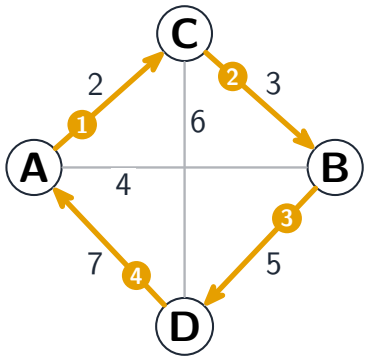
Exercise walk-through

# 10.2
# Minimum spanning trees and shortest paths

Pearson Edexcel · International A-Level · Further Mathematics

# Figure: The travelling salesman problem




nearest neighbour from A:
A–C–B–D–A $= 2+3+5+7 = 17$
upper bound

delete A: MST of B, C, D
$= 3+5 = 8$
plus AC + AB $= 2+4$:
lower bound $14$

$14 \leq$ optimal tour $\leq 17$

- Nearest neighbour gives an upper bound, the deleted-vertex MST a lower bound: the optimal tour lies between 14 and 17
- Match each part of the diagram to the text.

# Critical path analysis (2)

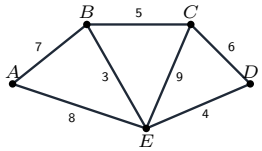
- A **critical path** is a path of activities with zero **total float**. Its length is the shortest time in which the project can be completed.
- Earliest times: A ends at 3, B at 4, C at 7, D at 5. E starts at $\max(7, 5) = 7$ and ends at 12.
- The **critical path** is A–C–E, and the project takes 12 days.
- D must finish by 7, so its total float is $7 - 4 - 1 = 2$ days. B must finish by 6, so its float is $6 - 0 - 4 = 2$ days.

# Key terms

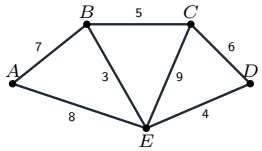
- **Minimum spanning tree**: a tree connecting all the vertices with the least total weight.
- **Dijkstra's algorithm**: an algorithm for the shortest path from one vertex to the others.
- **Route inspection problem**: finding the shortest closed route that uses every edge.

# Exercises 2.1·2.2

**2.1** Edges AB 3, AC 1, BC 2, CD 4, AD 5: find the MST by Kruskal, listing edges in order of choice, and give its total weight. [3]

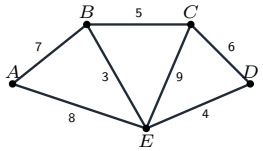



**2.2** For the same network, run Prim from vertex B and show the MST is the same. [3]



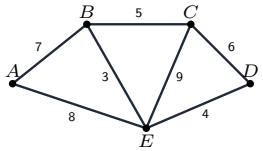
# Exercises 2.1·2.2

**2.1** Edges AB 3, AC 1, BC 2, CD 4, AD 5: find the MST by Kruskal, listing edges in order of choice, and give its total weight. [3]




**2.1** Sorted: AC 1, BC 2, AB 3, CD 4, AD 5. Take AC(1) ✓; BC(2) ✓ (B joins); AB(3) – cycle (A–C–B) reject; CD(4) ✓ (D joins); AD(5) – cycle reject. MST = AC + BC + CD, total 7.

**2.2** For the same network, run Prim from vertex B and show the MST is the same. [3]

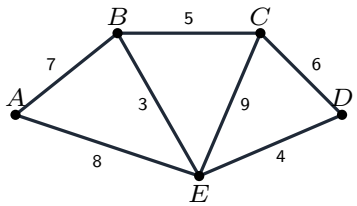



**2.2** Prim from B: cheapest edge from {B} is BC(2); from {B,C} cheapest to a new vertex is CA(1); from {B,C,A} cheapest to D is CD(4). Same tree, total 7 ✓.

exercises.lol exercises.com.cn

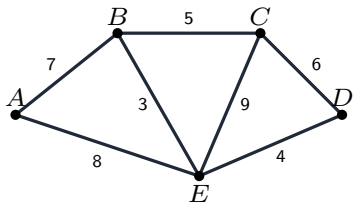
# Exercise 2.3

**2.3** Edges AB 2, BC 3, CD 1, AD 4, AC 2: use Dijkstra from A to find the shortest distance to D, showing each label as it becomes permanent, and state the route. [4]



