Exercise walk-through

10.1
Sorting and packing algorithms

Pearson Edexcel · International A-Level · Further Mathematics

7. Critical path analysis

Critical paths

Precedence table to activity-on-arc network; dummies carry dependencies. Forward pass takes maxima (earliest event times), backward pass minima (latest). Float = latest finish − earliest start − duration; critical activities float zero. The example A(3), B(4), C(4), D(1), E(5) has critical path A–C–E = 12 days and D floats 2; Gantt bars run from earliest starts.

Check yourself

Check yourself

Bubble and quick sorts pass by pass; first-fit, decreasing and full-bin placements; Kruskal and Prim (matrix form too); Dijkstra with permanence order and route recovery; odd-node pairings for the postman; TSP bounds both sides with nearest neighbour; dummies, forward and backward passes, floats, Gantt charts; LP formulation and vertex solutions with integer checks.

Exercises 2.1·2.3

2.1 Bubble sort 5, 1, 4, 2, showing the list after each pass. [3]

2.2 Quick sort 1, 6, 3, 5, 7, 2, 9, stating each pivot and the list after each partition. [4]

2.3 Bin capacity 8; items 5, 3, 4, 2, 6, 1. Apply first-fit, showing each placement, and state the number of bins used. [3]

Exercises 2.1·2.3

2.1 Bubble sort 5, 1, 4, 2, showing the list after each pass. [3]

2.1 Pass 1: 5>1 swap, 5>4 swap, 5>2 swap → 1, 4, 2, 5. Pass 2: 4>2 swap → 1, 2, 4, 5. Pass 3: no swaps → sorted, stop. Final: 1, 2, 4, 5.

2.2 Quick sort 1, 6, 3, 5, 7, 2, 9, stating each pivot and the list after each partition. [4]

2.2 Pivot (position 4 of 7) = 5: less = 1, 3, 2; greater = 6, 7, 9, giving 1, 3, 2, **5**, 6, 7, 9. Sublist 1, 3, 2 (pivot 3): items less than 3, in order, are 1, 2 and none are greater, so the sublist becomes 1, 2, **3**; the pair 1, 2 (pivot 1) gives **1**, 2. Sublist 6, 7, 9 (pivot 7): less = 6, greater = 9 – already sorted. Final list: 1, 2, 3, 5, 6, 7, 9.

2.3 Bin capacity 8; items 5, 3, 4, 2, 6, 1. Apply first-fit, showing each placement, and state the number of bins used. [3]

2.3 5→b1, 3→b1 (8), 4→b2, 2→b2 (6), 6→b3, 1→b2 (7): bins (5,3), (4,2,1), (6) – 3 bins.

Exercise 3.1

3.1 *(Original.)* A courier has parcels of weights 9, 4, 6, 2, 8, 5, 3, 7 (kg) and boxes of capacity 15 kg. (a) Use the first-fit algorithm to pack the parcels, listing the contents of each box in order. (b) Use the first-fit decreasing algorithm to pack them. (c) Show that the total weight implies at least three boxes, and comment on the two algorithms' results. [9]

Exercise 3.1

3.1 *(Original.)* A courier has parcels of weights 9, 4, 6, 2, 8, 5, 3, 7 (kg) and boxes of capacity 15 kg. (a) Use the first-fit algorithm to pack the parcels, listing the contents of each box in order. (b) Use the first-fit decreasing algorithm to pack them. (c) Show that the total weight implies at least three boxes, and comment on the two algorithms' results. [9]

3.1 (a) First-fit (capacity 15): 9→b1, 4→b1 (13), 6→b2, 2→b2 (8), 8→b3, 5→b2 (13), 3→b3 (11), 7→b4: boxes (9,4), (6,2,5), (8,3), (7) – 4 boxes. (b) Decreasing order 9, 8, 7, 6, 5, 4, 3, 2: 9→b1, 8→b2, 7→b3, 6→b1 (15), 5→b2 (13), 4→b3 (11), 3→b4 (3), 2→b4 (5): boxes (9,6), (8,5), (7,4), (3,2) – 4 boxes. (c) Total weight $9+4+6+2+8+5+3+7=44$, and $\frac{44}{15}=2.93\ldots$ so at least 3 boxes; both algorithms use 4 – first-fit decreasing is not guaranteed to beat first-fit, though it usually does.

Exercise 3.2

3.2 *(Original.)* The list 1, 6, 3, 4, 8, 2, 9 is to be sorted into ascending order. (a) Use the quick sort algorithm, stating the pivot at each stage and the list after each partition. (b) State one advantage quick sort has over bubble sort on large lists. [8]

Exercise 3.2

3.2 *(Original.)* The list 1, 6, 3, 4, 8, 2, 9 is to be sorted into ascending order. (a) Use the quick sort algorithm, stating the pivot at each stage and the list after each partition. (b) State one advantage quick sort has over bubble sort on large lists. [8]

3.2 (a) Pivot (position 4 of 7) = 4: less in order = 1, 3, 2; greater = 6, 8, 9, giving 1, 3, 2, **4**, 6, 8, 9. Sublist 1, 3, 2 (pivot 3): less = 1, 2; greater = none → 1, 2, **3**; then (1, 2) pivot 1: **1**, 2. Sublist 6, 8, 9 (pivot 8): less = 6; greater = 9 → 6, **8**, 9. Final: 1, 2, 3, 4, 6, 8, 9. (b) On large lists quick sort is generally much faster (fewer comparisons on average), because each partition splits the problem instead of moving items one position per comparison.

Local review guidance: pivots stated and partitions listed; placements shown item by item with running totals; lower-bound comparison quoted. This is authored classroom guidance.