, 5, 9, 12. Pass 4: **1,** 3, 5, 9, 12. Each pass leaves one more item fixed at the right-hand end, so after four passes on five items nothing remains to compare and the list is sorted. M1 for a correct first pass, A1 A1 for the later passes. Write the whole list after every pass. The intermediate lists are the answer here, not just the sorted result: a sort is a procedure, and showing it is showing the work. [3]

SECTION B · Standard

B1 The middle item of the whole list is 3, so it is the first pivot. Items smaller than it go to the left in their existing order and larger ones to the right, giving 1 **| 3 |** 5, 12, 9. The middle item of 5, 12, 9 is 12, giving 5, 9 **| 12 |** with nothing to its right. Of 5, 9 the right-hand middle is 9, giving 5 **| 9**. The sublist 1 is a single item and is already in place. Final order **1,** 3, 5, 9, 12, in three pivot steps against bubble sort's four passes. M1 for the first pivot and split, A1 for the resulting sublists, M1 for the second round of pivots, A1 for the sublists there, A1 for the final order. Ring each pivot as it is chosen and keep every sublist in view. A finished list with no working scores one mark at most. [5]

B2 The total is 45, so the lower bound is 45/10 = 4.5, rounded up to **5** bins. First fit fills bin 1 with 3 and 6, opens bin 2 for the 4 and later adds the 5, opens bin 3 for the 8 and later adds the 2, and gives the 7 and the 9 a bin each. The 1 goes back into bin 1. Packing (3, 6, 1), (4, 5), (8, 2), (7), (9), so **5** bins. M1 A1 for the lower bound, M1 A1 for the packing. That equals the lower bound, so this packing is optimal and no further work is needed. [4]

B3 Placing the large items first leaves the small ones to fill the gaps they create, whereas first fit can commit a bin to small items and then have no room for a large one. Neither looks ahead, so both can be beaten by a packing that sacrifices an early fit for a better later one; finding the optimum in general is a much harder problem. B1 for placing the large items first, B1 for the way first fit can block a bin, B1 for neither algorithm looking ahead. [3]

SECTION C · Stretch

C1 First fit: the 6 opens bin 1 and the 1 joins it; the second 6 opens bin 2 and its 1 goes back into bin 1; the third 6 opens bin 3 and its 1 again goes into bin 1. The bins so far are (6, 1, 1, 1), (6), (6), and no bin has room for a 7, so each 7 opens a bin of its own: **6** bins. [7]

The total is 42, so the lower bound is $42/10 = 4.2$, rounded up to **5** bins.

To show five is impossible, look at the six items of size 6 or more. Any two of them total at least 12, which exceeds the bin size, so no bin can hold two of them. Six such items therefore need six separate bins, and **6** is optimal.

M1 A1 for the first fit packing, B1 for six bins, M1 A1 for the lower bound, M1 A1 for the impossibility argument.

Falling short of the lower bound proves nothing on its own. An argument of this kind, counting items too large to share, is what turns a packing into a proved optimum.