# Exercise 2.3

**2.3** Edges AB 2, BC 3, CD 1, AD 4, AC 2: use Dijkstra from A to find the shortest distance to D, showing each label as it becomes permanent, and state the route. [4]

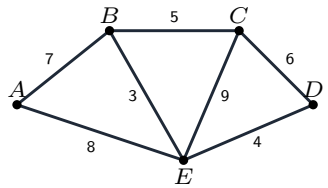



**2.3** A = 0 permanent. Neighbours: B 2, C 2. B permanent (2) – D has no edge from B, so no update. C permanent (2; the tie with B may be broken in either order); update D: $2 + 1 = 3$, smaller than the direct AD 4. D permanent (3): shortest A to D = 3, route A–C–D ($2 + 1 = 3 < 4$ ✓).

exercises.lol exercises.com.cn

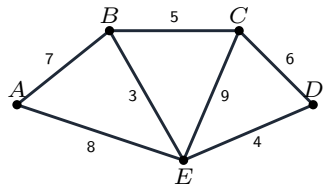
# Exercise 3.1

**3.1** *(Original.)* A network connects four villages P, Q, R, S with edge weights PQ 7, PR 3, PS 8, QR 2, RS 6, QS 5 (km). (a) Use Kruskal's algorithm to find the minimum spanning tree, listing the edges in the order chosen and rejecting cycles explicitly. (b) Use Prim's algorithm from P to find the same tree. (c) State the total weight of the MST and explain why both algorithms must agree. [8]



# Exercise 3.1

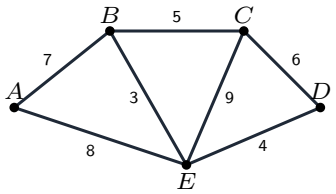
**3.1** *(Original.)* A network connects four villages P, Q, R, S with edge weights PQ 7, PR 3, PS 8, QR 2, RS 6, QS 5 (km). (a) Use Kruskal's algorithm to find the minimum spanning tree, listing the edges in the order chosen and rejecting cycles explicitly. (b) Use Prim's algorithm from P to find the same tree. (c) State the total weight of the MST and explain why both algorithms must agree. [8]




**3.1** (a) Sorted: QR 2, PR 3, QS 5, RS 6, PQ 7, PS 8. Take QR(2) ✓; PR(3) ✓ (P joins); QS(5) ✓ (S joins); RS(6) – cycle reject; PQ(7) – cycle reject; PS(8) – cycle reject. MST = QR + PR + QS, total 10. (b) Prim from P: PR(3); from {P,R} cheapest new edge RQ(2); from {P,R,Q} cheapest to S is QS(5). Same tree, total 10 ✓. (c) Both construct a spanning tree of least total weight; any MST of a network has the same minimum total, so the totals agree (the edges can differ only when weights tie).

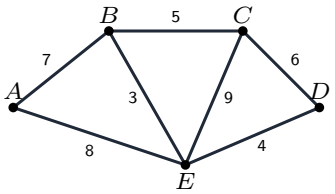
# Exercise 3.2

**3.2** *(Original.)* With the same network, use Dijkstra's algorithm from P to find the shortest route from P to S, showing the order in which labels become permanent and the final route with its length. [6]



# Exercise 3.2

**3.2** *(Original.)* With the same network, use Dijkstra's algorithm from P to find the shortest route from P to S, showing the order in which labels become permanent and the final route with its length. [6]




**3.2** P = 0 permanent. Update: R 3, Q 7, S 8. R permanent (3); update Q: 3 + 2 = 5 (smaller than 7), S: 3 + 6 = 9 (8 stays). Q permanent (5); update S via Q: 5 + 5 = 10 > 8, so S keeps 8. S permanent (8). Shortest P to S = 8, route P–S (the alternatives P–R–S = 3 + 6 = 9 and P–R–Q–S = 3 + 2 + 5 = 10 are both longer ✓).

Local review guidance: edges chosen in weight order with cycle rejections shown; Prim grows by cheapest new-vertex edge, not cheapest free edge; Dijkstra labels in permanence order with updates shown. This is authored classroom guidance.